Les tubes de mouvement : nouvelle représentation pour les séquences d'images

Matthieu Urvoy

▶ **To cite this version:**

Matthieu Urvoy. Les tubes de mouvement : nouvelle représentation pour les séquences d'images. Traitement du signal et de l'image [eess.SP]. INSA de Rennes, 2011. Français. NNT : . tel-00642973

HAL Id: tel-00642973

https://theses.hal.science/tel-00642973

Submitted on 20 Nov 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse

SPECIMEN NON IMPRIMABLE

THÈSE INSA Rennes
sous le sceau de l'Université Européenne de Bretagne
pour obtenir le titre de
DOCTEUR DE L'INSA DE RENNES
Spécialité : Traitement du signal et de l'image

présentée par
Matthieu URVOY
ECOLE DOCTORALE : Matisse
LABORATOIRE : IETR UMR CNRS 6164
Institut d'Electronique et des Télécommunications de Rennes

Les tubes de mouvement : nouvelle représentation pour les séquences d'images

Motion tubes: a new representation for image sequences

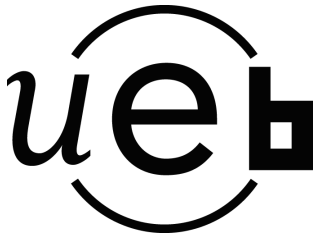
Thèse soutenue le 29/03/2011
devant le jury composé de :

Patrick LE CALLET
Professeur à Polytech Nantes / président
Henri NICOLAS
Professeur à l'Université de Bordeaux I / rapporteur
Frédéric JURIE
Professeur à l'Université de Caen / rapporteur
Olivier DÉFORGES
Professeur à l'INSA de Rennes / directeur de thèse
Marie BABEL
Maître de conférence à l'INSA de Rennes / co-encadrante
Stéphane PATEUX
Ingénieur de recherche à Orange Labs / tuteur industriel

Les tubes de mouvement : nouvelle représentation pour les séquences d'images

Motion tubes: a new representation for image sequences

Matthieu URVOY



En partenariat avec

In partnership with

Acknowledgements, remerciements

Si la thèse n'est jamais que le premier pas dans une carrière de chercheur, c'est avant tout le résultat d'un long cheminement. Un cheminement pavé de rencontres et d'échanges, avec collègues, famille et amis, qui tous, ont contribué à la concrétisation de ces travaux. Ces quelques lignes leurs sont destinées.

Tout d'abord, je tiens bien évidemment à remercier mes cinq encadrants sans qui ces travaux n'auraient jamais vu le jour. Si cinq encadrants sont parfois à l'origine de divergences, ils sont surtout une intarissable source d'idées et de conseils, qui, une fois confrontés et combinés, permettent d'avancer avec toujours plus de recul et d'efficacité.

A **Olivier**, merci pour toute la patience et la constance dont tu as su faire preuve pendant ces trois années et demi. A **Marie**, merci pour ton soutien et ta détermination, et pour la "grande soeur" que tu es devenue. A **Stéphane**, merci pour ta disponibilité, ta pédagogie, ton regard critique, et pour tout ce savoir encyclopédique que tu as mis à ma disposition pendant cette thèse. A **Nathalie** et **Muriel**, enfin, félicitations de la part de votre premier thésard pour votre suivi, vos conseils et votre écoute. Par dessus tout, merci à tous les cinq pour la confiance que vous avez eue en moi, et ce même dans les moments difficiles où moi-même je doutais.

Je tiens par ailleurs à remercier **M. Patrick Le Callet**, pour l'honneur qu'il m'a fait de présider le jury de thèse, ainsi que pour l'opportunité qu'il m'a offerte de poursuivre mes recherches au sein de son laboratoire. Je n'oublie pas **Messieurs Henri Nicolas** & **Frédérique Jurie** qui ont accepté la tâche délicate de rapporteurs, et je les remercie vivement pour leurs conseils et leurs encouragements.

Ces travaux ont été réalisé dans le contexte d'un contrat CIFRE. A ce titre, j'ai séjourné tantôt dans un laboratoire académique (l'IETR), tantôt dans une entreprise (France Télécom R&D – Orange Labs).

Merci à **Alexandre Nolle**, puis **Christine Marcatté** pour leur accueil au sein des laboratoires IRIS, puis OPERA, à Orange Labs. Un grand merci à toi, **Ludovic**, pour toute la disponibilité et l'écoute dont tu as fait preuve, malgré la multitude de sollicitations inhérentes au rôle de manager. Merci à toi, **Isabelle**, pour tes encouragements, ta bonne humeur, pour toutes ces discussions passionnantes que nous avons eues, et toutes les découvertes musicales et culturelles qui s'en sont suivies. **Matthieu**, **Thomas**, je ne vous oublie pas: collègues certes, mais désormais amis avant tout ; ces trois ans sont passés bien vite à vos côtés. Enfin, merci à vous tous, membres et anciens membres de CVA, pour votre accueil et pour votre dynamisme ; il fait bon travailler avec vous !

Merci également à tous les membres de l'équipe ARTIST de l'IETR ; votre accueil, comme d'habitude, a été des plus chaleureux. Si la vie de thésard n'est pas toujours simple, la compagnie de **Jonathan**, **Maxime**, **Médéric**, **Fabien**, **Jérôme**, **Matthieu**, **Emilie**, **Youssef**, **Clément**, **François** ... et tous les autres, a grandement contribué à sa réussite ! Je n'oublie pas non plus l'équipe enseignante, et en particulier ceux qui m'ont initié, à mon tour, à l'enseignement. Merci également à **Jocelyne** et **Aurore** pour toutes ces formalités administratives qu'elles m'ont épargnées.

Si le soutien de tous a permis à ces travaux d'aboutir, c'est aussi ma passion pour la recherche qui m'a animé pendant ces trois ans, et qui continue toujours de le faire. A ce titre, je tiens à remercier certains de mes professeurs qui m'ont inspiré. Je pense en particulier à **Olivier** et **Marie** (encore), à **Kidiyo**, et à **John Soraghan**.

Thanks to **Jim Davis** and his lazy cat for the everyday laugh.

Je tiens bien entendu à remercier tous mes proches pour leur soutien inconditionnel. **Maryse**, **Bernard**, merci à tous les deux pour avoir toujours cru en moi et m'avoir toujours poussé à aller plus loin. Et je n'oublie surtout pas de remercier mes amis : **Laurian** pour cet optimisme et cette insouciance que tu sais si bien communiquer ; **Ludo**, **Hélène**, **Stéphane**, **Emilie**, **Coralie** et **Victor** pour être toujours là dans les moments difficiles ; **Dalf** pour nos aventures en montagne, qui certes auront eu presque raison de nos genoux, mais qui restent avant tout d'excellents moments où les soucis s'effacent ; **Navin** for all the discussions we had about dealing with both a thesis and life issues at the same time ; **Binôme**, **Anthony**, **Arnaud**, **Marina**, **Billou** et tous les autres que je n'oublie pas.

Enfin, **Marie** (la mienne), merci du fond du cœur pour avoir su trouver la force de rester aux côtés d'un ours que ses recherches rendaient jour après jour de plus en plus taciturne, et qui grâce à toi est désormais "docteur en tubes" et parle encore un peu.

On veut toujours
Aller au-delà.

Au-delà de quoi,
On ne sait pas.

Au-delà
De ce qui est en soi ?

Au-delà
De ce qui n'y est pas ?

Au-delà de quelquechose
Qui n'existe pas ?

Eugène Guillevic, *Maintenant* (1993)

À Frédérique

Contents

Abbreviations

1G First Generation . . . 16
2G Second Generation . . . 13
3-PSNR Three-Component PSNR . . . 50
3-SSIM Three-Component SSIM . . . 50
3DTV Three-Dimensional TV . . . 50

ADPCM Adaptive DPCM . . . 6
AR Autoregressive . . . 6
ARMA Autoregressive Moving Average . . . 7
AS Analysis-Synthesis . . . 60
AVC Advanced Video Coding . . . 55
AWT Anisotropic Wavelet Transform . . . 16

B-frame Bidirectional frame . . . 52
BinDCT Binary DCT . . . 10
BMA Block Matching Algorithm . . . 38
BMC Block Motion Compensation . . . 30
BP Basis Pursuit . . . 17
BPDN Basis Pursuit De-Noising . . . 17

CABAC Context Adaptive Binary Arithmetic Coding . . . 58
CAVLC Context Adaptive Variable Length Coding . . . 58
CGI Control Grid Interpolation . . . 32
CGS Coarse Grain Scalability . . . 59
CIF Common Intermediate Format . . . 93
CRF Conditional Random Field . . . 33
CSF Contrast Sensitivity Function . . . 50
CWT Continuous Wavelet Transform . . . 12

DCT Discrete Cosine Transform . . . 10
DDCT Directional DCT . . . 13
DFB Directional Filter Bank . . . 15
DFD Displaced Frame Difference . . . 21
DFT Discrete FT . . . 9

DPCM Differential PCM . . . 6
DRB Digital Radio Broadcast . . . 1
DS Diamond Search . . . 40
DST Discrete Sine Transform . . . 10
DVB Digital Video Broadcast . . . 1
DVD Digital Versatile Disc . . . 56
DVQ Digital Video Quality . . . 50
DWT Discrete Wavelet Transform . . . 12

EBCOT Embedded Block Coding with Optimal Truncation . . . 62
EPZS Enhanced Predictive Zonal Search . . . 40
ESCOT Embedded Subband Coding with Optimized Truncation . . . 62
ESS Extended Spatial Scalability . . . 59

FFT Fast FT . . . 9
FGS Fine Grain Scalability . . . 59
FMO Flexible Macroblock Ordering . . . 57
FOCUSS FOcal Underdetermined System Solver . . . 17
FREXT Fidelity Range EXTensions . . . 11
FR Full Reference . . . 49
FS Full Search . . . 37
FSBMC Fixed Size BMC . . . 30
FSS Four-Step Search . . . 39
FT Fourier Transform . . . 9
FTV Free viewpoint TV . . . 51

GenLBT Generalized LBT . . . 11
GenLOT Generalized linear-phase LOT . . . 11
GMF Global Matched Filter . . . 17
GOP Group Of Pictures . . . 51
GRF Gibbs Random Field . . . 33
GSM Global System for Mobile communications . . . 1

HD High-Definition . . . 48
HD-DVD High Definition DVD . . . 60
HDMI High Definition Multimedia Interface . . . 59
HDTV HD Television . . . 60
HEVC High Efficiency Video Coding . . . 1
HLBT Hierarchical LBT . . . 11
HVS Human Visual System . . . 50

I-frame Intra-coded frame . . . 52

IBMCTF Inband MCTF . . . 19
IEC International Electrotechnical Commission . . . 55
IFS Iterated Function System . . . 8
IG-OBMC Irregular Grid OBMC . . . 45
IIR Infinite Impulse Response . . . 6
IntDCT Integer DCT . . . 10
IP Internet Protocol . . . 1
IPTV IP Television . . . 1
ISO International Organization for Standardization . . . 55
ITU International Telecommunication Union . . . 55
ITU-T ITU Telecommunication Standardization Sector . . . 55

JPEG Joint Photographic Expert Group
JPEG-XR JPEG eXtended Range . . . 11
JSVM Joint Scalable Video Model . . . 135
JTC Joint Technical Committee . . . 55
JVT Joint Video Team . . . 55

KLT Karhunen-Loève Transform . . . 10

LAOTMC Locally Adaptive OTMC . . . 92
LAR Locally Adaptive Resolution . . . 53
LBT Lapped Biorthogonal Transform . . . 11
LM Levenberg-Marquardt . . . 43
LMedS Least-Median-of-Squares . . . 36
LOT Lapped Orthogonal Transform . . . 11
LTE Long Term Evolution . . . 1

M-JPEG Motion JPEG . . . 47
M^2S Multiscale Modular Similarity . . . 50
M^3S Multiscale Modular Maxima Similarity . . . 50
MA Moving Average . . . 6
MAE Mean Absolute Errors . . . 35
MCTF Motion Compensated Temporal Filtering . . . 18
MDCT Modified DCT . . . 11
MDST Modified DST . . . 11
MGS Medium Grain Scalability . . . 59
MHDFD Multi-Hypothesis DFD . . . 40
MHE Multi-Hypothesis Expectation . . . 40
MLBT Modulated LBT . . . 11
MLT Modulated Lapped Transform . . . 11
MMO Memory Management Operation . . . 58

MOS Mean Opinion Score . . . 49
MP Matching Pursuit . . . 17
MPEG Motion Picture Expert Group . . . 49
MRA Multi-Resolution Analysis . . . 12
MRF Markov Random Field . . . 33
MSE Mean Squared Errors . . . 35
MS-SSIM Multi-Scale SSIM . . . 50
MVC Multiview Video Coding . . . 58
MVFAST Motion Vector Field Adaptive Search Technique . . . 40

NAL Network Abstraction Layer . . . 56
NAR Non-linear AR . . . 7
NARMA Non-linear ARMA . . . 7
NLBT Nonuniform LBT . . . 11
NLVE Non-Linear Video Editing . . . 47
NQM Noise Quality Measure . . . 50
NR No-Reference . . . 49

OBMC Overlapped BMC . . . 30
OBME Overlapped BME . . . 41
OOMP Optimized Orthogonal Matching Pursuit . . . 17
OSA Orthogonal Search Algorithm . . . 40
OTMC Overlapped Tube Motion Compensation . . . 78
OTS One at a Time Search . . . 40

P-frame Predicted-frame . . . 52
P2P Peer To Peer . . . 1
PCA Principal Component Analysis . . . 10
PCM Pulse Code Modulation . . . 57
PDC Pel Distortion Classification . . . 36
PEVQ Perceptual Evaluation of Video Quality . . . 50
PMVFAST Predictive MVFAST . . . 40
POD Proper Orthogonal Decomposition . . . 10
PSNR Peak Signal to Noise Ratio . . . 49
PVQM Perceptual Video Quality Measure . . . 50

QCIF Quarter CIF
QMF Quadrature Mirror Filter . . . 61
QoE Quality of Experience . . . 51
QoS Quality of Service . . . 51
QP Quantization Parameter . . . 54

RDO Rate-Distortion Optimization . . . 51
RGB Red Green Blue . . . 47
RMS Root Mean Square . . . 50
ROI Region Of Interest . . . 57
RR Reduced Reference . . . 49
RTP Real Time Protocol . . . 56

SA-DCT Shape-Adaptive DCT . . . 13
SAD Sum of Absolute Differences . . . 35
SAE Sum of Absolute Errors . . . 35
SATD Sum of Absolute Transformed Differences . . . 35
SB-FSSQ Subband-Finite State Scalar Quantizer . . . 61
SC Subcommittee . . . 55
SCGI Switched CGI . . . 43
SD Standard Definition . . . 48
SDMCTF Spatial Domain MCTF . . . 19
SEA Successive Elimination Algorithm . . . 37
SNR Signal-to-Noise Ratio . . . 49
SOBMC Switched OBMC . . . 43
SPIHT Set Partitioning In Hierarchical Trees . . . 17
SPSNR Synthesis PSNR . . . 94
SSD Sum of Squared Differences . . . 35
SSE Sum of Squared Errors . . . 35
SSIM Structural Similarity Image Metric . . . 50
SVC Scalable Video Coding . . . 58

TBCGI Tri-Bilinear CGI . . . 32
TDAC Time-Domain Aliasing Cancellation . . . 11
TDL 2-Dimensional Logarithmic . . . 39
TMC Tube Motion Compensation . . . 78
TSS Three-Step Search . . . 39
TvC Threshold vs Contrast . . . 50

UIQ Universal Image Quality . . . 50
UMCTF Unconstrained MCTF . . . 19
UMTS Universal Mobile Telecommunications System . . . 1

VCEG Video Coding Expert Group . . . 55
VCL Video Coding Layer . . . 56
VGA Video Graphics Array
VIF Visual Information Fidelity . . . 50
VLC Variable Length Code . . . 110
VOD Video On Demand . . . 1

VQ Vector Quantization 61
VQM Video Quality Metric 50
VSBMC Variable Size BMC 30
VSNR Visual SNR 50

WFT Windowed FT 11
WG Working Group 55
WHT Walsh-Hadamard Transform 10
WMSE Weighted MSE 50
WSNR Weighted SNR 50

Notations

General

$\mathbb{E}^n$	:	N-dimensional Euclidian space
$\mathcal{C}^n$	:	Space of functions that are n times continuously derivable
$\mathcal{C}^\infty$	:	Space of functions that are indefinitely derivable
$\text{Card}\,(S)$	:	Cardinal of set S
$\mathcal{L}(\mathbb{R})$	:	Set of integrable (or Lebesque-measurable) functions
$\mathcal{L}_2(\mathbb{R})$	:	Set of square-integrable (or Lebesque-measurable) functions
(x, y, z)	:	Coordinates whose components are x, y and z
$[x, y, z]^T$	:	Vector whose components are x, y and z
$\langle \overrightarrow{u}, \overrightarrow{v} \rangle$	:	Inner product (generalization of the dot product) of vectors $\overrightarrow{u}$ and $\overrightarrow{v}$
$\boldsymbol{A}$	:	Matrix $\boldsymbol{A}$
$\boldsymbol{A}^T$	:	Transpose of matrix $\boldsymbol{A}$
$\text{Tr}\,(\boldsymbol{A})$	:	Trace of matrix $\boldsymbol{A}$
$\det(\boldsymbol{A})$	:	Determinant of matrix $\boldsymbol{A}$
a_{ij}	:	Coefficient of matrix $\boldsymbol{A}$ at row i and column j
∇f	:	Gradient of function f
$\Delta f = \nabla^2 f$	:	Laplacian of function f
$\overrightarrow{\nabla} \cdot \overrightarrow{a}$	:	Divergence of vector $\overrightarrow{a}$
$\text{rot}\left(\overrightarrow{a}\right)$	:	Curl of vector $\overrightarrow{a}$

Image processing

I_t	:	Image at time instant t in a sequence
$I_t(x, y)$	:	Luminance value at position (x, y) in I_t
$\mathcal{S}_\mathcal{I}$	:	A sequence of images
$\boldsymbol{\Omega}$	:	Support of an image
$\overrightarrow{\nabla I}$	:	Spatial gradient of image I
$P_t(x, y)$	:	A point from image I_t located at coordinates (x, y)
$\mathcal{P}_t(x, y)$	:	Pixel colour intensity at coordinates (x, y) in I_t
$\mathcal{O}$	:	An object contained by an image or a scene
$\mathcal{R}$	:	A region of an image
$\mathcal{B}$	:	A block of an image
$\mathcal{Q}$	:	Quantization operator

Motion estimation

$\overline{I_t}$	:	Synthesized image at time instant t
$\widehat{I_t}$	:	Motion compensated image at time instant t
$\widetilde{I_t}$	:	Decoded image at time instant t
$\overrightarrow{d}$	:	Displacement vector of the motion field
$\overrightarrow{v}$	:	Speed vector of the optical flow
τ_s	:	Temporal sampling period
$\mathcal{M}_t$	:	A meshing of the image I_t
$w_{i\to j}$	:	Warping operation from time instant t_i to time instant t_j
$\mathcal{V}$	:	The motion field
$\mathcal{V}(x,y)$	:	The value of the motion field at position (x,y)

Motion tubes

$\mathcal{M}_\mathcal{T}$	:	A *motion tube*
$\mathcal{T}$	:	Textural information of a *motion tube*
$\mathcal{L}$	:	Lifespan of a *motion tube*
$\mathcal{W}$	:	Motion information of a *motion tube*
$\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t)$	:	Support of a *motion tube* at time instant t
$\mathcal{G}_{\mathcal{M}_\mathcal{T}}(t)$	:	Center of gravity of the support $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t)$ at time instant t
$T_{\mathcal{M}_\mathcal{T}}(t)$	:	Trajectory of a *motion tube*
$\mathcal{C}_{\mathcal{M}_\mathcal{T}}$	:	*Motion tubes* coding operator
$\mathcal{C}_\mathcal{T}$	:	*Motion tubes* texture coding operator
$\mathcal{C}_\mathcal{L}$	:	*Motion tubes* lifespan coding operator
$\mathcal{C}_\mathcal{W}$	:	*Motion tubes* deformation coding operator
$\mathcal{R}_{\mathcal{M}_\mathcal{T}}$	:	*Motion tubes* rendering operator
$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$	:	A family of *motion tubes* referenced at time instant t_{ref}
${}^N\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_i,\ldots,t_j)$	:	A *bi-family* of *motion tubes* $\left\{\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_i),\ldots,\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_j)\right\}$ at temporal level N
$\bigcup$	:	*Motion tube* composition operator
${}^\mathcal{F}\bigcup$	:	*Motion tubes* family composition operator

Introduction

Digital signal transmission has been first introduced to secure highest-level allied communications during World War II. The underlying secure system, known as SIGSALY, mainly relied on a modulation technique introduced by Ralph Miller and Bob Badgley: the Pulse Code Modulation (PCM). By regularly sampling an analogue signal and quantizing resulting samples to a series of symbols into a numeric code, the PCM turned out to be a pioneering technique in digital signal processing field. Later, C. E. Shannon would laid the foundation stone of modern information theory, with the introduction of his famous Nyquist-Shannon theorem, providing the minimum sampling rate which should be used to achieve perfect reconstruction of a signal. From there, considerable efforts have been put into finding ways to reduce the bandwidth required to properly transmit a signal, and the field of its application considerably broadened throughout the years.

Nowadays, digital communications have become essential in numerous domains, including Internet Protocol (IP) traffic, mobile communications - Global System for Mobile communications (GSM), Universal Mobile Telecommunications System (UMTS) and Long Term Evolution (LTE) standards -, Digital Radio Broadcast (DRB) and Digital Video Broadcast (DVB), etc. Amongst ever increasing amounts of information being transmitted across these various communication channels, a large part of the traffic consists of video information. By 2014, it is expected that the global IP traffic will quadruple, and the sum of all forms of video information - IP Television (IPTV), Video On Demand (VOD), Internet, Peer To Peer (P2P) - will represent 91% of this traffic [CIS10]. In mobiles, during the same period, the video traffic is expected to increase by 6000% [CIS10]. In both cases, the increase in bandwidth provided by future optical networks and mobile communication standards will not be able to compensate for such an increase.

Henceforth, the need for increased compression abilities is more than ever critical for the establishment of future communication systems. Despite the current efficiency of state of the art video compression schemes, it is crucial for research to focus on ways to further improve video compression mechanisms. In this connection, forthcoming video compression standard, namely High Efficiency Video Coding (HEVC), has been required to provide 50% in bitrate savings compared to the actual standard ITU-T H.264/AVC.

Problem statement

During the last 30 years, researches have been actively focusing on ways to compact the spatio-temporal information hold by image sequences. As a result, numerous video compression schemes have been provided, with increasing compression abilities. Typically, the compression is carried out by several decorrelation mechanisms in charge of detecting and capturing the different forms of redundancies an image sequence may hold.

Three main approaches have been proposed throughout the years. On one hand, predictive decorrelation techniques rely on a prediction model to guess future values of a signal. Assuming that the prediction model is appropriate, it may be expected that most of the images contents can be blindly retrieved from already available samples. On the other hand, transform-based techniques can also be used to describe the images in alternative spaces which may naturally provide a compact representation. Finally, synthetic techniques can also be used whenever the signal contents can be specifically modelled.

In image sequences, however, the decorrelation along the temporal axis is drastically improved when motion compensation is used. Indeed, successive images from a sequence can generally be interpreted as successive projections of an original 3D scene onto the image plane at different time instants. In that capacity, modelling and estimating the motion information is a central problem in video compression. This gave birth to numerous motion compensation schemes, in particular those providing an optimal trade-off between motion representation abilities and amount of motion information have been extensively used in video compression.

In the end, the combination of multiple decorrelation techniques, along with an efficient motion compensation, led to numerous video compression schemes. Classically, the standard approach to video compression locally process both temporal and spatial dimensions. Images are partitioned into blocks, typically called *macroblocks*; each of which may then be decorrelated from either a spatial approach (also known as *intra* coding) or a temporal approach (also known as *inter* coding). As for the temporal axis, it is processed on a frame by frame basis. Locally describing the images contents is an attractive attribute, however standard approaches may suffer from their inability to provide a continuous temporal representation. As a consequence, this may penalize their abilities to extract the redundancies along the temporal axis.

Besides standardized video compression schemes, numerous disruptive approaches have also been provided throughout the years. In particular, 3D Wavelets have been extensively investigated as they provide a continuous representation of both spatial and temporal domains. They later inspired analysis-synthesis schemes which generally provide a continuous representation of the textural information across time. Despite all their attractive features however, they were a limited success with normalization committees which mostly advocate the classic low-computational block-based approach. In particular, the inability of analysis-synthesis techniques to provide a perfect representation was highly criticized.

An original approach to video compression: the *motion tubes*

From the above discussion, it appears that an ideal compression system might be obtained by locally decorrelating the spatial contents, and globally processing the information along the temporal axis. In practice, this can be obtained by continuously tracking local areas of an image sequence across time.

Putting aside the traditional frame-*macroblock* paradigm, it is proposed to construe image sequences as a collection of spatio-temporal units which deform and move across time and space. These structures are called *motion tubes*, and rely on three types of information:

- a **textural information** accounting for the textural contents of the area being tracked;
- a **lifespan information** controlling the time interval during which the patch can be tracked;
- a set of **motion descriptors** modelling the deformation of the patch across time and space.

Problem raised

From such a spatio-temporal structure, the resulting representation exhibits the temporal persistence of the textural information. However, this raises several problems:

- how can various **deformations** that a patch of texture undergoes can be **modelled** in a **simple** way?
- unlike other approaches to video compression, *motion tubes* cannot guarantee the images to be **perfectly** and/or **completely reconstructed**: which additional mechanisms need to be used to address these issues?
- it is expected that an image area may be accounted by several *motion tubes*: which **life and death** mechanisms should be used to limit these redundancies, and remove or shorten unneeded *motion tubes*?

Proposed solutions: key contributions

Once the notion of *motion tube* has been further introduced, their motion, textural and lifespan parameters, along with their effects on the provided representation, will be investigated within this thesis. Key contributions brought by this manuscript include:

- a **simple** and **effective motion model** which allies both the advantages of disruptive approaches and the relative simplicity of the standardized approach;
- several mechanisms in charge of **improving** the **synthesized textures** will address a critical problem: *motion tubes* may fail at accounting for the whole textural information of an image sequence. As a consequence, it is essential for unregistered areas to be handled through additional mechanisms;
- a **life and death** mechanism in charge of detecting, shortening and/or removing redundant or inneficient *motion tubes* in regards to representation and compression.

Organization of the thesis

This manuscript is organized into three main parts.

- At first, **chapters 1, 2 and 3** will dwell upon a large range of **antecedent signal and image processing techniques** commonly used in **video compression**. In particular, decorrelation techniques, motion compensation techniques, and existing video compression schemes will be reviewed.
- Then, **chapter 4** will introduce a **spatio-temporal structure** that can be used to compactly describe the spatio-temporal contents of an image sequence, by exhibiting the temporal persistence of the textural information. This structure will be called a ***motion tube*** .
- Finally, **chapters 5, 6 and 7** will respectively focus on the **motion information**, the **textural information** and the **lifespan** of the *motion tubes*: three types of parameters driving their behaviour across time and space.

Antecedent approaches dedicated to the representation and the compression of signals

Chapter 1 will focus on a wide range of techniques aiming at capturing the correlation hold by image sequences. Typically, they fall into three categories: predictive, synthetic and transform decorrelation techniques. In image sequences, these techniques often focus either on the spatial domain or on the temporal domain. In the spatial domain, on one hand, transform techniques have been extensively used to describe the spatial features hold by an image. Some of them locally model the geometry, thus adapt to the geometrical contents. On the other hand, in the temporal domain, direct applications of classical decorrelation techniques often fail at efficiently capturing the redundancies. As a consequence, they generally rely on the motion information to further decorrelate the images along the temporal axis.

Chapter 2 will be dedicated to motion compensation, and its underlying problems: the representation and the estimation of the motion information which links successive images from a sequence. Typically, image sequences can be interpreted as the projection of a 3D scene onto the image plane. Under this assumption, the real 3D motion is projected onto the image plane as well; it can then be measured from the optical flow corresponding to the displacement of the pixellic intensities. From this optical flow, the motion can be conceptualized in various ways, and numerous motion models have been provided throughout the years including dense motion fields, parametric motion models and models based on the deformation of geometrical shapes. Besides the problem of representation, a large number of estimation techniques have been provided to fit the parameters of these models to the actual motion information. In an eye to compression, chapter 2 will highlight two motion models which have been extensively used in video compression: blocks and meshes. Dedicated motion estimation techniques will also be investigated.

Chapter 3 will focus on video compression schemes. Typically, they combine a set of decorrelation techniques; in particular, spatial decorrelation techniques are called *intra* coding techniques, while temporal decorrelation techniques are called *inter* coding techniques. At first, chapter 3 will review several processing tools on which most compression schemes rely. Then, state of the art ITU-T H.264/AVC video compression standard will be briefly reviewed. Finally, disruptive approaches will also be investigated, whether they rely on Wavelets or analysis-synthesis paradigms.

Introducing the notion of motion tube

Chapter 4 will introduce the concept of *motion tubes*: the ability to track a patch of texture across time and space. Indeed, the key feature of a *motion tube* is its ability to exhibit the temporal persistence of a local patch of texture. This spatio-temporal structure will be driven from three sets of parameters: its motion parameters, its textural information, and its lifespan.

Investigating the threefold nature of the motion tubes: motion, texture and lifespan

Chapter 5 will propose a motion model able to describe the deformation of a set of *motion tubes*. Inspired from both standard approaches and analysis-synthesis approaches, the proposed model is able to describe various deformations, while sticking to a low-computational block-based representation. In between block-based and mesh-based motion compensation techniques, it will be provided with a set of additional features, including a spatio-temporal regularization of the motion, the ability to connect or disconnect adjacent *motion tubes*, and a local adaptation of the dimensions of the *motion tubes* to the images contents.

Chapter 6 will focus on the the textural information of the *motion tubes* and the images they are able to synthesize. Any change in the textural information of a patch being tracked should ideally be accounted for. In addition, synthesized images may not be completely reconstructed Several mechanisms will be provided to improve the quality of synthesized textures. At first, a relatively crude mechanism will be in charge of accounting for the textural changes across the temporal dimension: even though the textural information is persistent, it may be still be affected by various phenomena (illumination changes, resolution losses). Then, a second mechanism will be in charge of registering as much textural information as possible, such that synthesized images do not show unpredicted areas. Finally, as a last resort, an inpainting mechanism will be provided to complete the synthesized images.

Chapter 7 will investigate several life and death mechanisms in charge of discarding *motion tubes* which provide poor contributions to the reconstruction of images, including those which are redundant. As a first step towards a *motion tube* quality/efficiency measurement, it will be proposed to compare the performances of the *motion tubes* with those of the state of the art ITU-T H.264/AVC compression standard. From this competition, H.264/AVC's decisions will be used to discard and/or shorten any *motion tube* whose use is not significant enough.

Conclusion and perspectives

Finally, the conclusion will highlight and discuss key contributions the motion tubes bring to the field of video representation and compression. In addition, various perspectives regarding further enhancements and additional functionalities of *motion tubes* will also be provided.

Chapter 1

Capturing the correlation within image sequences

With increasing needs to send data over ever evolving transmission channels, numerous challenges emerged. Among them, the need for fast and accurate ways to transmit an information has been widely considered. In particular, compression aims at finding the most compact way to express a signal, thus requiring less bandwidth and time for it to be transmitted.

Prior to the compacting operation, the representation used to model the signal takes an essential part into the compression process. Indeed, signals typically hold a significant amount of redundancies which can be removed through *decorrelation* techniques: the signal correlation level is closely related to the amount of redundancies it contains.

The natures of the redundancies which can be found into a signal are extremely various: no decorrelation techniques is able to detect them all. Consequently, numerous decorrelation techniques have been proposed throughout the years, and target a large range of redundancies. Compression and transmission schemes now unveil different processes, each of which processing a specific type of correlation. As an illustration, figure 1 represents a generic video coding system:

- the **input signal** is likely to hold **distinctive geometrical** and/or **temporal features** which, once they have been detected, can be advantageously used to drive a set of decorrelation tools;
- the **transform** operation processes the signal through a set any analysis functions: resulting coefficients express whether or not the signal contains any of the corresponding patterns. Periodic variations, for instance, can be exhibited by transforming the original signal into a frequency or a time-frequency domain;
- the **prediction** operation is used to compensate for local spatio-temporal correlations;
- the **quantization** operation is generally a lossy operation which reduces the dynamic of input coefficients, hence requiring a smaller amount of information to be transmitted;
- finally, the **entropy coding** operation takes advantage from the statistical correlations of resulting coefficients and further compacts the overall information.

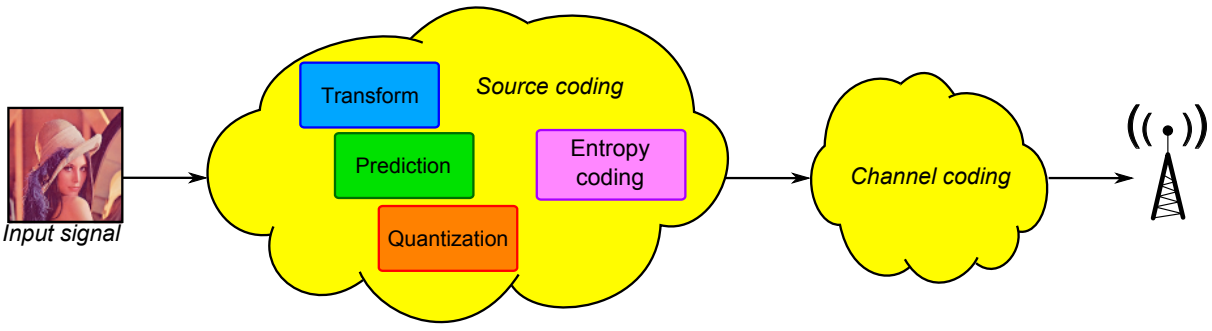


Figure 1: *A traditional transmission scheme*

Typically, decorrelation techniques can be categorized into three families, according to the nature of the targeted redundancies. The three of them are listed below.

Predictive techniques build a prediction of the current sample from its available neighbourhood. The prediction model is designed from spatial and/or temporal correlation patterns.

Synthetic techniques propose to describe the signal from dedicated synthesis models. Synthetic representations are not sample-based, and generally do not provide a perfect reconstruction of the signal. In images, for instance, the textures may be synthesized from polynomial or fractal (self-similar) descriptions.

Transform techniques describe the signal into another space within which its representation is naturally more compact. For instance, any signal may be described from a smaller amount of significant coefficients in an appropriate transformed space, in comparison to its original formulation. The compactness, however, has been subject to numerous discussions throughout the years, and may be interpreted in various ways (it is not exclusively related to the amount of significant coefficients).

In image and video compression, each of these techniques have been considered. Predictive and synthetic approaches have been used to decorrelate the signals across both space and/or time. Both these approaches will be briefly reviewed in section 1.1, which will also explain why statistical coding techniques have not been included within this review. Transform decorrelation techniques, in image compression, have been mostly dedicated to the representation of geometrical patterns, hence focusing on the spatial dimension. Section 1.2 will review a large number of transforms which are commonly used in still image compression. Then, section 1.3 will raise the problem of the decorrelation along the temporal axis, and how essential is the role played by the motion information. Finally, section 1.4 will conclude the chapter.

1.1 Predictive, synthetic, and statistical techniques: a brief overview

1.1.1 A glimpse at predictive decorrelation techniques

Predictive decorrelation techniques aim at extracting the most obvious redundancies of a signal from its original expression: a series of samples. Indeed, successive samples of a signal are very likely to be strongly correlated, especially near-stationary signals. From a prediction model and previously processed samples, predictive decorrelation techniques are used to predict future samples. Synchronizing the prediction model between the transmitter and the receiver, only the prediction residues (*i.e.* the difference between the actual value of the current sample and its prediction) need to be transmitted. Taking advantage from the Pulse Code Modulation (PCM), this is known as Differential PCM (DPCM).

A basic approach, for instance, predicts the current sample by its predecessor. This technique has been introduced by Cassius Chaplin Cutler [Cut52] in 1952, then improved by E. Cummiskey in 1973 through the Adaptive DPCM (ADPCM) [CJF73]. Since then, numerous alternatives have been proposed and further improve the prediction model. Most of them consider a finite set of previous samples as parameters for their prediction model; this set is known as the *causal neighbourhood*. In images, for instance, the causal neighbourhood is typically a set of pixels located on top and left of the current pixel (provided that images are raster scanned from left to right and top to bottom). Figure 1.1.1 shows a typical causal neighbourhood.

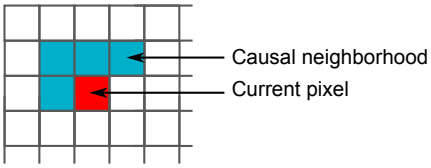


Figure 1.1.1: *A classic causal neighbourhood used for image pixel prediction*

From a set of past samples, Autoregressive (AR) models have been widely proposed as a way to model the signal future behaviour [PW83, Li07]. An auto-regressive model of order p is defined as

$$s[t] = c + \sum_{i=1}^{p} \phi_i s[t-i] + \varepsilon_t \tag{1.1}$$

where $\phi_1, \ldots, \phi_p$ are the parameters of the AR model, c is a constant offset and ε_t is white noise. Omitting the constant c, such a model can be seen as the output of an all-pole Infinite Impulse Response (IIR) filter. The Moving Average (MA)

extension of the latter model leads to the Autoregressive Moving Average (ARMA) model, which also takes into account its previous predictions to build the current one. ARMA models are generally fitted to the reference data from a least-square regression: MA parameters are optimized in order to minimize the prediction errors/residues. Non-linear versions of these models have also been proposed to catch up with non-linear stationarities: Non-linear AR (NAR) and Non-linear ARMA (NARMA) models.

Alternatively, probabilistic approaches were also proposed to predict the value of a signal. Markov chains provide a powerful tool to model the behaviour of a random variable. Naturally, they have been extended to two-dimensional random variables: Markov fields, Bayesian networks and Gibbs fields have been used to model the spatial behaviour of such a random process [KS80]. In such case, images are considered to be a two-dimensional set of random events.

However, both AR and probabilistic models may end up into very computationally demanding prediction structures. In practice, simpler predictors are rather used in image processing. The median predictor, for instance, is one of the most popular prediction structure: it predicts the current sample by the median sample from the considered neighbourhood. Some other advanced prediction schemes have been dedicated to images. X. Wu's predictor [Wu97] builds a prediction from several steps which successively enrich the considered neighbourhood. The final step, in particular, benefits from both causal and anti-causal prediction samples. Along with hierarchical decorrelation approaches such as Gaussian and Laplacian pyramids [AAB+84], X. Wu's predictor has been shown to be particularly efficient [BDR05].

More recently, texture and structure synthesis approaches have been proposed to predict whole image areas. Using advanced interpolation mechanisms, inpainting [CS02, LSW07a] and template matching [TBS06, GWL08], in particular, are able to provide an accurate prediction of both the textural and the structural information of unknown areas. The latter are interpolated from available textural and structural patterns of neighbouring areas.

In general, lossless still images representation and compression schemes mostly rely on a set of several consecutive predictive decorrelation techniques. Lossy representations, on the other hand, typically hybridize predictive techniques with alternative techniques to provide further decorrelation abilities. Among them, transform approaches are very popular; they will be tackled in section 1.2.

1.1.2 A brief overview of synthetic decorrelation techniques

Synthetic decorrelation techniques are based on a synthetic model which is fit to the image from an appropriate parametrization. Fractals, in particular, rely on a self-similar parametrization of the images In any case, synthetic approaches do not consider the image as a set of pixels, but directly focus on its contents. Consequently, they are generally unable to provide a perfect representation of the signal, and simply provide a good approximation.

1.1.2.1 Model based techniques

Most of the time, the contents of an image can be described through the parametrization of a specific model. Due to the potential complexity of these contents, images are often partitioned into appropriate blocks or regions; each of these are then parametrized by a particular model. In [KaMK85], regions are synthesized by appropriate two-dimensional polynomial functions. In [Car88], images are represented by the shapes and the average intensities of segmented regions. In [DR99, DBBR07, BD09], blocks of pixels are seen as the combination of a uniform function and a textural residue.

1.1.2.2 Fractal-based techniques

One can easily observe that parts of an image often resemble to other parts of the same image. This is known as self-similarity and can be favourably exploited to represent an image in a compact manner. Indeed, fractal-based representations only require *fractal-codes* to represent an image, such that it can be synthesized at any scale without suffering from any loss of sharpness, which usually occurs when images are upsampled. However, it is generally impossible to provide an exact representation of the original image.

The research on fractal image representation are consequences of works on chaos and fractals in the years 1978-1985. A common metaphor to fractal representation is the following [Gal03]: consider a special type of photocopying machine which reduces the image to be copied by half and reproduces it three times on the copy according to a given pattern. The output of the machine is then iteratively taken back as an input. What is to be observed ? Whichever the original image, assuming that the exact same reproducing pattern is used for each iteration, one can observe that the system converges towards the same final image. This reproducing pattern thus fully determines what the final image will look like.

As a consequence, any image can be considered as the output of such a system, and may be fully characterized by an appropriate set of reproducing patterns. Indeed, it has been shown that complex images could be obtained in only a few iterations using simple transformations as reproducing patterns: this has been motivating the hypothesis that any image could be represented by a relatively small set of transformations, thus offering a compact representation. Any contractive transformation may be employed: $\mathcal{W}$ is said to be contractive if, for any two points P_1 and P_2,

$$d\left(\mathcal{W}(P_1), \mathcal{W}(P_2)\right) < d(P_1, P_2)\,, \tag{1.2}$$

where $d(P_1, P_2)$ is the distance between points P_1 and P_2. In other words, this simply indicates that, by repeatedly applying this transform on P_1 and P_2, their projections converge towards the same point. Otherwise, this would lead to images whose size is infinite. In practice, affine transforms able to skew, stretch, rotate, scale and translate have proved to be sufficient enough to generate complex images.

Mathematically, the decomposition of an image into a set of transformations is described as an Iterated Function System (IFS) [BJM+88, BS90]. An IFS is a set of N contractive transformations $\{\mathcal{W}_1 \dots \mathcal{W}_N | \mathcal{W}_i : \mathbf{\Omega} \to \mathbf{\Omega}\}$ (also called *mapping functions*). $\mathbf{\Omega} \subset \mathbb{R}^2$ is the support of the image. According to the set of mapping functions, the IFS describes a set A_∞ known as the fixed point of the Hutchinson operator $\mathcal{H}$ [Hut81], which is defined as

$$\mathcal{H}(A) = \bigcup_{i=1}^{N} \mathcal{W}_i(A), \qquad A \subset \mathbf{\Omega}. \tag{1.3}$$

Thus, taking $I = A_\infty$ as the input image, successive iterations of $A_{k+1} = H(A_k)$ will converge towards I for any non-empty initial set A_0: I is a union of mapped copies of itself. In other words, it is possible to reconstruct the image I from any initial image using the appropriate transformation $\mathcal{H}$. A practical approach to fractal representation is given in [Fis92]:

1. the input image (or *range image*) is partitioned into blocks $\mathcal{B}_{\text{range},i}$ of size $s \times s$;
2. the image is down-sampled by a factor 2 and low-pass filtered; this new image is called the *domain image*;
3. for each $\mathcal{B}_{\text{range},i}$ of the range image, a similar block $\mathcal{B}_{\text{domain},i}$ is searched within the down-sampled *domain image*;
4. a mapping (or transform) function $\mathcal{W}_i$ is chosen such that $\forall i$, $\mathcal{B}_{\text{domain},i} = \mathcal{B}_{\text{range},i}$.

However, such an approach is very computationally demanding of the encoder side; and practical implementations failed at providing a reasonable computing speed. Further improvements to fractal representations yet proposed faster ways to find appropriate transformations to represent an image using fractals [Kom97, HKFT00, WJC05, RKDM06, WJH07].

1.1.3 Why statistical coding techniques should not be considered as decorrelation techniques

In parallel, statistical coding techniques (also known as source coding or entropy coding techniques) have been derived from the information theory introduced by C. E. Shannon as a way to detect and remove the statistical correlations of the signals. C. E. Shannon's *source coding theorem* establishes the limits to possible data compression as the entropy of the signal [Sha48]. Since then, numerous source coding techniques have been proposed: recent source coding techniques now practically reach the entropy limit. Among entropy coding algorithms, one could mention in particular Huffman codes [Huf52] which build a binary tree of symbols, Golomb-Rice run-length encoders [Gol66] which use symbols whose length is a function of the value to encode, Elias gamma codes [Eli75], and arithmetic codes [Jel68, Ris76, RLJ79, LJ84] which use floating point intervals to describe a sequence of symbols.

Whichever the technique considered, decorrelating a signal changes its intrinsic nature into a representation which exhibits a specific type of redundancies. On that account, statistical coding (also known as entropy coding) will not be classified as a pure decorrelation technique. Indeed, statistical coding (or statistical decorrelation) does not change the nature of the signal: it simply restructures the symbols to be transmitted in order to minimize their entropy. As a consequence, statistical coding will not be further reviewed in this chapter, which only focuses on pure decorrelation techniques.

1.2 Spatial correlations: a focus on transform techniques

Previous section has introduced the notion of decorrelation and has provided a brief overview of predictive and synthetic approaches. Predictive techniques, on one hand, provide an efficient mechanism which locally models the signal, but fail

at detecting global redundancies which require the signal to be considered in its entirety. Synthetic techniques, on the other hand, consider the signal its entirety, but fail at providing an accurate representation of the signal.

In between those two approaches, transform-based techniques provide a global approach to the decorrelation, while providing an accurate representation. From a set of analysis functions, they enable the signals to be processed in alternative domains, wherein the correlations are much more obvious. Ideally, resulting transform coefficients form a more compact description of the original signal. For these reasons, transforms have been extensively used to decorrelate images. In an eye to compression, an idealistic transform should:

1. fit the correlation patterns of the signal;
2. lead to a compact representation, with respect to the way compactness has been defined (repartition of the energy throughout the spectrum, variance of the transform coefficients, etc);
3. be near-lossless or lossless invertible to guarantee the ability to retrieve the original signal expression into temporal or spatio-temporal series of samples;
4. finally, be as low-computational as possible.

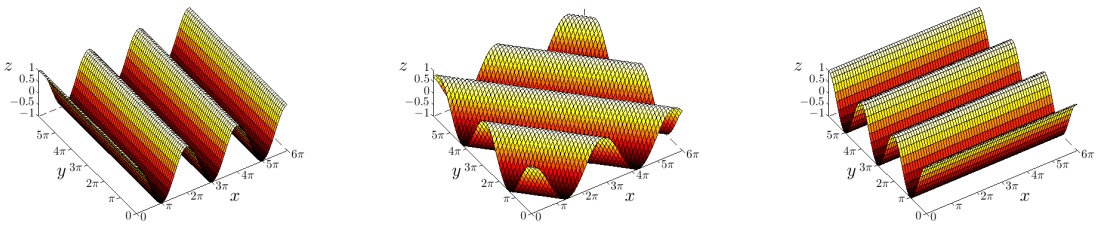
Historically, transforms have been introduced through the frequency analysis, whose foundation stone was laid by the Fourier Transform (FT). The latter will be briefly reviewed in section 1.2.1. Numerous other transforms have then been built after the FT, and have been extensively reviewed throughout the years. At first, numerous transforms employing a fixed set of basis functions were proposed; some of them will be reviewed in section 1.2.2. Then, a second generation of transforms were designed to take into account the image geometry; they will be reviewed in section 1.2.3.

1.2.1 The emergence of the frequency analysis through the Fourier Transform

The FT [Bra00] laid the foundation stone of the frequency analysis, also known as the Fourier analysis. FT was later linked to similar transforms such as Laplace Transform and its discrete version, the Z-transform [RZ52]. Since then, FT became very popular as a way to identify the harmonics of a periodic signal. The Fourier Transform of an integrable function $f(t) \in \mathcal{L}(\mathbb{R})$ is given by

$$\mathcal{F}[f(t)](\omega) = \int_{-\infty}^{+\infty} f(t) e^{-i\omega t}\, dt \tag{1.4}$$

where ω is the angular frequency of the harmonic on which the signal is projected. $\mathcal{L}(\mathbb{R})$ is the set of integrable (or Lebesque-measurable) functions. For two-dimensional signals, the FT basis functions $e^{-i(\omega_x x+\omega_y y)} = e^{i\rho(x\cos\theta+y\cos\theta)}$ correspond to plane waves which propagate in direction θ and oscillate at frequency ρ (see figure 1.2.1). In order to handle discrete signals (of limited duration), FT has been discretized into the Discrete FT (DFT) and expresses the spectrum of a signal by a finite number of harmonics. In practice, the DFT is generally implemented with fast algorithms such as the Fast FT (FFT) [CLW69]. However, the FT suffers from the complexity of its basis functions and does not provide a very compact representation.



(a) *Horizontal,* $\theta = 0$ **(b)** *Diagonal,* $\theta = \pi/4$ **(c)** *Vertical,* $\theta = \pi/2$

Figure 1.2.1: *Real part of the Fourier Kernel for three different directions*

1.2.2 First generation transforms: no adaptation to the image geometry

Originally, transforms were applied to blocks of pixels: similarly to the predictive approach to decorrelation, they only provided a local representation of the signal. However, splitting the signal into disjoint blocks would introduce some blocking artefacts; as a workaround, some lapped transforms were designed to be applied to overlapping blocks, which drastically reduced blocking artefacts. Besides frequency transforms built after the FT, the time-frequency transform domain was also investigated through the Wavelet transform, which can be applied on entire signals at once, thus removing blocking effects, but introducing some ringing artefacts.

Alternatively, an optimal transform known as the Karhunen-Loève Transform (KLT) [Kar47, Loè78] is used to express a signal in the space of its principal components. In other words, the basis functions used to model the signal are given by the eigenvectors of its covariance matrix. In other fields, it is also known as Principal Component Analysis (PCA) [Pea01], Hotelling transform, or even Proper Orthogonal Decomposition (POD). It has been shown to be, so far, the most efficient transform in terms of compaction. However, the KLT is very computationally demanding and was not used in practice until recently. Until then, it was considered as a compaction reference, that other transforms could be compared to. For these reasons, the KLT will not be further reviewed.

Whichever way, all these transforms share a common limitation: they cannot take into account the image geometry. In particular, they are blind to the directionality of geometrical patterns hold by the images. In practice, however, these transforms still provide great decorrelation abilities to numerous compression schemes. Section 1.2.2.1 will review the most popular block-based transform: the Discrete Cosine Transform (DCT). Then, lapped transforms will be reviewed in section 1.2.2.2. Finally, the Wavelet transform will be reviewed in section 1.2.2.3.

1.2.2.1 The block-based Discrete Cosine Transform

Both the DCT [ANR74, RY90, Str99] and the Discrete Sine Transform (DST) were introduced as simplified versions of the FT. The DCT uses cosine basis functions, while the DST uses sine basis functions. The DCT, in particular, was first introduced by Ahmed *et al.* in [ANR74]. Its requires the signal to be symmetrical, thus cancelling the imaginary part of the FT: this artificially compacts the transform spectrum. In other words, the DCT basis functions are equal to the real part of the Fourier kernel (see figure 1.2.1). In addition, symmetrizing the signal removes the discontinuity problem at the signal boundaries.

Consequently, the DCT has become a very popular transform: its two-dimensional version has been extensively used in many still images coders (JPEG [ISO94a]) and video coders (MPEG-x [ISO93, ISO94b, ISO00a], H.26x [ITU90, ITU94, ITU95]). Usually, the DCT is applied on 8×8 blocks, i.e. an 8×8 matrix of pixel intensities. Under these circumstances, it returns an 8×8 matrix $\boldsymbol{C}$ whose DCT coefficients $c_{m,n}$ are given by

$$c_{m,n} = \alpha(m)\beta(n) \sum_{i=0}^{7} \sum_{j=0}^{7} \mathcal{B}_{i,j} \cos\left(\frac{\pi(2i+1)m}{16}\right) \cos\left(\frac{\pi(2j+1)m}{16}\right) \tag{1.5}$$

where $\mathcal{B}_{i,j}$ is the pixel (i, j) of the original block, and $\alpha(m)$ and $\beta(n)$ are scaling coefficients. Its original formulation given in equation (1.5) induces an important amount of computations. Furthermore, floating points computations may suffer from a lack of standardization homogenization, and successive rounding operations lead to a divergence between the coder and the decoder.

Consequently, integer approximations of the DCT, Integer DCT (IntDCT), were proposed to reduce its complexity and eliminate floating points computations. They rely on various factorizations of the floating point DCT matrix, and lead to integer approximations of the DCT:

- **direct approaches** factorize the DCT matrix into a product of sparse matrices, which enables the DCT [CSF77];
- **indirect approaches** ease the factorization process through the Walsh-Hadamard Transform (WHT) [CON00];
- **Alternative approaches** iteratively apply the indirect factorization: the coefficients of a $N \times N$ IntDCT are recursively computed from those of a $N/2 \times N/2$ IntDCT [CXL01, ZCBK01, WHS01, Abh02, COT+02, Abh03, HM03, PT03a, PT03b].

The IntDCT, for instance, introduces a lifting [Swe96] decomposition of the 8×8 DCT. Alternatively, the Binary DCT (BinDCT) [Tra99, LT00, Tra00, LT01] only requires binary operations to be performed . Both BinDCT and IntDCT proved to be as efficient as the standard DCT from a rate-distortion perspective. Alternatively, ITU-T H.264/AVC video

compression standard [JVT03, Ric03] applies an integer approximation of the floating point DCT on 4×4 blocks. Its matrix formulation is given by

$$\boldsymbol{C} = \boldsymbol{H}.\boldsymbol{B}.\boldsymbol{H}^T \qquad with \qquad \boldsymbol{H} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \tag{1.6}$$

where $\boldsymbol{B}$ is the matrix representation of the input block $\mathcal{B}$, and $\boldsymbol{H}$ is the transform matrix. This approximation of the DCT proved to be very efficient and low computational. Later, a simplified 8×8 integer matrix transform was introduced by H.264/AVC first amendment through the Fidelity Range EXTensions (FREXT) [STL04, JVT05].

1.2.2.2 Extension of the DCT to lapped transforms

Usual block-based transforms do not have any knowledge about the neighbourhood of the current block. Consequently, they introduce some blocking artefacts stemming from the block boundaries. As a solution, it was proposed to take the local neighbourhood of the current block into account to perform the transform (*e.g.* the Windowed FT (WFT) [HBS84]). In particular, *lapped transforms* use overlapping blocks as support for the transform, which reduces potential representation mismatches at the boundaries of the blocks. Enlarging the support of the transform requires an extended transform matrix to be designed. However, the extension should not introduce any redundancies in the representation.

The family of lapped transforms was first introduced by P. M. Cassereau *et al.* in [Cas85, CSJ88]. Later, J. P. Princen introduced the Modified DCT (MDCT) as a consequence of its work on Time-Domain Aliasing Cancellation (TDAC) in [PJB87]. The MDCT has been used in audio coding MPEG-1/2 Audio Layer 3 standards [ISO93, ISO94b]. There also exists an analogous transform, the Modified DST (MDST).

Later, Henrique S. Malvar and David H. Staelin finally provided in [MS89] a rigorous definition of lapped transforms as lapped extensions of the DCT: the Lapped Orthogonal Transform (LOT). Instead of using $N \times N$ blocks, the LOT uses $L \times L$ overlapping blocks, where $N < L \leq 2N$. In particular, when $L = 2N$, the maximum overlapping ratio is obtained and any pixel from the image plane belongs to up to 4 LOT blocks. Henrique S. Malvar later introduced the Modulated Lapped Transform (MLT) in which the basis functions are built from a modulated filter bank structure [Mal90].

However, such transforms do not provide a great increase in a rate-distortion context, while being very computationally demanding. As a consequence, they were never really considered as a potential alternative for a while. The breakthrough came from the Lapped Biorthogonal Transform (LBT) [Mal98] which is less complex than previous lapped transforms. Derived versions of the LBT have also been proposed: the Hierarchical LBT (HLBT) for the images, the Modulated LBT (MLBT) and the Nonuniform LBT (NLBT) for the audio [Mal98]. The LBT has been subject to numerous optimizations which proposed fast, integer and lifting version of this transform [Tra00, ZCC01, CZL02]. Lapped transforms have also been studied by Trac D. Tran and T. Nguyen who proposed the Generalized linear-phase LOT (GenLOT) and the Generalized LBT (GenLBT) [Tra00, ITN02].

Lapped transforms have later been conceptualized as a DCT transform coupled with a post-processing filter. However, such consideration implies that the post-processing is entirely dependent on the form of the DCT. This approach was later reversed and described as a pre-processing filter followed by a DCT transform. In such case, the pre-processing filters are independent from the transform [TT01, LTT01, TLT03, DT03]; this allowed to further optimize lapped transforms. Such an implementation of a lapped transform has been recently used in the JPEG eXtended Range (JPEG-XR) standard [ITU09].

1.2.2.3 Time-frequency representation: the Wavelet transform

The frequency representation describes a signal from a set of periodic waves which last over the whole signal domain. Non-stationary signals, however, generally do not hold such kind of harmonics; they are much more likely to hold local harmonics specifically located in time (or space) and frequency. The frequency representation will spread their discontinuities over a large number of harmonics, which will not end up in a compact representation. This is why, among other reasons, frequency transforms such as the DCT are applied to tiny blocks of pixels. Besides, harmonics that are neglected in the representation introduce some oscillations around the discontinuities, known as Gibbs phenomena [Wib48].

To address these issues, an idealistic transform should be able to account for both global and local spatio-temporal characteristics of a signal. In other words, it should be able to detect signal characteristic specifically located in time

(or space) and frequency. As a first step towards time-frequency representation, the WFT employs basis functions both localized in time and frequency. Its basis functions are given by the product of the Fourier basis function $e^{\iota\omega t}$ by a sliding window $g(t - t_c)$ successively centred around time instants t_c. In other words, the WFT is given by the Fourier Transform of the windowed function $f(t)g(t - t_c)$

$$\mathcal{F}[f(t)](\omega) = \int_{-\infty}^{+\infty} f(t)g(t - t_c)e^{-i\omega t}\, dt \tag{1.7}$$

where $g(t - m)$ may be, as proposed by Gabor in the 50's, a Gaussian window which only keeps the original signal in the neighbourhood of time instant t_c. Typically, the shape of the window $g(t - t_c)$ is fixed, which does not allow the WFT to grasp the characteristics of the signal at different scales.

Later, Wavelets were introduced as a way to perform a multi-scale time-frequency signal analysis. They were first introduced by J. Morlet and A. Grossmann in early 80's [GM84]. Rather than using a fixed window $f(t)g(t - t_c)$, Wavelets rely on a set of translated and dilated versions of an elementary function $\psi(t)$, known as the *mother Wavelet*

$$\psi_{a,b}(t) = a^{-1/2}\psi\left(\frac{t - b}{a}\right) \tag{1.8}$$

where $a > 0$ is the dilatation (scale) coefficient and b is the translation (shift) coefficient, and $\psi \in \mathcal{L}^2(\mathbb{R})$. Low frequency patterns, on one hand, are easily captured using large windows. High frequency patterns, on the other hand, are easily captured using narrow windows. As a consequence, it is naturally proposed to partition the time-frequency domain into bands whose duration is inversely proportional to the targeted frequency range. Similarly to the FT, the Continuous Wavelet Transform (CWT) is then defined as

$$\mathcal{W}[f(t)](a, b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t)\psi_{a,b}(t)\, dt =< \psi_{a,b}, f > \tag{1.9}$$

where $f \in \mathcal{L}^2(\mathbb{R})$ is a square-integrable function. The CWT has been later discretized into the Discrete Wavelet Transform (DWT).

Location and *regularity order* are the main characteristics to be considered when designing a set of Wavelets [Pey05]. The regularity order limits the set of regularities which can be modelled by the Wavelets. The time-frequency location enables the representation to capture specific oscillating phenomena located inside the considered window. Numerous Wavelet basis have been proposed through the years, both for the CWT (hermitian Wavelet, Mexican Hat Wavelet, Shannon Wavelet, etc) and the DWT (Daubechies Wavelets [Dau88], Haar Wavelets, Meyer Wavelets [Mey88], etc) versions of the transform.

Later, Daubechies [Dau88], Mallat [Mal89a, Mal00], Meyer [Mey90] and Sweldens [JS94] expressed the CWT as a form of Multi-Resolution Analysis (MRA). They split the time-frequency plane into a dyadic partition and introduce a scaling function $\phi(t)$ known as the *scaling function* to access the different resolution levels. For each resolution level j, the provided representation is refined from the information of resolution level $j - 1$: the Wavelet shows intrinsic *scalability* features.

Finally, Wavelets have also been expressed in terms of filter banks [Mal89a, Mal89b, Mey90]. The transform is then simply performed through an iteration of two filtering operations. At resolution level j:

1. a **low-pass** filter L_j, defined as the quadrature mirror filter of the **scaling function** $\phi(t)$, outputs the approximation coefficients, which will be further filtered at next resolution level $j + 1$
2. a **high-pass** filter H_j, defined as the quadrature mirror filter of the **mother Wavelet** $\psi(t)$, outputs the detail coefficients.

Figure 1.2.2 illustrates this iterative process applied to image *Lena*; LL_n, HL_n, LH_n and HH_n refer to the output of a pair of low-pass (L) and high-pass (H) filtering operations. Later, Biorthogonal Wavelets have been proposed to use linear-phase filters whose design is much more simple [CDF92]. Finally, the lifting was introduced to further facilitate the design of mirror filters, and factorize them into simple filters. It was first introduced by W. Sweldens in [Swe96, Swe97a, Swe97b].

The main interest of the Wavelets lies in their scalability features, which model a fundamental characteristic of the human vision: the masking phenomenon [Fie93] which organizes the visual stimuli into a hierarchy. What is more, it is essential to any scalable coder to dispose of such features; for this reason, the popularity of the Wavelets grew strongly over the years. In particular, it strongly inspired the design of JPEG 2000 standard [ISO00b].

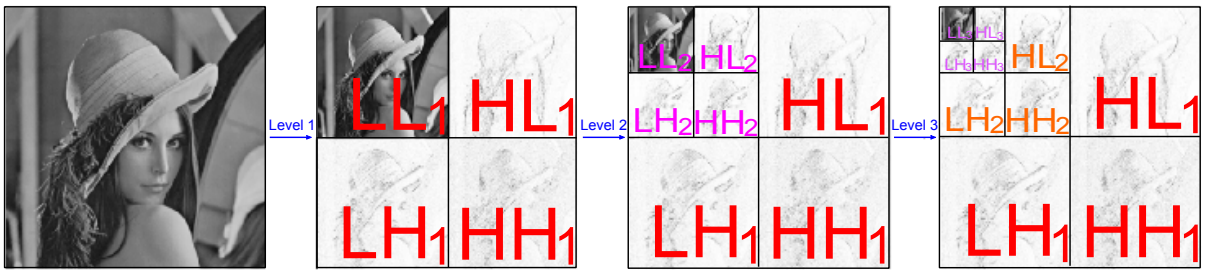


Figure 1.2.2: *Wavelet decomposition of image* Lena *on three levels using a Daubechies 9/7 [Dau88] filter*

Despite all the additional functionalities provided by the Wavelets, they do not provide any kind of directional variety. In two dimensions, the horizontal and vertical atoms only capture horizontal, vertical and diagonal characteristics. Furthermore, the ratio of the support of the mother Wavelet ψ and the scaling function ϕ being fixed, the number of coefficients used to represent an edge will be proportional to its length. In the end, the Wavelet transform, similarly to the Fourier transform and its derivatives, is blind to the orientation and the size of the image contents.

1.2.3 Second generation transforms: towards geometry adaptive transforms

Classic frequency and time-frequency representations have proved to be limited regarding their content adaptation abilities. As a step towards fully-adaptive transforms, directional transforms have been proposed to account for the orientation and the dimension of the image patterns. Both frequency and space-frequency representations inspired a wide variety of adaptive transforms and representations. A detailed review of such transforms can be found in [Rob08, LG08].

Directional frequency transforms have been proposed through modified versions of the DCTs. They will be further reviewed in section 1.2.3.1. As for space-frequency representations, they also inspired numerous directional transforms known as Second Generation (2G) Wavelets. Some of them intrinsically model the geometry of images, including Ridgelets, Curvelets and Contourlets, and use a fixed set of basis functions to model various regularity patterns. They will be investigated in section 1.2.3.2. Alternatively, Bandelets, Directionlets and Wedgelets explicitly model the geometry of images, thus providing an adaptive set of basis functions which consists of a dictionary of geometrical patterns. They will be further reviewed in section 1.2.3.3. Finally, sparse representations provide a more general approach to the adaptive description of a signal according to its contents. They will be investigated in section 1.2.3.4.

1.2.3.1 Directional Discrete Cosine Transforms

While most directional transforms have been built after the Wavelet transform, a few of them have been derived from the DCT. The Directional DCT (DDCT) was first introduced by Fu and Zeng in [ZF06, BZ07a, BZ07b]. They took inspiration from the Shape-Adaptive DCT (SA-DCT), an object-based DCT transform [SM95, KS98, FKE07] used in MPEG-4 Part 2 standard [ISO00a]. Whatever the chosen orientation, the DDCT follows the exact same four steps (see figure 1.2.3):

1. the coefficients of the input block are transformed by a set of one-dimensional DCT along the chosen direction;
2. transformed coefficients are then organized into columns;
3. each line is then further transformed by a one-dimensional DCT;
4. coefficients are aligned from the left, then quantized and scanned by a modified zig-zag scan.

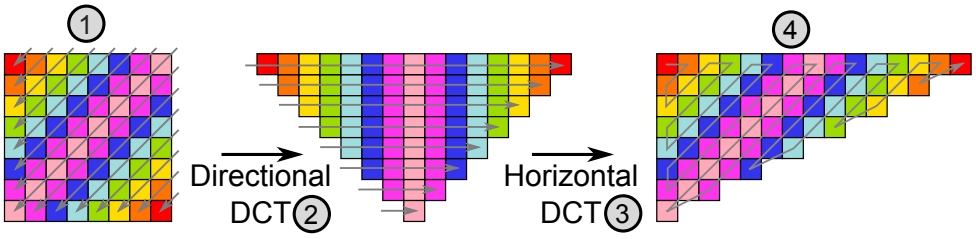


Figure 1.2.3: *DDCT decomposition for a bottom-diagonal direction* ($\theta = -3\pi/4$)

Both transforms suffer from the "mean weighted defect" [KS98]: the DC value is spread over several coefficients, and quantization may have a disastrous effect on the representation. Consequently, it has been proposed to apply these transforms on zero-mean blocks, and to separately send the DC value.

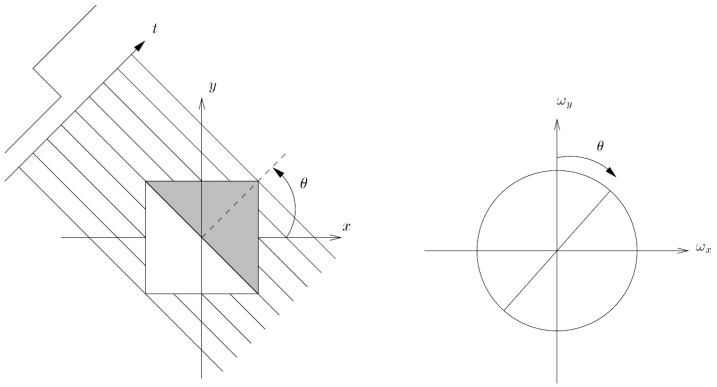
1.2.3.2 Establishing implicit geometry models: Ridgelets, Curvelets and Contourlets

a Detecting the orientation of the singularities: the Radon transform

The Radon transform [Rad17] extracts the singularities from a two-dimensional signal according to their orientation θ. For any two-dimensional function $I(x, y)$, it projects the signal into a direction given by an angle $\theta \in [0, 2\pi[$, so that

$$Rad[I](\theta, t) = \iint_{-\infty}^{+\infty} I(x, y)\delta(x\cos\theta + y\sin\theta - t)\,dx\,dy \tag{1.10}$$

where t runs through all the lines of orientation θ, and $\delta(x)$ is the Dirac delta function (see figure 1.2.4). $Rad[I](\theta, t)$ corresponds to a radial section of the spectrum of I. Assuming that I is crossed by a one-dimensional singularity whose orientation is θ, the Radon projection of $I(x, y)$ in this direction transforms the singularity into a point.



(a) *A Radon section* **(b)** *A radial cut in I's spectrum*

Figure 1.2.4: *A line singularity oriented along direction θ is transformed into a single point in the Radon projection; its Fourier coefficient are given by the corresponding radial section in $I(x, y)$'s spectrum*

Thus, an image transform based on the Radon transform will be able to capture the orientations of the contents in images (see figure 1.2.5). A discrete and exact implementation of the Radon transform has been provided through the Mojette transform [Gué09]. From the Radon transform, *Ridgelets*, *Curvelets* and *Contourlets* were proposed as oriented transforms; they ally the advantages of both Radon and Wavelet transforms.

b The Ridgelets: towards a multi-resolution analysis of directed singularities

The *Ridgelet transform* [Can98, CD99b, CD99a] cascades Radon and Wavelet transforms:

$$Rid[I](a, b, \theta) = \int_{-\infty}^{+\infty} \psi_{a,b}(t) \cdot Rad[I](\theta, t)\,dt = \int_{-\infty}^{+\infty} \psi_{a,b,\theta}(x, y)I(x, y)\,dx\,dy \tag{1.11}$$

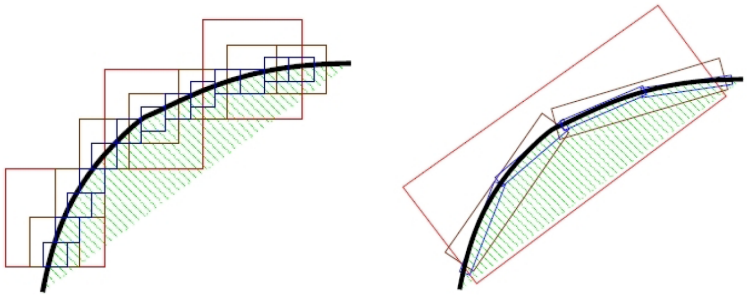
where $\psi_{a,b}(t)$ is a mother Wavelet whose scale and translation factors are respectively a and b -see equation (1.8)-. The new atom for the representation $\psi_{a,b,\theta}(x, y) = a^{-1/2}\psi\big((x\cos\theta + y\sin\theta - b)/a\big)$ is called a *Ridgelet*. Its orientation θ enables the Ridgelets to efficiently represent straight contours. As for curved contours (especially $\mathcal{C}^2$ curvatures), they can be easily represented as a succession of small straight pieces of contours.

Ridgelets, however, are only parametrized from their position along their direction axis θ: they cannot be be precisely localized within the image plane. To address this issue, Donoho [Don00] proposed the construction of Orthonormal Ridgelets as a windowed version of the Ridgelet transform. In practice, however, this family of Ridgelets are discretized with difficulty.

c Curvelets and Contourlets: discrete and practical directional transforms

Later, Candès and Donoho provided the *Curvelets* [CD99b, DV03] as a multi-scale transform which partitions the frequency domain into ring-shaped sub-bands, which implicitly discretizes the direction θ. Each of these sub-bands are split into blocks with a set of smoothing windows whose size depends on the resolution level. Finally, the transform is applied to each of these blocks.

As an alternative, Do and Vetterli made a connection between the Curvelets and directional filter banks [DV01], thus introducing the *Contourlets* [DV05]. In opposition to the Curvelets, Contourlets are intrinsically discrete. The sub-band decomposition is partially performed by a Laplacian pyramid [BA83] which generates, for each resolution level, a low-pass sub-sampled version of the input signal. Finally, the band-pass output is further input to a Directional Filter Bank (DFB) [BS92], which gets rid of the intrinsic redundancy brought by the pyramidal representation. However, their discrete implementations are currently highly redundant, thus fail at efficiently compacting the images, despite the accurate local adaptation to the geometry which is provided.



(a) *Wavelets: non-oriented* **(b)** *Curvelets and Contourlets: oriented*

Figure 1.2.5: *Analysis patterns of Wavelet and Contourlets transforms [DV05]*

d Brushlets: adaptive tiling of the frequency plane

Brushlets were introduced by Meyer *et al.* in [MC97] as an alternative way to account for images contents in an adaptive fashion. Instead of focusing on the spatial contents, Brushlets directly process the frequency plane, which is adaptively partitioned into tiles of variable sizes. Where pure geometrical approaches may fail (*e.g.* locally periodic textures), Brushlets proved to be an valuable alternative. However, they cannot handle piecewise smooth images.

1.2.3.3 Establishing explicit geometry models: Bandelets, Wedgelets and Directionlets

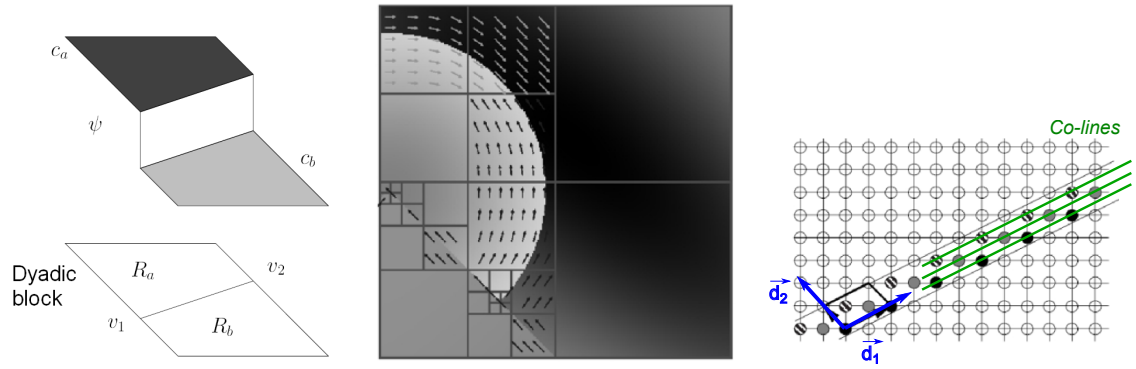
While the transforms designed after the Radon transform use a fixed set of basis functions, some other approaches explicitly describe the images geometry. In particular, *Bandelets*, *Directionlets* and *Wedgelets* provide three different representations of the geometry hold by a block of pixels.

a The Wedgelets: simple discontinuities

Donoho, Romberg and Wakin proposed to model the geometric structure of a block $\mathcal{B}$ from two uniform regions separated by a straight discontinuity: the *Wedgelet* [Don99, RWB02, WRCB02]. The discontinuity splits $\mathcal{B}$ into two regions $\mathcal{R}_a$ and $\mathcal{R}_b$ (see figure 1.2.6a). The Wedgelet is parametrized by:

1. the coordinates of the two points v_1 and v_1 where the discontinuity intersects with $\mathcal{B}$'s boundaries;
2. and the average intensities c_a and c_b of regions $\mathcal{R}_a$ and $\mathcal{R}_b$.

Very efficient on images made of uniform regions (*e.g.* cartoon pictures), they are not a good match for natural images. As a consequence, Wakin *et al.* hybridized the Wedgelets and the classic Wavelet transform in [WRCB02]: both representations are competing to describe the current block.



(a) *Wedgelets: a simple discontinuity* **(b)** *Bandelets: geometric flow* **(c)** *Directionlets:* co-lines*, integer lattice*

Figure 1.2.6: *Three ways to establish a model of the images geometry*

b Modelling the geometric flow: the Bandelets

In images, the geometric flow corresponds to the field of spatial gradients of the pixels intensities. The geometry flow is constant along isophotes (contours of equal intensity), hence along images contours. As a consequence, *bandelets* were introduced by Mallat *et al.* and describe the structure of a block [PM05a, PM05b] through the directions of constant geometry flow. The image is partitioned into a quadtree according to its geometric flow (see figure 1.2.6b). Two versions of these Bandelets have been developed: 1G Bandelets and 2G Bandelets.

The 1G Bandelet transform performs, on each block of pixels, the following operations:

1. it re-samples the intensities of the pixels along the geometric flow;
2. it applies a deformed Wavelet transform to these intensities, folowing the geometric flow ;
3. finally, the *bandeletisation* further compacts the the Wavelet coefficients into Bandelets, according to their correlation along the geometric flow.

Yet, First Generation (1G) Bandelets are not intrinsically discrete, and do not provide any scalability features. 2G Bandelets [Pey05, PM05b, PM05c] were consequently proposed as a workaround. They directly apply the Wavelet transform on the original image. Then, space-frequency coefficients from each sub-band are projected along the direction of the geometrical structures of the considered sub-band. Finally, a one-dimensional Wavelet transform is applied to the projected coefficients, which results in Bandelets coefficients.

From the provided geometry model, Bandelets are more efficient than Curvelets or Contourlets in terms of compression. However, they are very computationally intensive. Also, they suffer from the problem of optimization of bitrate allocation between the geometry description and the Wavelet coefficients.

c A lattice-based geometry model: the Directionlets

Even more recently, Vetterli *et al.* proposed another directional transform: the *Directionlets* [VBLVD05, VBLVD06, VBLVD07, VBLV07]. The Directionlet transform is a separable, integer and multi-directional transform. To this end, Directionlets are defined on an integer lattice Λ. Λ is defined as a linear combination of two vectors $\vec{d_1}$ and $\vec{d_2}$; $\vec{d_1}$ is the direction of the transform and corresponds to the orientation of the regularity which needs to be represented, $\vec{d_2}$ is the alignment direction. Each of these directions are partitioned into *co-lines*, defined as the intersections of the integer lattice $\mathbb{Z}^2$ with Λ's subsets (see figure 1.2.6c).

The image is then decomposed into anisotropic Wavelets; this is known as Anisotropic Wavelet Transform and written AWT(n_1, n_2). The Anisotropic Wavelet Transform (AWT), originally dedicated to horizontal and vertical directions only, performs n_1 transforms along the first direction and n_2 transforms along the second direction. An oblique version of the AWT is applied to each of these co-lines, thus following directions $\vec{d_1}$ and $\vec{d_2}$. This oblique version of the AWT is called the Directionlet transform and written S-AWT$(\boldsymbol{M}_\Lambda, n_1, n_2)$, where $\boldsymbol{M}_\Lambda$ is the generating matrix of Λ.

In practice, the Directionlet transform is performs as follows:

1. it first decomposes the integer lattice $\mathbb{Z}^2$ into sub-lattices Λ;
2. it then rotates the pixels using the lattice generating matrix $\boldsymbol{M}_\Lambda$: the rotated singularity is now oriented horizontally;
3. it finally applies the AWT transform along the horizontal direction (lifting versions of this transform have been proposed [CGM04, DWL04, WZVS06, DWW+07]).

Even though Directionlets outperform the traditional Set Partitioning In Hierarchical Trees (SPIHT) Wavelet coder [KXP97], they do not provide an embedded bitstream, *i.e.* rate scalability, which is now strongly required from modern coding schemes. Furthermore, they are much more computationally demanding than fixed transforms: the computation of the geometry is quite costly.

1.2.3.4 Sparse representations

Besides geometry-based transforms, *sparse representations* can be seen as a generic transform whose basis functions (or atoms) are optimized in regards to the currently processed signal. While previous adaptive representations explicitly exhibit the images geometry, sparse representations use optimal dictionaries of atoms which implicitly account for any kind of image features: they are not limited to the geometry information.

Sparse representations are representations that account for most or all information of a signal with a linear combination of a small number of elementary signals called atoms. They are also called *parsimonious representations*. Often, the atoms are chosen from a so-called over-complete dictionary. Formally, an over-complete dictionary is a collection of atoms whose cardinality exceeds the dimension of the original space which is used to describe the signal. Hence, any signal can be represented by more than one combinations of different atoms. Numerous over-complete dictionaries have been proposed, including stationary Wavelets, Wavelet packets, Cosine packets, Chirplets and Warplets.

Sparse coding refers to techniques that aim at finding a representation of a signal which uses a small number of significant coefficients. Decoding merely consists of the appropriate weighted summation of the relevant atoms. However, finding an over-complete dictionary is a non-trivial problem. In particular, finding a representation with the smallest number of atoms from an arbitrary dictionary has been shown to be a NP-hard problem.

As a consequence, considerable efforts are being put into the development of many sub-optimal schemes. Some algorithms iteratively build up an approximation of the signal one coefficient at a time (Matching Pursuit (MP) [MZ93], Optimized Orthogonal Matching Pursuit (OOMP) [DMA97]) while some others process all the coefficients simultaneously (Basis Pursuit (BP) and Basis Pursuit De-Noising (BPDN) [CDMS98], FOcal Underdetermined System Solver (FOCUSS) family of algorithms [GGR95, Rao97], Global Matched Filters (GMFs) [MF06]).

Sparse representations have been introduced as candidates to represent still and moving images by R. Neff and A. Zakhor in [NZ97]. Among recent works, over-complete dictionaries made after the DCT [MFGT07] and the DDCT [DHGF10] have been proposed. Very encouraging results have been obtained; still, practical solutions are often computationally intensive. Also, basis functions (or atoms) may not be known in advance, which prevents the transform from being thoroughly optimized.

1.3 A focus on the temporal correlation of image sequences: the importance of the motion

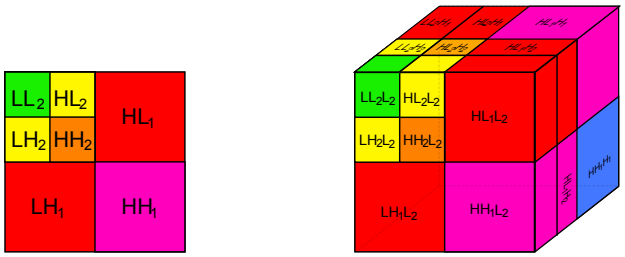
Previous sections introduced various techniques to decorrelate a signal along its temporal (for one-dimensional signals) or spatial (for two-dimensional signals) axis. In image sequences, however, the correlation is spread over both spatial and temporal dimensions. Classical correlation techniques could be blindly applied to decorrelate the sequences along the temporal axis. This is known as blind decorrelation; such approaches will be brought up in section 1.3.1.

However, it is easily observed that successive images are highly correlated as they often correspond to successive states of a scene. The difference between successive images is then mostly characterized by the evolution of the scene: moving object, camera motion, illumination changes, etc. Assuming that this evolution can be modelled, it can drastically improve the temporal decorrelation. Section 1.3.2 will very briefly introduce the concept of motion compensation (chapter 2 will be dedicated to motion compensation). Finally, section 1.3.3 will briefly review a family of transforms which decorrelate spatio-temporal signal along the motion trajectory.

1.3.1 Blind decorrelation

Direct approaches to the problem of decorrelation do not make any hypothesis on the nature of the correlations. They are known as *blind decorrelation* techniques, and are well-suited to signals whose correlations are of heterogeneous nature. Predictive, transform and even self-similar techniques were proposed, among other things, to extract the redundancies of images. The same techniques can be used in a similar way to decorrelate an image sequence along its temporal dimension.

Predictive approaches have been introduced through *conditional replenishment* [Kor67, CH73, Jon77, HJJ79]: they detect and skip the images areas which remain still, and only send the changes in remaining areas. Both predictive and transform techniques have been proposed to represent the changes (*e.g.* predictive interpolation in [Kor67, CH73], WHT intra-coding in [HJJ79], etc). Later, it was also proposed to directly apply the Wavelet transform along the temporal dimension [KV88, PJF95, KXP97] (see figure 1.3.1). Each dimension of the image sequence was thus equally processed; due to the multi-resolution analysis performed by the Wavelets, fully scalable coders were obtained.



(a) *Spatial subbands of a 2D Wavelet decomposition*

(b) *Spatio-temporal subbands of a 3D Wavelet decomposition*

Figure 1.3.1: *2D and 3D Wavelet decompositions of order 2*

However, these approaches did not proved to be efficient in terms of compression, for image sequences which undergo complex changes. In addition, the nature of the temporal correlations is different from the nature of the spatial correlations. Indeed, a sequence of natural images often represents the same scene at different instants; as a consequence, successive images will be highly redundant. Their differences are most often the consequences of textures displacements and deformations, both consequences of object or camera motion. Hence, it was naturally proposed to apply predictive and/or transform techniques along the trajectory of the motion.

1.3.2 A predictive approach to the temporal decorrelation of image sequences: texture alignement and motion compensation

It was soon identified that, for natural image sequences, *motion compensation* is an essential step towards an efficient spatio-temporal decorrelation scheme [Gir87]. *Motion compensation* describes an image in terms of relative deformations with respect to a reference image, and assumes that

$$I_t(x, y) \approx I_{t+1}(x + dx, y + dy) \tag{1.12}$$

where $\overrightarrow{d} = [dx, dy]^T$ is the displacement vector which describes the displacement undergone by the current pixel from instant t to instant $t + 1$. In order to perform an efficient motion compensation, an appropriate motion model and an efficient motion estimation technique are both required.

When the motion is computed over a Group Of Pictures, depending on the motion model, one may be able to deduce the trajectory of any pixel. By performing a global motion compensation on several successive images onto a reference grid, one may then apply classic predictive or transform techniques to remove the redundancies along the motion trajectory. This is known as *texture alignement* [TZ94].

1.3.3 Applying transforms along the motion trajectory

More recently, a large number of pioneering works proposed to incorporate a motion compensation step within the temporal transform [Ohm94, TZ94]. This led to the class of transforms known as Motion Compensated Temporal Filtering (MCTF);

they use short-kernel filter banks and provide a compact representation of image sequences. They also decouple the transform step from the coding step: both of them may then be independently optimized.

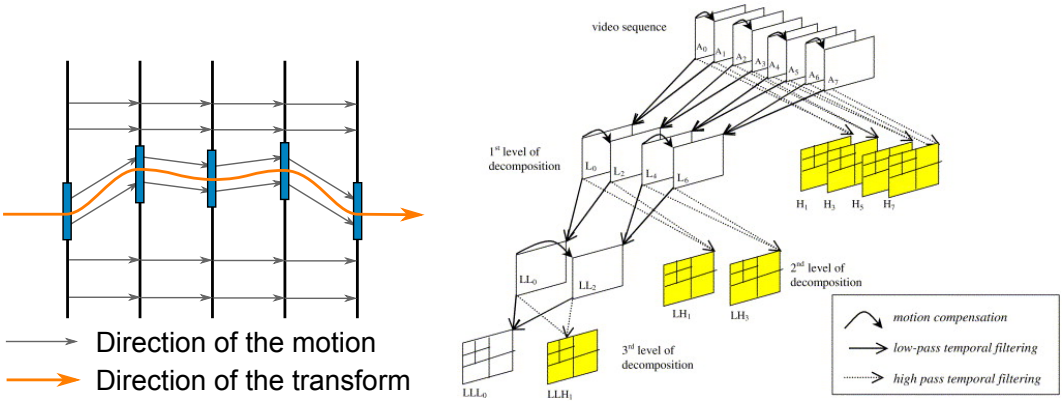
The Wavelet transform and its lifting optimization have been identified as a very handy way to perform a three-dimensional transform on the aligned coefficients output by the MCTF (see figure 1.3.1). The lifting framework provided a way to handle arbitrary sub-pixel accuracy, while guaranteeing perfect reconstruction [PPB01, ST01]. Finally, the motion is decorrelated from the texture, which enables the Wavelet transform to provide a fully scalable representation: spatial and temporal scalability.

The MCTF can be performed either in the spatial domain, or in the transform domain:

$t+2D$ **lifting** refers to early techniques that apply the MCTF prior to the three-dimensional DWT [Ohm94, TZ94, CW99, CW04, CP03, LG08]. In such approaches, also known as Spatial Domain MCTF (SDMCTF),the MCTF is performed in the spatial domain.

$2D+t$ **lifting** refers to late approaches that apply the MCTF after the spatial DWT decomposition [AMB+04]. In such approaches, also known as Inband MCTF (IBMCTF), the MCTF is performed in the time-frequency domain. IBMCTF improves the performances with spatial scalability by using a leaky motion compensation. In this particular case, the motion compensation is performed using either low or high quality reference images, as a way to introduce a trade-off between the precision of the motion and the coding efficiency.

Later, *motion thread* [XXLZ01], then *Barbell lifting* [XWX+04] were proposed as a way to deal with non-connected and multi-connected pixel which are often problematic for the representation of the motion (see 3.4.1.2). Finally, the Unconstrained MCTF (UMCTF) modifies the lifting process and deactivates its update step [TvdS02, TvdSA+05] in order to remove the artefacts of the Wavelet representation at low bit-rates (*e.g.* ringing effects).



(a) *Temporal filtering along the motion trajectory (section view)*

(b) *MCTF [TvdSA+05]*

Figure 1.3.2: *How the MCTF filters the image sequence along the trajectory of its motion*

Even though MCTF's main advantage surely is its ability to provide a fully-scalable representation, it provides an open-loop prediction structure whose use is much more convenient that closed-loop structures. Also, it provides a continuous representation of the information across the temporal axis. However, MCTF cannot efficiently represent areas where motion estimation has failed.

1.4 Conclusion

This chapter reviewed many approaches which can be used to decorrelate a signal. They were categorized into three main families: predictive, synthetic and transform techniques. Predictive and transform techniques are especially popular in image and video compression: most recent compression schemes now rely on the combination of several predictive and transform techniques. In particular, frequency and time-frequency representations proved to be very efficient as ways to describe the image singularities. Last developments notably led to a large variety of image-adaptive transforms which implicitly or explicitly exhibit the images geometry. As for synthetic decorrelation techniques, they are most often dedicated to specific signals: their use is generally restricted to corresponding applications.

In image sequences, however, classical decorrelation techniques fail at capturing the correlation along the temporal axis. Indeed, the changes that occur between successive instants of a sequence of natural images are the consequences of the evolution of the scene across time (moving objects, illumination changes, etc) and the evolution of the camera parameters. Motion compensation was then naturally introduced as a way to describe the current image as a deformed version of one or more reference images. This allows the textures to be aligned; temporal correlations are then exhibited and easily captured from additional decorrelation techniques.

From their use, decorrelation techniques can alternatively be classified into two categories: local and global approaches. On one hand, local approaches (predictive techniques, image-adaptive transforms) provide an accurate information about the characteristics a signal locally holds. These characteristics may even be processed at different scales, provided that the analysis method shows scalability features. On the other hand, global approaches (non-adaptive transforms, synthetic techniques) process the signal in its entirety and provide valuable information regarding the overall correlations it holds.

Amongst the ever increasing number of image sequences processing schemes, both global and local approaches have been employed:

- in the **spatial domain**, for instance, individual images can be analysed either locally (*e.g.* block-based transforms) or globally (this generally implies the use of the Wavelet transform). Yet, **local approaches** are generally favoured as they are a better match to the non-stationary nature of the spatial information;
- as for the **temporal domain**, all modern image sequences processing schemes now employ motion compensation. From then, the temporal decorrelation is generally performed on a frame-by-frame basis, *i.e.* through a local approach.

In conclusion, one may remember that **individual images** should be **locally decorrelated** as they hold a non-stationary information. On the contrary, the **temporal information**, despite the displacements and the deformations undergone by the textures, is highly persistent across time: a **global decorrelation** approach is very suitable. Most compression schemes (they will be further reviewed in chapter 3):

- either they perform a **local spatial** analysis and a **local temporal** analysis (the classical approach);
- or, they perform a **global spatial** analysis and a **global temporal** analysis (MCTF-based compression schemes).

Instead, we propose to combine the advantages from both local and global representations. In our opinion, an idealistic representation should account for the local geometrical patterns hold by individual images, while processing the temporal axis as continuously as possible. However, as continuous the information may be along the temporal axis, it is essential for the representation to be able to detect when and where discontinuities occur, in order to employ more suited decorrelation techniques. This thesis focuses on this problem, and will most exclusively focus on ways to detect, assess, and represent the temporal persistence of the information along the temporal axis. As the motion prediction plays an essential role into the proposed representation, next chapter will be entirely dedicated to motion compensation.

Chapter 2

Motion compensation: models and estimation techniques

As a predictive temporal decorrelation technique, the motion compensation has been identified, in previous chapter, as a crucial step in a eye to video compression. Indeed, it is particularly efficient when it comes to a sequence of images describing the successive contents of the same scene at several time instants. Motion compensation describes the current image in terms of deformations with respect to a reference image. Motion compensation aims at minimizing the energy hold by the Displaced Frame Difference (DFD) (the difference between the current image and its motion compensated prediction). In order to perform motion compensation, one has to define what exactly the motion is regarding image sequences, and how it can be computed. Indeed, the paradigm of motion compensation lies on two concepts: the motion representation and the motion estimation.

But, first and foremost, what is motion exactly ?

- From a **perceptual** perspective, neurophysiologists, psychophysicists and physicians showed that, in animals [HW65] and humans [MA88], specialized structures are in charge of the motion perception. These structures put in place two processes, according to the importance of the information. In eye-attracting areas, a *focalised* vision perceives an acute representation of the motion. Elsewhere, a *peripheral vision* settles for a rough detection of the motion.
- From an **engineering** perspective, the motion of a scene can then be seen as the combination of two components: a global motion (camera or background displacements), and a set of specific local motions (moving objects).
- From a **signal processing** perspective, finally, the images colour intensities are the only reminiscences of the real 3D motion. Assuming that the colour of a moving object invariant across time, the motion can be estimated from the variations of the colour information across time. Thus, most approaches aim at minimizing an energy functional which measures the distance between the original current image and its motion-compensated prediction.

Typically, the temporal variations of an image sequence result from several processes, including:

- the **projection** of the **real 3D motion** into the image plane;
- the **illumination** changes and corresponding **transparency** and **shadowing** effects;
- the **occlusions** which hide some parts of the scene;
- the **camera parameters** changes;
- and the **noise**.

In practice, an exhaustive model of all the aforementioned phenomena would result in a huge amount of complexity: the model needs to be simplified. Various motion models have been designed and conceptualize the nature of motion in different ways. In any case, they all define how the real 3D motion of a scene is linked to the temporal variations of resulting image sequence. Finally, an appropriate motion estimation technique is used to evaluate the motion parameters.

In the last thirty years or so, the interest in motion representation and estimation has been widely growing, and many studies have been focusing on these matters. According to the targeted applications, a wide range of motion models have been provided to fit the corresponding requirements. Nowadays, numerous models and corresponding estimation techniques

are available. This chapter will review key contributions to motion compensation, including motion representations and motion estimation techniques.

Section 2.1 will explain how the motion is captured throughout the shooting system, and introduce the optical flow as a projection of the motion field in the image plane. Section 2.2 will then review classic motion representations which have been provided throughout the years. Then, section 2.3 will provide an insight into estimation algorithms.Section 2.4 will focus to models widely used in video compression: blocks and meshes. Finally, section 2.5 will conclude the chapter.

2.1 From the real three-dimensional motion to the optical flow

Most often, the contents of an image sequence is a real life scene which has been captured by a shooting system. Thus, two-dimensional images from such a sequence are the successive projections of the three-dimensional real-life scene onto the camera plane (or image plane). Obviously, the motion information will not have the same nature whether we consider the real-life scene or the camera plane. This section will explain how exactly the motion is captured throughout the shooting system. It will also mention several issues which alter the motion information and interfere with its evaluation in the camera plane.

2.1.1 Motion throughout the shooting system

The *real* 3D motion is the motion that animates the real scene, in the 3D real space. It can only be captured by an optical system such as the eyes or a camera. Once the scene has been seen or shot, the *real* 3D motion is not available anymore; one can only perceive its 2-dimensional projection onto the retina or the camera plane. This 2-dimensional projection is know as the *apparent* motion. Motion estimation techniques mostly aim at measuring the *apparent* motion; yet, some of them, including the homographies, may aim at measuring the *real* 3D motion.

2.1.1.1 Real and apparent motion

When considering a shooting system, the *apparent* motion is considered to be either the **orthographic projection** (also called parallel projection), or the **perspective projection** (considering the pinhole camera model) of the real 3D motion onto the image plane. Figure 2.1.1 illustrates the perspective projection of a point belonging to an object in motion. Let P_t be a point of the real 3D scene which belongs to an object $\mathcal{O}$ at time instant t. Let P_{t+1} be its collocated point at instant $t+1$. Finally, let P'_t and P'_{t+1} be their projections on the image plane:

- in the 3D scene, $\overrightarrow{P_t P_{t+1}}$ is the **real** motion vector;
- on the image plane, $\overrightarrow{P'_t P'_{t+1}}$ is the **apparent** motion vector.

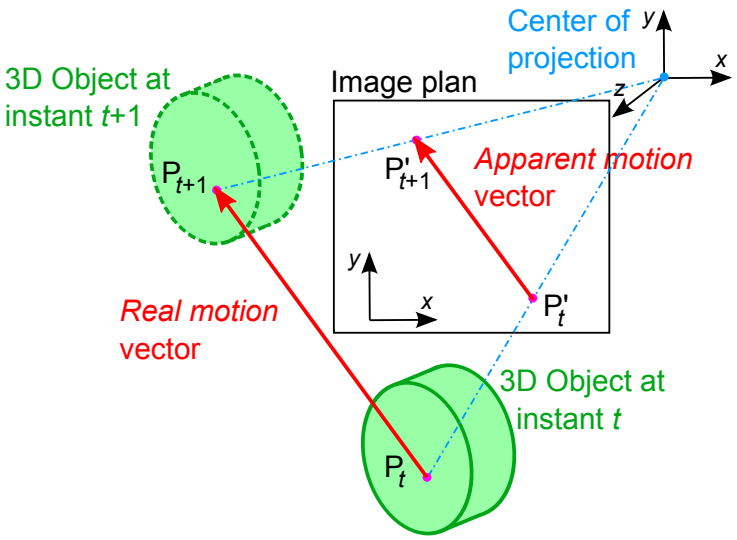


Figure 2.1.1: Real *and* apparent *motions in an optical shooting system [Gra03]*

2.1.1.2 Occlusions and disocclusions

Assuming that the shooting system provides a single angle of view, parts of the scene may be hidden or unhidden to the observer: they are respectively known as occlusion and disocclusion phenomena. In that event, the projected motion does not include anymore the motion information of hidden areas. This problem is shown in figure 2.1.2. While most approaches do not handle occlusions, some region-based and boundary-based approaches [Tzi92, Kim94, Pet99] track moving objects and their boundaries across time.

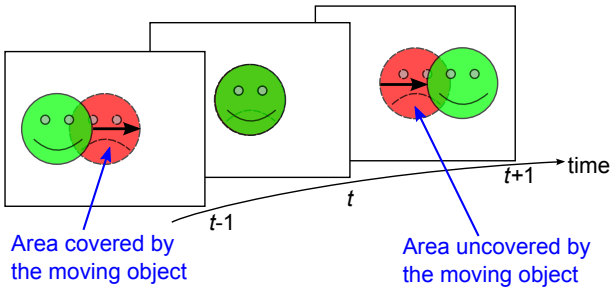


Figure 2.1.2: *Occlusion phenomenon: object ☹ is temporarily hidden behind object ☺.*

2.1.1.3 Global and local motion

The *apparent* motion results from two processes: the real 3D motion of the scene itself, and the *parallax component*. The parallax component corresponds to the evolution of the shooting system parameters (camera motion, zooming operations, etc): a motionless scene captured by a camera in motion will still show an apparent motion. Some motion compensation techniques exhibit both scene and parallax motion components.

In [SA96], the global motion (referred to as *dominant motion*) is coupled with a set of local motions (referred to as *multiple motions*). In [HFB04], both local and global motion components are concurrently estimated. In general, however, most techniques do not differentiate the local motion from the global motion. Whichever, being able to separate global and local motions provide invaluable information regarding the images contents, especially helping with the identification of the different elements of the scene.

2.1.2 Motion field, displacement vector

Once the shooting process has been performed, the image sequence is hold onto the camera plane (or image plane). The *motion field* is an ideal representation of the *real* motion as it is projected onto the image plane. It is formally defined as the time derivative of the positions of the original 3D real points projections onto the image plane. In the end, the *motion field* function maps the image coordinates into a 2D *velocity*.

Let us consider a point P from the image plane. Its instantaneous *velocities* are given by the successive time derivatives of its trajectory across time within the image plane. Hence, P's *velocity* is tangent to its temporal trajectory. In the context of motion compensation, though, the *displacement* vector $\overrightarrow{d} = [dx, dy]^T$ is often preferred to the *velocity* $\overrightarrow{v} = [vx, vy]^T$ to describe the compensation processus. The *displacement vector* expresses the displacement undergone by a point of the image plane between two time instants. Figure 2.1.3 illustrates the difference between *velocity* and *displacement vector*.

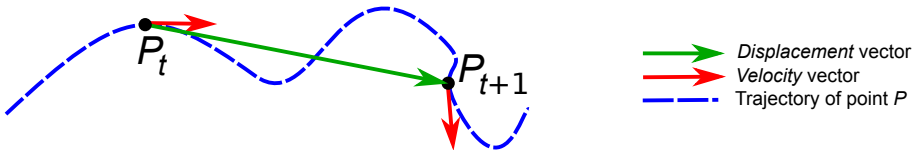


Figure 2.1.3: Velocity *and* displacement vector*: two different notions*

In practice, however, both vectors are often confused as they are related by the temporal sampling period. Assuming that the sampling frequency approaches infinity (hence the sampling period approaches 0), *velocity* and *displacement vectors* are linked:

$$\overrightarrow{v} = \lim_{\tau_s \to 0} \frac{\overrightarrow{d}}{\tau_s} \tag{2.1}$$

2.1.3 Optical flow

2.1.3.1 From the motion field to the optical flow

Due to the *apparent* motion, objects, surfaces and edges may be displaced across time in a sequence of images. In practice, one can only measure the variations of the intensities of the pixels. The displacement of the pixel intensities observed by the *motion field* is known as the *optical flow*. In other words, the *optical flow* is the projection of the motion field into the space of pixel intensities. Provided that objects, surfaces, and edges hold the same colour information throughout time, the *apparent* motion field is more or less given by the displacement of the pixel intensities. This is known as the *luminance invariance hypothesis* (in practice, both luminance and chrominance colour components may be considered), and laids the foundation stone of the motion compensation paradigm. This can be expressed as

$$I_{t+\tau_s}(x + vx, y + vy) = I_t(x, y) \tag{2.2}$$

where $\overrightarrow{v} = [vx, vy]^T$ is the velocity of the *optical flow* at position (x, y). From now on, the notation $\mathcal{V}$ will designate both the *optical flow* and the *motion field*. However, mapping the motion field into the optical flow is a not bijective operation: one cannot retrieve the exact *apparent* motion field from the optical flow. For instance, a uniformly coloured sphere rotating around its centre will not result in any pixel intensity displacement. As a consequence, resulting optical flow is null, even though the apparent motion is not. This phenomenon is known as the *aperture problem*.

2.1.3.2 The aperture problem

In 1911, psychologist Pleikart Stumpf made the observation that each neuron dedicated to motion perception was only sensitive to a small part of the visual field. Later on, it was shown that the human brain perceives the motion with two kinds of structures. At first, *Reichardt detectors* perform a first-order analysis of the motion. Then, the visual cortex performs a second-order analysis to disambiguate true global motion detection. Pleikart Stumpf's work anticipated the existence of the Reichardt detectors, and provided an early description of what is now known as the *aperture problem* [Tod96]. Indeed, Stumpf's observation can be formulated as follows: each neuron is looking at the visual field through a small window, or *aperture* [Hil84]. First-order motion detectors are then only sensitive to a single component of the apparent motion, whose orientation is perpendicular to local contours in motion. This is known as the *aperture problem*.

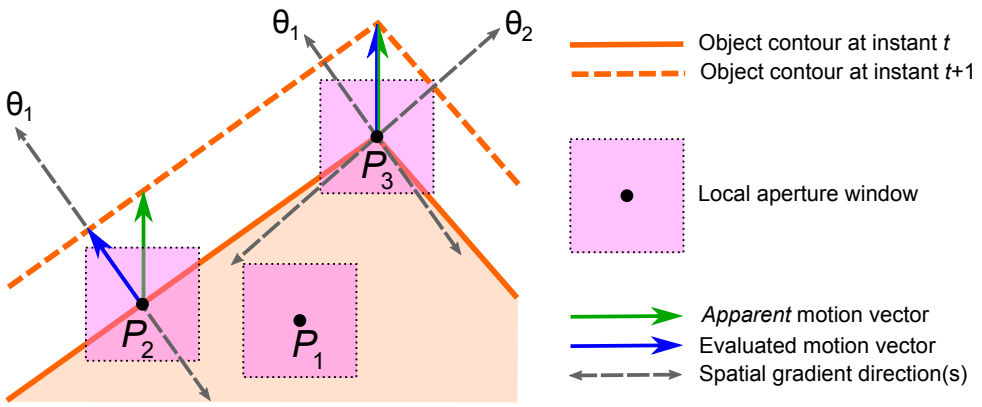


Figure 2.1.4: *The aperture problem*

The estimation of the motion field through the optical flow suffers from the same problem. Indeed, first-order motion estimators are only able to detect the amplitude of the apparent motion along the direction of the images spatial gradient $\overrightarrow{\nabla I}(x, y)$ [HS81, BL02]. In other words, the system to be solved is under-determined. Figure 2.1.4 shows the corner of a uniform object in vertical motion:

1. in P_1, the spatial gradient is **locally null** as the aperture window holds a uniform information: $\overrightarrow{\nabla I}(P_1) = \overrightarrow{0}$. The *apparent* motion is **not observable**;
2. in P_2, a single gradient is observable along a specific direction θ_1: only θ_1's component of the apparent motion is measurable, and $\left\langle \overrightarrow{\nabla I}(P_2), \overrightarrow{\theta_1} \right\rangle = 0$;
3. in P_3, multiple local gradients $\overrightarrow{\nabla I}(P_3)$ can be observed along at least two orientations θ_1 and θ_2: the apparent motion can be fully evaluated.

2.1.4 Backward and forward motion compensation

Equation (2.2) showed that successive images from an image sequence are linked by the optical flow. Let I_{ref} and I_{cur} be two images respectively captured at reference instant t_{ref} and current instant t_{cur}. Let us apply equation (2.2) to time instants (resp.) t_{ref} and t_{cur} and corresponding images (resp.) I_{ref} and I_{cur}. The compensation can then be performed in two different fashions

$$\begin{cases} \widehat{I_{cur}}(x,y) &= I_{ref}(x+dx, y+dy) & \textbf{backward} \text{ motion compensation} \\ \widehat{I_{cur}}(x-dx, y-dy) &= I_{ref}(x,y) & \textbf{forward} \text{ motion compensation} \end{cases} \tag{2.3}$$

In the first case, backward motion compensation predicts any position of I_{cur} by shifting its collocated position in I_{ref}. To ensure a complete reconstruction of I_{cur}, one simply requires an injective motion field. In the second case, forward motion compensation projects any position of I_{ref} in the domain $\mathbf{\Omega}_{cur}$ of the current image I_{cur}. As the optical flow may not be surjective, this last approach cannot guarantee a complete reconstruction of I_{cur}. Furthermore, several positions of I_{ref} may be projected onto the exact same position of I_{cur}, which further complicates the compensation process. Consequently, *backward* motion compensation is generally preferred to *forward* motion compensation. In particular, standardized video coders such as MPEG-x [ISO93, ISO94b, ISO00a] and H.26x [ITU90, ITU94, ITU95, Ric03] use a *backward* approach.

Often, yet, the reference instant t_{ref} is prior to the current instant t_{cur}. This misled the original definitions: time-backwards and time-forwards designations sometimes inappropriately overtake the original definitions:

1. **backward** motion compensation corresponds to situations where $t_{ref} < t_{cur}$;
2. **forward** motion compensation corresponds to situations where $t_{ref} > t_{cur}$.

May *backward* or *forward* compensation fashions be mentioned in this document, they will refer to their original definitions.

2.1.5 Frame by frame versus trajectory representations

Local motions, on one hand, result from the displacement of an object moving across the scene. Global motion, on the other hand, results from a change in the camera parameters. In both cases, either the object or the cameras focus are following a given 3D trajectory in the real 3D space. From the projection of the real 3D space into the 2D image plane, original 3D trajectories are projected into 2D trajectories. As a consequence, the motion can be intuitively conceptualized as a set of trajectories across space and time.

As seen in section 1.3.3 from chapter 1, MCTF, for instance, relies on this concept as it transforms the spatio-temporal information along the motion trajectories (the motion information does not necessarily needs to be locally described: MCTF simply requires the ability to extract a set of complying trajectories). Provided that the current image can be projected onto a reference grid, known as the textural domain, any representation modelling the motion as a trajectory will be able to continuously process the temporal correlations. However, it proved to be difficult to rely on such a representation, despite all the efforts which have been made [XWX+04, Cam04, LG08] in this direction, including in particular the Barbell lifting [XXLZ01].

Most approaches, though, avoid this problem by representing the motion as a set of successive displacements. As a consequence, they define a reference grid for each image, which guarantees the prediction of the current image to be complete. Yet, the lack of explicit information regarding the successive positions of a specific pixel across time makes impossible to guarantee any kind of temporal coherence. Resulting representations may then be affected by temporal discrepancies, which reduces the overall quality of the temporal decorrelation.

2.2 Modelling the motion

Previous section investigated the nature of the motion information hold by an image sequence and its relationship with the original 3D real motion. In particular, the apparent motion was proved to be somehow measurable through the optical flow. Whichever way, it is first required for the motion information to be parametrized through an appropriate model. The motion information is generally processed in terms of displacements rather than velocities. From now, may this chapter mention the motion field, this will refer to the displacement field rather than to the original motion field definition.

In order to describe the motion field of an image, the most straightforward solution probably consists of providing a displacement vector for each observable position, *i.e.* each pixel. This representation is known as *dense motion field* and has been extensively used in video analysis applications.

2.2.1 From dense motion fields towards practical representations of the motion

Dense motion fields define a displacement vector (more simply, a motion vector) for each position of the image plane. Typically, this ends up in a large amount of motion information. As a straightforward solution, it was proposed in [HS81] to subsample the motion field $\mathcal{V}$, down to an accurate enough resolution. Typically, images are subsampled into a grid of pixels or sub-pixels: a motion vector is then associated to each of these pixels and/or sub-pixels. The resulting motion field is typically highly redundant, and can be decorrelated from classical predictive or transform techniques. Nevertheless, compression approaches based on dense motion field generally failed at compacting the dense motion information down to a reasonable bit-rate [Sti94, MKW97, DH98, SP98, LG99, HP01].

For this reason, numerous simplified motion models have been proposed. Not only they describe the motion field from a reduced number of parameters, but they may also implicitly include some additional coherence or regularity features, addressing some of the problems that have been raised in previous sections (*e.g.* occlusions and aperture issues) Most of the time, the motion is represented through a parametric, a geometrical, a frequency or a model-based approach.

1. **Parametric models** are inspired from the way the shooting process projects the 3D scene onto the image plane. Going by a specific camera model, they analyse and model the motion information through parametric polynomials. These functions, also known as parametric transforms, correspond to the projection of 3D geometrical transformations onto the image plane. Parametric models will be reviewed in section 2.2.2.
2. **Geometrical models** locally analyse the images deformations. Various patterns have been proposed to partition the image plane into locally deforming areas, including blocks, meshes and regions. Block and meshes will be respectively reviewed in sections 2.2.3.1 and 2.2.3.2.

   Region-shaped patterns are fit to the contours of the sequence elements. In other words, they directly describe the displacements and the deformations of the objects in the scene [Nic92, Gam92, San95, RM04]. Region-based models require the images to be spatio-temporally segmented.

   As both segmentations and deformation representations and estimations are difficult tasks, these models are seldom used in video compression [SGPK94, SM99, AFBD06]. Besides, deformation model of each region is itself generally handled through simpler models (*e.g.* parametric). For both these reasons, region-based models will not be further reviewed in this chapter.
3. **Frequency and time-frequency models** describe the motion field through classical Fourier Transforms [Bru01] or Wavelet Transforms [WKcLC00, Bru01]. Such models will not be reviewed in this chapter as they are seldom used in practice, and will not be of any help regarding our work.
4. **Model-based representations** are employed when the motion is known to behave in a specific manner. In such case, a model of the behaviour is parametrized and matched with the actual motion information [NH87, HW88, HBH+89, Car89, Hee90, BAHH92]. As frequency and time-frequency models, model-based representations will not be reviewed in this chapter.

Both frequency and model-based representations are seldom used in practice, and will not be subject to further review. On the other hand, parametric and geometric models are commonly used in video compression and will be further detailed.

2.2.2 Parametric models

2.2.2.1 From a 3D motion model to its projection into a 2D motion model

The real 3D scene is included within the 3D Cartesian space $\mathbb{R}^3$. As a consequence, the deformation and the displacements of a rigid object $\mathcal{O}$ can be parametrized from the evolution of the 3D coordinates of its points. Let $\mathcal{G}$ be the gravity centre of $\mathcal{O}$, and P an arbitrary point belonging to $\mathcal{O}$'s support $\mathbf{\Omega}_{\mathcal{O}}$. From time instant t_{ref} to t_{cur}, P's displacement vector $\overrightarrow{d}(P) = \overrightarrow{P_{\text{ref}}\,P_{\text{cur}}}$ can be interpreted as the result of two distinct processes:

1. $\mathcal{O}$'s **translation** $\overrightarrow{T} = \overrightarrow{\mathcal{G}_{\text{cur}}\,\mathcal{G}_{\text{ref}}} = [t_x, t_y, t_z]^T$ with respect to its gravity centre;
2. $\mathcal{O}$'s **rotation** $\boldsymbol{R} = \boldsymbol{R_z} \cdot \boldsymbol{R_y} \cdot \boldsymbol{R_x}$ around its gravity centre, which affects the relative position of P with respect to $\mathcal{G}$.

In the end, the coordinates of P at time instant t_{ref} are given by

$$P_{\text{cur}} = \mathcal{G}_{\text{ref}} + \overrightarrow{T} + \boldsymbol{R} \cdot \overrightarrow{\mathcal{G}_{\text{ref}}\,P_{\text{ref}}} \tag{2.4}$$

where $\boldsymbol{R} = \boldsymbol{R_z} \cdot \boldsymbol{R_y} \cdot \boldsymbol{R_x}$ is given by the product of three matrices expressing individual rotations around axis z, y and x:

$$\boldsymbol{R_z} = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{R_y} = \begin{bmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{bmatrix}, \quad \boldsymbol{R_x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} \tag{2.5}$$

Section 2.1.1 showed that the *apparent motion* can be interpreted as the projection of the real 3D motion onto the camera plane $\mathbb{R}^2$. Similarly, the motion parametrization can be obtained from the projection of the 3D motion parametrization provided in equation (2.4) onto the image plane. To this end, several hypothesis need to be done:

- from the pinhole camera model, a perspective projection is used to map the motion parameters from the Euclidian space $\mathbb{R}^3$ onto the image plane $\mathbb{R}^2$;
- the shooting is supposed to be performed along axis z;
- $\mathcal{O}$'s depth is supposed to be constant ($\forall P \in \boldsymbol{\Omega}_{\mathcal{O}}$, $z_P = z_{\mathcal{O}}$, constant).

Through various simplifications of the projection of the 3D parametrization onto the image plane, a large number of parametric motion models have been provided. In such case, the motion is analysed through polynomials describing various geometrical transforms (*e.g.* translations, rotations, stretches, ...). Available models can be categorized into two families:

1. **3D parametric motion models** project the apparent motion from the image plane $\mathbb{R}^2$ back into the original space $\mathbb{R}^3$, through homogeneous coordinates. They are extensively used in computer vision to track the deformation of 3D objects [SFZ00, SB02, PXC02, Pre06] or in vision-based control (visual servoing) [VM05, MV07].
2. **2D parametric motion models** [BL02] stick to the image plane and rely on 2D projections of 3D transformations onto the image plane, including linear, non-linear, and homographic models (further reviewed in forthcoming sections).

2.2.2.2 Rigid objects in motion: linear models

Typically, the rotation matrix $\boldsymbol{R}$ is simplified in two steps:

1. rotation angles (resp.) θ_x, θ_y and θ_z around axis (resp.) x, y and z are small: sine and cosine functions can be approximated by the first two terms of their Taylor expansions; resulting quadratics terms are then be neglected;
2. only rotations around optical axis z are taken into account: those around axis x and y are consequently neglected.

$$\boldsymbol{R} \approx \underbrace{\begin{bmatrix} 1 & -\theta_z & -\theta_y \\ \theta_z & 1 & -\theta_x \\ \theta_y & \theta_x & 1 \end{bmatrix}}_{\text{Simplification 1}} \approx \underbrace{\begin{bmatrix} 1 & -\theta_z & 0 \\ \theta_z & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Simplification 2}} \tag{2.6}$$

a The affine motion model

A generic 2D *affine motion model* $\Theta(t'_x, t'_y, k_1, k_2, \theta_1, \theta_2)^T$ can then be interpreted as the projection of a 3D motion parametrization (see equation (2.4)) onto the image plane[BL02]. It provides a good trade-off between complexity and representativeness, as its six parameters can handle translations, rotations, scaling and linear deformations (see figure 2.2.1b). It is given by

$$P'_{\text{cur}} = \mathcal{G}'_{\text{ref}} + \underbrace{\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}}_{\overrightarrow{T'}} + \underbrace{\begin{bmatrix} k_1 & \theta_1 \\ \theta_2 & k_2 \end{bmatrix}}_{\boldsymbol{R'}} \cdot \overrightarrow{\mathcal{G}'_{\text{ref}} P'_{\text{ref}}} \tag{2.7}$$

where P'_{ref}, P'_{cur}, $\mathcal{G}'_{\text{ref}}$ and $\mathcal{G}'_{\text{cur}}$ are the respective projections of P_{ref}, P_{cur}, G_{ref} and G_{cur} from the real space $\mathbb{R}^3$ onto the image plane $\mathbb{R}^2$. Similarly, $\overrightarrow{T'}$ and $\boldsymbol{R'}$ are the respective projections of , $\overrightarrow{T}$ and $\boldsymbol{R}$.

b Alternative linear models

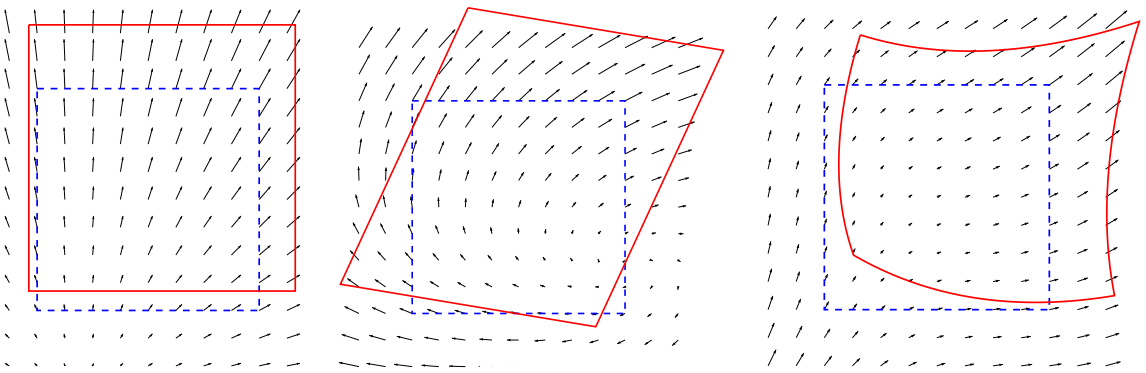
By restraining the the affine model, various simplified models can be derived. Some of them, along with their parameters and abilities, are listed in table 2.2.1.

Models	Model parameters			Displacement and deformation abilities			
	#	Translation $\overrightarrow{T'}$	Rotation $\boldsymbol{R'}$	Translations	Rotations	Scaling	Deformations
Translational	2	$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	✓	×	×	×
Rotational	3	$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}$	$\begin{bmatrix} 0 & \theta \\ -\theta & 0 \end{bmatrix}$	✓	✓	×	×
Divergence	3	$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}$	$\begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$	✓	×	✓	×
Shearing	3	$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}$	$\begin{bmatrix} 0 & \theta \\ 0 & 0 \end{bmatrix}$ or $\begin{bmatrix} 0 & 0 \\ \theta & 0 \end{bmatrix}$	✓	×	×	∼
Simplified affine	4	$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}$	$\begin{bmatrix} k & \theta \\ \theta & k \end{bmatrix}$	✓	×	✓	∼
Affine	6	$\begin{bmatrix} t'_x \\ t'_y \end{bmatrix}$	$\begin{bmatrix} k_1 & \theta_1 \\ \theta_2 & k_2 \end{bmatrix}$	✓	✓	✓	✓

Table 2.2.1: *Various linear parametric motion models: parameter set and motion abilities*

2.2.2.3 Moving and deforming non-rigid objects: non-linear models

Previous section showed that linear parametric models are able to model a wide range of deformations. However, it was assumed for the deforming object $\mathcal{O}$ to be rigid: its shape was constant across time. This is not always the case, and non-linear parametric models [OB94, OB95] have been introduced to model the deformations of non-rigid objects. Figure 2.2.1c) illustrates a typical case of non-rigid deformation.



(a) *Divergence model* **(b)** *Affine model* **(c)** *Quadratic model*

Figure 2.2.1: *Several deformations handled by various parametric models*

A N order non-linear model is obtained from the N first terms of the Taylor expansion of the displacement vector in the image plane. For instance, quadratic models are obtained for $N = 2$, and given by

$$P'_{\text{cur}} = \mathcal{G}_{\text{ref}} + \begin{bmatrix} t'_x \\ t'_y \end{bmatrix} + \begin{bmatrix} k_1 & \theta_1 \\ \theta_2 & k_2 \end{bmatrix} \cdot \overrightarrow{\mathcal{G}'_{\text{ref}} P'_{\text{ref}}} + \begin{bmatrix} k_3 & \theta_3 \\ \theta_4 & k_4 \end{bmatrix} \cdot \overrightarrow{\mathcal{G}'_{\text{ref}} P'_{\text{ref}}} \circ \overrightarrow{\mathcal{G}'_{\text{ref}} P'_{\text{ref}}} + \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix} \cdot \overrightarrow{\mathcal{G}'_{\text{ref}} P'_{\text{ref}}} : \overrightarrow{\mathcal{G}'_{\text{ref}} P'_{\text{ref}}} \tag{2.8}$$

where $\overrightarrow{a} \circ \overrightarrow{b}$ is the matrix Hadamart product of $\overrightarrow{a}$ by $\overrightarrow{b}$, and $\overrightarrow{a} : \overrightarrow{b}$ the matrix Frobenius inner product of $\overrightarrow{a}$ by $\mathbf{b}$. Again, various models can be derived from equation (2.8); in particular, cancelling parameters k_3, k_4, θ_3 and θ_4 result in a model whose individual deformations along each axis x and y are both linear. This is known as a **bilinear** deformation, which might be the most popular quadratic model due to its interesting geometric and algebraic properties, and section 2.2.3.2 will explain how it is used in mesh-based motion representations. Generally, however, quadratics terms are very small ahead of linear terms: non-linear models are seldom used.

2.2.2.4 The homographic model

Homography is a 2D projective transformation implicitly relying on the camera parameters and corresponding perspective projection of the scene [FL88, HZ00]. They are typically used to address changes of camera parameters across time.

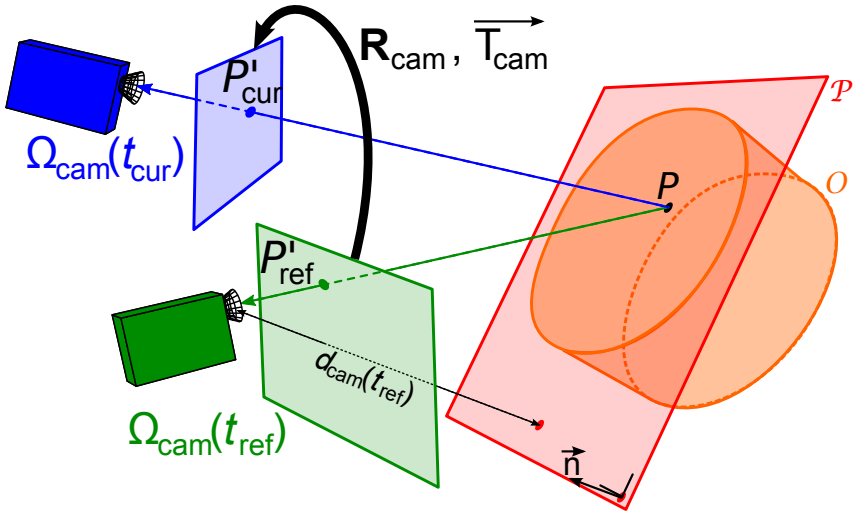


Figure 2.2.2: *Homography of a plane structure shot from different view angles*

- Let P be a point belonging to a fixed and flat 3D object $\mathcal{O}$. Let $\mathcal{P}$ be the plane which holds $\mathcal{O}$'s surface.
- Let P'_{ref} be P's projection at time instant t_{ref} in the current camera reference plane $\mathbf{\Omega}_{\text{cam}}(t_{\text{ref}})$.
- Similarly, let P'_{cur} be P's projection at time instant t_{cur} onto the new camera reference plane $\mathbf{\Omega}_{\text{cam}}(t_{\text{cur}})$.

In this context, an homography $\mathcal{H}$ models the relative displacement of P' within the camera reference plane [Pre06]:

$$\begin{aligned} \mathcal{H} : \mathbf{\Omega}_{\text{cam}}(t) &\longrightarrow \mathbf{\Omega}_{\text{Cam}}(t+1) \\ P'_{\text{ref}} &\longmapsto P'_{\text{cur}} \propto \left(\boldsymbol{R}_{\text{cam}} + \frac{\left\langle \overrightarrow{T_{\text{cam}}}, \overrightarrow{n} \right\rangle}{d_{\text{cam}}(t)} \right) \cdot P'_{\text{ref}} \end{aligned} \tag{2.9}$$

where $\boldsymbol{R}_{\text{cam}}$ and $\overrightarrow{T_{\text{cam}}}$ are respectively the rotation and the translation of the camera from instant t_{ref} to instant t_{cur}. $\overrightarrow{n}$ is a vector normal to $\mathcal{O}$'s plane surface, and $d_{\text{cam}}(t_{\text{cur}})$ the distance between P and the focus of the camera at instant t_{cur}. This situation is illustrated in figure 2.2.2. Non-planar structures can also be handled through more complex homographies (see [Pre06] for further information on the matter). Besides, homographies avoid ill-conditioned problems brought by the parametrization of a 3D transform. For all these reasons, they are commonly used in computer vision applications.

2.2.3 Motion models based on geometrical patterns

While parametric models are able to represent a wide range of deformations, they are often limited to global motion analysis and compensation, as they cannot locally adapt their parameters to the local displacements and deformations. As a solution, it has been proposed to partition the image domain into patterns whose deformations will locally adapt to the motion field variations. In each pattern, the motion field is interpolated from pattern's motion parameters.

- **Blocks in translation** describe the motion field through uniformly moving blocks of pixels; they will be reviewed in section 2.2.3.1.
- **Active meshes** describe local deformations through a set of warping operations; they will be reviewed in section 2.2.3.2;
- **Deforming regions** adapt their shape to the images contents and describe the deformation of semantic or pseudo-semantic elements. They will not be further reviewed in this chapter as they mostly rely on local parametric models to track the displacements and the deformations of each region. They may rely on active contours [KWT88] and/or Kalman filtering [Kal60] to interpolate hidden objects boundaries, thus handling occlusions.

2.2.3.1 The translational block-based model

The *translational block-based model* partitions the image plane into blocks. To each of these blocks is granted a single displacement vector $\vec{d}$ which uniformly describes the motion of its pixels: the deformation of the reference image I_{ref} into the current image I_{cur} is described as a set of translations applied to each bock of the partition.

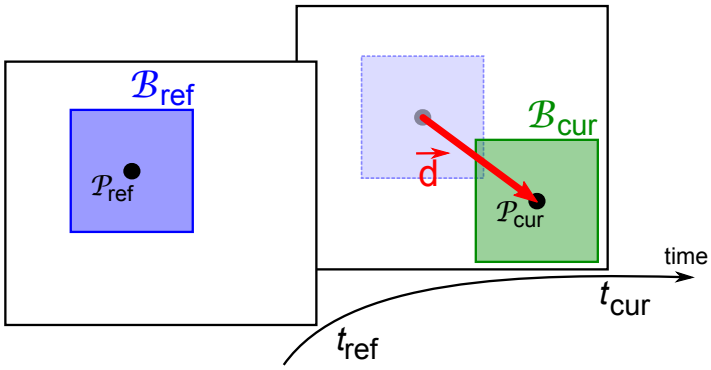


Figure 2.2.3: *Translational motion representation*

Figure 2.2.3 illustrates this translational block-based model, often referred to as Block Motion Compensation (BMC). Here, the reference block $\mathcal{B}_{ref}$ is centred on pixel $\mathcal{P}_{ref}(x, y)$, at time instant t_{ref}. At time instant t_{cur}, $\mathcal{B}_{ref}$ is matched with target block $\mathcal{B}_{cur}$, centred on pixel $\mathcal{P}_{cur}(x + dx, y + dy)$. Thus, $\forall (x, y) \in \mathcal{B}_{ref}$

$$\mathcal{B}_{ref}(x, y) = \mathcal{B}_{cur}(x + dx, y + dy) \tag{2.10}$$

where $\vec{d} = [dx, dy]^T$ is the displacement vector. Its accuracy proved to be essential to the compensation process: sub-pixellic motion vectors are now commonly used [BDH99].

a Fixed and variable-size Block Motion Compensation

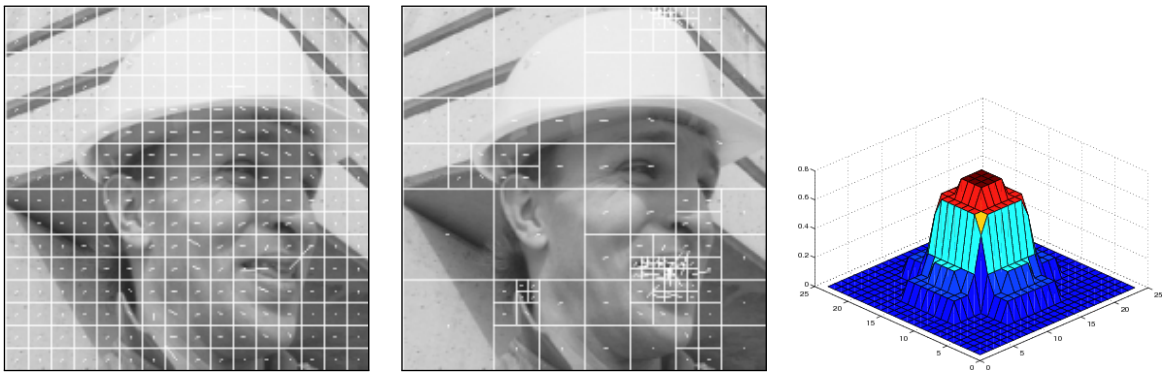
Various partitions of the image plane have been proposed; some of them may provide adaptive features by adapting the block sizes according to the image contents. Basic approaches, known as Fixed Size BMC (FSBMC), simply partition the image plane into regular blocks of fixed size (see figure 2.2.4a), thus do not provide any adaptation to the image content [JJ81, RCN97]. Conversely, alternative approaches have been proposed to optimize the locations of the motion vectors: uniformly moving regions are partitioned into large blocks, while non-uniformly moving areas are partitioned into small blocks. They are known as Variable Size BMC (VSBMC) [CYC90, SB94, GMR96, Mar97] (figure 2.2.4b).

Consequently, the translational block-based model can be locally adapted to the non-stationary nature of the motion information held by an image. At the block boundaries, yet, the influence zone of the current block abruptly ends, substituted by the influence zone of one of its neighbours. This generally result in *blocking effects* in motion compensated images: the block boundaries are much too obvious. In other words, block-based models may be very handy when it comes to represent local motion discontinuities, but cannot provide a smooth and continuous motion field.

b Reducing the blocking artefacts

Typically, a *deblocking filter* [KYKR99, LP01, Ric03, RM06] is used to reduce the blocking effects. As a post-processing technique, it simply smoothes the images along the blocks boundaries. Alternatively, intrinsically blocking-free representations have been provided and get rid of the block boundaries problem.

It was shown in section 1.2.2.2 that blocking artefacts, in block-based transforms (*e.g.* LOT, LBT, etc), can be reduced through lapped transforms. Similarly, it has also been proposed to partition the image plane into overlapping blocks in translation [NO92, AKOK92, OS94, CHJ+06]. Such techniques are known as Overlapped BMC (OBMC): any motion compensated pixel is a weighted combination of several contributions provided by a set of overlapping blocks. Appropriate weighting windows are centred on each block centre and play down the pixels located near the block boundaries (see figure 2.2.4c). Besides, overlapping blocks are also able to represent slight deformations and provide an interesting trade-off between translational and affine motions.

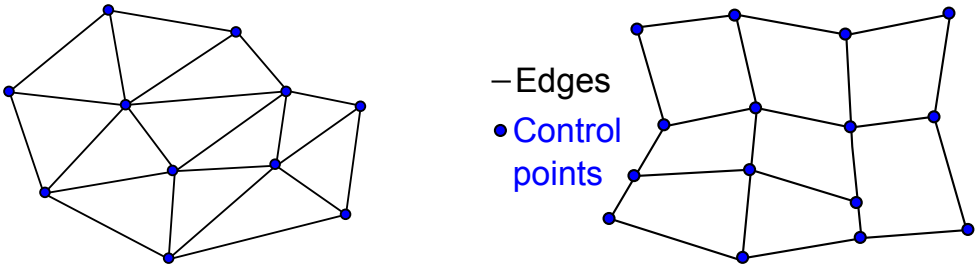


(a) *FSBMC: fixed block partitioning* (b) *VSBMC: Adaptive block partitioning* (c) *OBMC: weighting window*

Figure 2.2.4: *Various kinds of partitions of the image plane into blocks*

2.2.3.2 Active meshes and control grid interpolation

While block based representation models the motion as a set of local displacements, it can barely model any kind of deformation. Consequently, control grids have been proposed as a way to model both local displacements and local deformations. Control grids emerged from the theory of finite elements, first introduced by Argyris [Arg60] and Clough [Clo60]. Later, Zienkiewicz provided in [ZT67] a comprehensive summary of finite elements. Peculiar to this theory, control grids have been used to model the behaviour of complex structures by splitting them into small connected independent cliques known as *meshes*.



(a) *Triangular mesh* (b) *Quadrangular meshing*

Figure 2.2.5: *Triangular and quadrangular meshing*

In terms of topology, a *control grid* partitions a plane into polygonal elements [Lec99]. Each of these elements is called a *mesh*; its vertices are called *nodes* or *control points*. Any polygon can be used to partition the plane; triangular and quadrangular meshes, however, are commonly used. Both kinds of meshes are illustrated in figure 2.2.5.

Naturally, it has been proposed to partition the image plane into meshes. *Deformable control grid*, also known as *active meshes* [Lec99, Mar00] can then model local displacements and deformations by displacing their control points accordingly. Control grids are ruled by the non-superposition constraint: the equivalent graph needs to be planar [Lec99]. In particular, any displacement leading to overlapping meshes is forbidden.

Various partitions of the image plane have been proposed. Regular meshes initially partition the image into fixed size meshes, while irregular meshes may adapt the size of the meshes to the contents of the image [YW94, AT97a]. In addition, hierarchical control grids have been proposed to provide a robust representation of the deformation [HH94, TEST96, BTZ+99, HMCP01, ARAM03, MJZ06]

a Parametrization of a deformable control grid

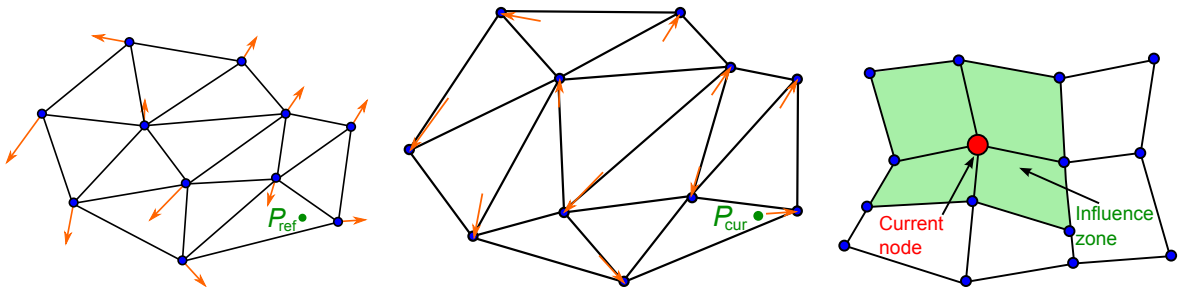
Let (resp.) I_{ref} and I_{cur} be two images captured at (resp.) time instants t_{ref} and t_{cur}, defined over (resp.) domains $\mathbf{\Omega}_{\text{ref}}$ and $\mathbf{\Omega}_{\text{cur}}$. Let $\mathcal{M}(t)$ be a deformable control grid partitioning the image I_t, and $\mathcal{C}(t) = \{ C_i(t)\}_{i=1\ldots N}$ its N control points.

According to the motion information which links images I_{ref} and I_{cur}, $\mathcal{M}(t_{cur})$ is a deformed version of $\mathcal{M}(t_{ref})$ given by

$$\mathcal{M}(t_{cur}) = w_{ref \to cur}\Big(\mathcal{M}(t_{ref})\Big) \tag{2.11}$$

where $w_{ref \to cur}$ is a deformation operator, known as *warping*. Provided that meshes are not overturned, this transformation is non-ambiguous, total and symmetric: to any point $P_{ref} \in \mathbf{\Omega}_{ref}$ corresponds a unique match $P_{cur} \in \mathbf{\Omega}_{cur}$ (see figure 2.2.6). Reciprocally:

$$\exists w_{cur \to ref} = w^{-1}_{ref \to cur} \qquad \text{such that} \qquad \mathcal{M}(t_{ref}) = w_{cur \to ref}\left(\mathcal{M}(t_{cur})\right) = w^{-1}_{ref \to cur}\left(\mathcal{M}(t_{cur})\right) \tag{2.12}$$



(a) *Initial control grid at instant t_{ref}* **(b)** *Deformed control grid at instant t_{cur}* **(c)** *TBCGI's meshes influence zone*

Figure 2.2.6: *Deformable meshing: interpolation of the motion*

In each control point $C_i(t)$, the motion vector is given by the displacement it undergoes. Elsewhere, the displacement is interpolated from neighbouring control points by an inner-mesh interpolation process. The displacement $\overrightarrow{d_{ref \to cur}}(P) = \overrightarrow{P_{ref}\,P_{cur}}$ of an arbitrary point P is then given by

$$\overrightarrow{d_{ref \to cur}}(P) = \begin{cases} \overrightarrow{d_{ref \to cur}}(C_i) & \text{if } P = C_i \in \mathcal{C} \\ \sum_{j=1}^{\overline{N}} \phi_j\left(P - C_j(t_{ref})\right) \cdot \overrightarrow{d_{ref \to cur}}(C_j) & \text{elsewhere} \end{cases} \tag{2.13}$$

where $\left\{C_j(t)\right\}_{j=1\ldots\overline{N}}$ is the set of control points whose influence zone contains P, and $\overrightarrow{d_{ref \to cur}}(C_j)$ is the displacement vector of control point C_j. Finally, $\phi(P - C_j(t))$ is the inner-mesh interpolation function of C_j: it outputs the relative weight of C_j's contribution to P's displacement. This is known as Control Grid Interpolation (CGI).

For instance, in [SB91a], Sullivan and Baker partition the image plane into rectangular meshes with their Tri-Bilinear CGI (TBCGI). The influence zone of each control point is limited to adjacent meshes: here, $\overline{N} = 4$ (figure 2.2.6c). A bilinear function (see section 2.2.2.3) is used to perform the inner-mesh interpolation. Their approach will be further reviewed in section 2.4.2.1.

b Handling both continuities and discontinuities of the motion field

The ability to project the entire image domain into itself, in control grids, is an essential feature. Indeed, successive deformation can be applied to the same deformable control grid $\mathcal{M}$, enabling the motion information of a whole group of pictures to be continuously described from the $\mathcal{M}$'s successive warping. In particular, deformable control grids can continuously track the deformation of a patch of texture on the long run, this exhibiting the notion of motion trajectory.

From the inner-mesh interpolation process, control grids naturally provide a continuous representation of the motion field over the whole image domain, while accounting for its local variations. Unlike the block-based model, they are not limited to translations, but model a larger set of transformations: translations, rotations of small angles, divergences (stretching and contraction).

However, control grids are not able to properly capture the discontinuities of the motion field. To address this issue, hybrid approaches allowed meshes to be disconnected when needed [IM00]; alternatively, solutions that use rupture lines have also been proposed [Cam04, CPW+09]. In the end, yet, these solutions are not as good as the block-based model regarding their ability to represent motion discontinuities.

2.3 Tools and techniques for motion estimation

Previous section introduced a large range of motion models. No clue, however, was provided regarding the way their parameters Θ can be estimated. Numerous estimation algorithms have been propose, and generally carry out a combination of several basic steps. This section will review common tools and techniques likely to be used in motion estimation algorithms. Once again, let I_{ref} and I_{cur} be two images respectively captured at reference instant t_{ref} and current instant t_{cur}. Section 2.1.4 showed that motion compensation can be performed in both forward and backward directions. Consequently, motion estimation provides a set of optimal motion parameters $\Theta^{\mathcal{V}}$ with respect to the considered compensation direction:

1. either $\Theta^{\mathcal{V}}$ optimizes $\widehat{I_{cur}}$, the **backward** motion compensated prediction of I_{cur} from I_{ref};
2. or $\Theta^{\mathcal{V}}$ optimizes $\widehat{I_{ref}}$, the **forward** motion compensated prediction of I_{ref} from I_{cur}.

Video compression schemes, however, build the temporal prediction of I_{cur} from I_{ref}, through a backward motion compensation step. In this case, solution 2 requires for the motion field to be inverted in order to retrieve the backward compensation parameters. As this operation is ambiguous (few models provide a bijective representation of the motion), most estimation techniques directly optimize the backward motion compensated prediction $\widehat{I_{cur}}$ (solution 1). Furthermore, it was shown in [Mar00] that, in active meshes, backward approaches generally provide better results than forward ones.

Generally, the optimization is done through the minimization of an energy measured between I_{ref} and I_{cur}. This search for optimal parameters can be performed in numerous ways.

1. **Gradient-driven** techniques directly minimize the Displaced Frame Difference (DFD). They will be briefly reviewed in section 2.3.1.
2. **Matching criteria** are used to evaluate the distance between two pixellic information which are matched by the motion estimation. They will be reviewed in section 2.3.2.
3. **Regularization techniques** aim at limiting the incoherences of estimated motion fields. Not only do they prevent the estimation problem from being ill-conditioned, but they also robustify the estimation. They will be further investigated in section 2.3.3.
4. **Probabilistic approaches** model the pixellic intensity error by a white noise, and the optical flow by a random field (Markov Random Fields (MRFs), Gibbs Random Fields (GRFs), Conditional Random Fields (CRFs), Gaussian random fields, ...) which is then regularized [BA91, HB93]. Alternatively, it was proposed in [Li07] to use an AR model to estimate the motion field. Probabilistic techniques usually require a large number of computations.
5. **Frequency approaches** rely on the fact that translations, in the spatial domain, correspond to phase-shifts in the frequency domain [DCM87, Tho87]. They are highly resilient to noise and provide an accurate estimation of the motion. However, they do not provide any information regarding the positions in which detected displacements occur. Consequently, they are generally followed by a classic matching technique. They also require a large number of computations, including a transform and its inverse.

2.3.1 Gradient-driven search strategies

As seen in section 2.1.3.1, pixel-based motion estimation techniques venture the *pixellic intensity invariance hypothesis* -see equation (2.2)- [HS81, BL02]. Both luminance and chrominance information can be used to drive the estimation. Whichever the motion model is, this hypothesis may be expressed as the minimization of the DFD

$$\begin{aligned} \varepsilon_{DFD} =& \arg\min_{\Theta^{\mathcal{V}}} \left\| \mathrm{DFD}(x, y, \overrightarrow{d}) \right\| \\ =& \arg\min_{\Theta^{\mathcal{V}}} \left\| I_{ref}(x + dx_{\Theta^{\mathcal{V}}}, y + dy_{\Theta^{\mathcal{V}}}) - I_{cur}(x, y) \right\| \end{aligned} \tag{2.14}$$

where ε_{DFD} is the minimized DFD residue. $[dx_{\Theta^{\mathcal{V}}}, dy_{\Theta^{\mathcal{V}}}]^T$ is the displacement vector provided by the motion parameters $\Theta^{\mathcal{V}}$. Several strategies may be used to address this problem:

1. **global approaches** minimize the DFD of all pixel at the same time;
2. **local approaches** iteratively minimize the DFD on a pixel-by-pixel basis, or on a parameter-by-parameter basis;
3. **semi-global approaches** lie in between global and local approaches, and globally minimize the DFD of local areas of the image domain $\mathbf{\Omega}$ [AT97a, Lec99].

2.3.1.1 Global minimization

Global approaches minimize an error functionals defined on $\mathbf{\Omega}_I$. Several iterations are successively performed to reach an optimal solution. Each iteration aims at reducing the norm of the DFD of all pixels from the image domain $\mathbf{\Omega}_I$. Though many error functional have been provided throughout the years, most of them are built after the *apparent motion gradient constraint* equation.

a The apparent motion gradient constraint equation

From the pixellic intensity invariance hypothesis, it naturally follows that the material derivative $\frac{\mathcal{D}I}{\mathcal{D}t}$ of an image $I(x, y)$ is null. This results in the *apparent motion gradient constraint* equation

$$\frac{\mathcal{D}I}{\mathcal{D}t} = \frac{\partial I}{\partial t} + \overrightarrow{v} \cdot \overrightarrow{\nabla I} \triangleq 0 \tag{2.15}$$

where $\overrightarrow{v}$ is the velocity vector, which can be easily linked to the displacement vector from equation (2.1). In practice, equation (2.15) is generally not verified due to occlusions, aperture issues, and noise. Instead, the estimation aims at minimizing the residual prediction error ξ

$$\min_{\Theta^v} \xi = \frac{\partial I}{\partial t} + \overrightarrow{\nabla I} \cdot \overrightarrow{v} \tag{2.16}$$

b Minimization of the error functional

Classic minimization techniques can be used, including differential approaches (*e.g.* Gauss-Seidel), exhaustive or sub-optimal searches, ... The differential approach, in particular, cancels ξ's partial derivatives with respect to vx and vy, the horizontal and vertical components of the optical flow. The corresponding matrix system is iteratively solved until it converges to an acceptable solution.

2.3.1.2 Local minimization

Among gradient-driven search strategies, local minimization techniques consider each pixel independently and optimize corresponding motion parameters accordingly. Such techniques are called *pel-recursive* and directly minimize the energy hold by the DFD by locally tuning the motion parameters. They successively build a prediction of the displacement vector $\overrightarrow{d}$. At k^{th} iteration, $\overrightarrow{d}^k = \left[dx^k, dy^k\right]^T$ is predicted from its value at previous iteration $\overrightarrow{d}^{k-1} = \left[dx^{k-1}, dy^{k-1}\right]^T$.

a A recursive scheme performed on a pixel-by-pixel basis

Traditionnaly, pel-recursive techniques are derived from the first order Taylor serie expansion of $\mathrm{DFD}(x, y, \overrightarrow{d^k})$ around $\overrightarrow{d^{k-1}}$. Such an approach naturally leads to the formulation of a local gradient descent on $\overrightarrow{d}$:

$$\mathrm{DFD}(x, y, \overrightarrow{d}^k) = \mathrm{DFD}(x, y, \overrightarrow{d}^{k-1}) - \left(\overrightarrow{d}^{k-1} - \overrightarrow{d}^k\right) \cdot \overrightarrow{\nabla I}(x + d_x^{k-1}, y + d_y^{k-1}) \tag{2.17}$$

Alternatively, this directly formulated as a function of $\overrightarrow{d}^k$ and $\overrightarrow{d}^{k-1}$, and

$$\overrightarrow{d^k} = \overrightarrow{d}^{k-1} - \varepsilon \cdot \underbrace{\mathrm{DFD}(x, y, \overrightarrow{d}^{k-1}) \cdot \overrightarrow{\nabla I_t}(x + dx^{k-1}, y + dy^{k-1})}_{\text{correction term}} \tag{2.18}$$

where ε is a gain term which controls the convergence of the method. This approach is used by Netravali and Robbins in [NR84]. It is later modified in [Sab84] by Sabri into a simplified algorithm where the prediction step is fixed (the correction term is replaced by its -1 or +1 according to its sign).

b Adaptive pel-recursive techniques

Classic pel-recursive techniques usually suffer from their inability to adjust the value of the prediction step according to the image contents. Consequently, adaptive gains ε were proposed to overcome this particular issue. In [WR84], Walker and Rao advised that ε should be low in areas whose spatial activity is high, and high in areas whose spatial activity is low: ε is then inversely proportional to the norm of the image spatial gradient. In [CR83], Cafforio and Rocca further modify ε and add a correction term σ which stabilizes the algorithm whenever the spatial gradient vanishes:

$$\varepsilon = \frac{1}{\sigma^2 + \left\| \overrightarrow{\nabla I_t}(x + dx^k, y + dy^k) \right\|^2} \tag{2.19}$$

Nevertheless, the initialization of such algorithms strongly influences their outcome; error functionals are seldom convex and there is a great risk of reaching a local minimum. As a consequence, the strategy used to initialize the gradient descent is crucial. Pel-recursive techniques often rely on temporal or hierarchical predictions.

2.3.2 Matching criteria

As motion estimation techniques aim at providing a prediction as accurate as possible, they need to assess the distance between the provided motion compensated image $\widehat{I}$ and the original image I (see equation (2.16)). This distance ξ is generally expressed as a function of the difference between $\widehat{I}$ and I. According to the provided motion representation, ξ may be computed over a single pixel, or over a group of pixels. Let $\mathcal{R}$ be a region from I; $\mathcal{R}$ is an arbitrary set of pixels defined over the domain $\mathbf{\Omega}_{\mathcal{R}}$, and may only consist of a single pixel.

2.3.2.1 Absolute and quadratic errors

Absolute and quadratic errors are respectively based on the absolute value and the squared values of the difference between the intensity of an original pixel and the intensity of its motion compensated prediction. Consequently, the distance between a region $\mathcal{R}$ and its prediction $\widehat{\mathcal{R}}$ may be measured by the Mean Absolute Errors (MAE) or the Mean Squared Errors (MSE)

$$\begin{cases} \xi_{\text{MAE}}(\mathcal{R}, \widehat{\mathcal{R}}) = \dfrac{\xi_{\text{SAE}}}{|\mathbf{\Omega}_{\mathcal{R}}|} = \dfrac{1}{|\mathbf{\Omega}_{\mathcal{R}}|} \displaystyle\sum_{(x,y)\in\mathbf{\Omega}_{\mathcal{R}}} \left| \mathcal{R}(x,y) - \widehat{\mathcal{R}}(x,y) \right| \\ \xi_{\text{MSE}}(\mathcal{R}, \widehat{\mathcal{R}}) = \dfrac{\xi_{\text{SSE}}}{|\mathbf{\Omega}_{\mathcal{R}}|} = \dfrac{1}{|\mathbf{\Omega}_{\mathcal{R}}|} \displaystyle\sum_{(x,y)\in\mathbf{\Omega}_{\mathcal{R}}} \left(\mathcal{R}(x,y) - \widehat{\mathcal{R}}(x,y) \right)^2 \end{cases} \tag{2.20}$$

where $|\mathbf{\Omega}_{\mathcal{R}}|$ is the cardinal of $\mathcal{R}$'s support, *i.e.* the number of pixels it holds. Often, $|\mathbf{\Omega}_{\mathcal{R}}|$ is fixed, such that distances do not need to be normalized. Sum of Absolute Errors (SAE) and Sum of Squared Errors (SSE) (also known as Sum of Absolute Differences (SAD) and Sum of Squared Differences (SSD)) simply sum individual errors instead of computing their mean over $\mathbf{\Omega}_{\mathcal{R}}$. The SAE, in particular, is often preferred to the SSE in image and video compression as it is well suited to signal coding and requires fewer computations than the SSE.

Numerous variants have been proposed. In [CCCL95], the distance corresponds to the maximum difference between a pixel from $\mathcal{R}$ and its match into $\widehat{\mathcal{R}}$. Its minimization is called *minimax*. In [FS98], the *s*MAE is designed such that if offers a criterion whose characteristics lie in between the MSE and the MAE. It is obtained by multiplying the MAE by its mean deviation s.

2.3.2.2 Sum of Absolute Transformed Differences

The Sum of Absolute Transformed Differences (SATD) takes the frequency transforms of the original block of pixels $\mathcal{R}$ and of its prediction $\widehat{\mathcal{R}}$. It then computes the sum of the absolute differences between both sets of transforms coefficients. Usually, the Hadamard transform $\mathcal{H}$ is used, and

$$\xi_{\text{SATD}}(\mathcal{R}, \widehat{\mathcal{R}}) = \sum_{(\omega_x,\omega_y)\in(\mathbf{\Omega}_{\mathcal{R}})} \left| \mathcal{H}(\widehat{\mathcal{R}})(\omega_x, \omega_y) - \mathcal{H}(\mathcal{R})(\omega_x, \omega_y) \right| \tag{2.21}$$

2.3.2.3 Cross-correlation coefficient

The cross-correlation usually measures the similarity between two waveforms. It is especially used to search for a pattern within a signal of long-duration. Its expression is close to a convolution, and writes as

$$\xi_{CC}(\mathcal{R}, \widehat{\mathcal{R}}) = \sum_{(x,y)\in\mathbf{\Omega}_{\mathcal{R}}} \Big[\mathcal{R}(x,y) - \mu_{\mathcal{R}}\Big] \cdot \Big[\widehat{\mathcal{R}}(x,y) - \mu_{\widehat{\mathcal{R}}}\Big] \tag{2.22}$$

where $\mu_{\mathcal{R}}$ and $\mu_{\widehat{\mathcal{R}}}$ are respectively the average pixellic intensities of regions $\mathcal{R}$ and $\widehat{\mathcal{R}}$. The cross-correlation coefficient usually provides good results, but suffers from its computational complexity.

2.3.2.4 Pel difference classification

The Pel Distortion Classification (PDC) counts the number of pixels of $\mathcal{R}$ whose absolute difference between its original and motion compensated intensities are inferior or equal to a given threshold T. Such criterion may be used in signature based Block Matching Algorithms (see section 2.3.3.2), by successively reducing the threshold T [CRGP94].

$$\begin{aligned} \xi_{PDC} &= \sum_{(x,y)\in\mathbf{\Omega}_{\mathcal{R}}} \delta\Big(\mathcal{R}(x,y) - \widehat{\mathcal{R}}(x,y)\Big) \\ \text{where} \quad \delta\Big(\mathcal{R}(x,y) - \widehat{\mathcal{R}}(x,y)\Big) &= \begin{cases} 1 & \text{if} \quad \Big|\mathcal{R}(x,y) - \widehat{\mathcal{R}}(x,y)\Big| \leq T \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{2.23}$$

2.3.2.5 Histogram and integral projection

Alternatively, one may rely on statistical features to compute the distance between $\mathcal{R}$ and its prediction $\widehat{\mathcal{R}}$. Their histograms, for instance, can be compared [HBL+02]. Similarly, integral projections [SS96] can be used, in blocks, to process complete rows or columns of pixels at once. In this case, horizontal and vertical measurements can be performed separately: horizontal and vertical displacement searches can be performed concurrently.

2.3.2.6 Robust estimators

Previous criteria are, for most of them, very sensitive to noise. Consequently, robust criteria have been proposed; some of them are reviewed in [OB94]. Generally, they are much more computationally demanding. The Least-Median-of-Squares (LMedS), for instance, measures the median value of the squared difference on $\mathbf{\Omega}_{\mathcal{R}}$. It allows the search to filter out up to 50% of the outliers.

M-estimators have also been proposed as a way to provide a robust measure of the distance between two sets of data. They minimize the influence of the observations whose probabilities are low. For instance, Geman and McLure's estimator is used by Black and Anandan in [BA91]. Alternatively, Tukey's bi-weight estimator is used by Odobez and Bouthémy in [OB94]. Both of them use a bounded influence function, such that outliers do not have a strong impact on the measure

2.3.3 Encouraging and exploiting the regularity of the motion field

Minimization techniques are often hybridized with additional regularization techniques. Indeed, several motion estimation specific issues (including aperture and occlusions problems) requires the motion field to be regularized. This prevents the problem from being ill-determined, and forces the motion field to behave under specific regularity constraints. The regularization can be performed in various manners:

- the motion representation itself, as mentioned in the introduction of section 2.2, may implicitly regularize the motion field from its ability to represent coherent motions only;
- at the estimation, the error functional ξ being minimized can be modified to include a regularity penalty ξ_R. This approach will be investigated in section 2.3.3.1;

- the estimation process can be locally modified to avoid incoherent motion parameters. Conversely, the estimation process can benefit from the assumption that the motion field is regular. In particular, the estimation process may be drastically simplified by only evaluating a subset of the possible deformations. Sub-optimal search strategies, for instance, sparsely parse the search area $\mathcal{A}$ to find an appropriate motion vector. They will be reviewed in section 2.3.3.2.
- finally, the overall minimization process can be globally modified to encourage the regularity through hierarchical approaches. These will be reviewed in section 2.3.3.3.

2.3.3.1 Including explicit regularization terms into the error functional

In [HS81], Horn and Schunck, add penalize incoherent motions by including a smoothing term ξ_C to the error functional: this tends to minimize the amplitude of the optical flow gradient.

The minimization of the pixellic error ξ_L is now subject to the constraint ξ_C. Numerous techniques have been provided to solve such problems; Lagrange multipliers, in particular, are commonly used. However, they are highly sensitive to noise; iterative techniques such as gradient descents and other numerical optimization techniques are often preferred. In this case, the error functional is a weighted sum of the pixellic error ξ_L and the regularity constraint ξ_C

$$\xi^2 = \sum_{(x,y)\in\Omega} \xi_L^2 + \alpha^2 \xi_c^2 \tag{2.24}$$

where α is a weighting coefficient which sets the relative importance of the regularity term ξ_C^2 with respect to the prediction error ξ_L^2, and ξ_L the matching error. Horn and Schunck applied this constraint in an isotropic fashion, which generally provides a poor estimation of motion discontinuities.

Later approaches proposed the use of explicit anisotropic constraints. In [WSD81], Wu *et al.* propagate the optical flow along the contours of the motion. Similarly, Nagel minimizes the optical flow in a direction perpendicular to the spatial gradient in [Nag83]. In [Hil86], Hildreth applies Horn & Schunck smoothing term along the images contours. In [Enk88], Enkelmann penalizes the variations of the motion field according to those of the luminance. In [Fog91], Fogel use a directed smoothness constraint and locally weight the optical flow, thus introducing an additional adaptivity to its estimation.

Alternatively, implicit constraints were also proposed and pre-filter either the images or the optical flow, In [LK81], Lucas and Kanade apply Horn and Schunck's isotropic constraint to a local neighbourhood only. In [Bru01], Bruno filters the partial derivatives of the optical flow with a Gabor filters bench.

2.3.3.2 Heuristic-based search strategies

Section 2.2.3 showed that the motion field can be described through individual deformations of patterns in motion. In such cases, heuristics can be used to search the best motion parameters within a list of parameters candidates. Optimal Full Search (FS) strategies exhaustively evaluate all the candidates from the considered search area $\mathcal{A}$, thus requiring a large amount of computations to perform.

Consequently, numerous suboptimal search strategies have been proposed and perform a non-exhaustive parameter search: they provide a trade-off between the appropriateness of the estimated parameters and the search speed. Unlike FS strategies, they may end up in local minima. They generally fall into four categories:

1. **Signature-based algorithms** [Kha89, AFJ90, CRGP94, ZK96, KDST05] aim at reducing the search complexity, while preserving most advantages from the FS. The search is generally performed in several stages. During the first stage, every position from the search area $\mathcal{A}$ is evaluated via a computationally simple matching criterion. Only most promising positions are further evaluated in the next stages using progressively more selective matching criteria. The Successive Elimination Algorithm (SEA) [LS95, GDZ00], for instance, successively eliminates remaining candidates according to the sum of their pixels.
2. **Distance-diluted algorithms** [Gil88, PM96] are inspired from the fact that humans cannot accurately perceive fast moving objects. Consequently, slow moving areas may need to be accurately processed, while fast moving are can be more crudely processed. Distance-diluted algorithms evaluate fewer and fewer candidates as they get farther away from the center of the search area: the search step is a monotonically increasing function of the distance to $\mathcal{A}$'s center.

3. **Locality-based algorithms** [Gil88, JRG92, NKPS95] are based on the *principle of locality* which suggests that very good matches are likely to be found in the neighborhood of other good matches. They initially examine a number of sparsely spaced positions within the search area, then narrow the search on promising areas (*e.g.*: the binary search divides the search area into four quadrants and performs a FS within a single promising quadrant).
4. **Quadrant-monotonic algorithms** [JJ81, KIH+81, PHS87, ZK96, ZM97] are special cases of locality-based algorithms and consider the image to fit the quadrant monotonic model. This model assumes that the value of the distortion function increases with the distance to the minimum distortion position. This paradigm has grown very popular, and was used to derive numerous search algorithms. They parse the search areas from specific patterns; they are very popular in Block Matching Algorithms. They will be further reviewed in section 2.4.1 .

2.3.3.3 Hierarchical minimization

Whichever the numerical algorithm used to optimize the motion parameters Θ, it may get stuck onto a local minimum. On top of previously mentioned regularization techniques, hierarchical approaches have been provided as ways to robustify and speed up the estimation. They optimize the parameters at different levels of precision; either using different version of the input signal, or using different versions of the motion model. Hence, two main families of hierarchical search strategies are available.

1. **Multiresolution techniques** subsample the original images at different resolutions [Fog91, BL91]. Pyramids of images (Gaussian, Laplacian), for instance, are commonly used [AAB+84]. For each resolution level, the motion parameters are first predicted from their values at previous level, then refined by another estimation stage performed on subsampled images from the current level.
2. **Multigrid techniques** define a set of motion models of various accuracy [Duf94]. For instance, pattern-based models may use hierarchical patterns. For each description level, the motion parameters are first predicted from their values at previous level. At next level, the corresponding motion model is initialized from the current motion model.

In both cases, moving up into the hierarchy tends to smooth the error curve, thus reducing the risk of falling into a local minimum. Several strategies have been proposed to run the different resolution or description levels, including

- *coarse-to-fine* strategies [Fog91]: they start by estimating the motion parameters of the hierarchy top-level, then progressively climb down the hierarchy;
- *fine-to-coarse-to-fine* strategies [Duf94]: they start by estimating the motion parameters at the bottom of the hierarchy, then progressively climb up to its top, and finally back to its bottom.

2.4 A focus on block-based and mesh-based techniques

In an eye to compression, it is crucial for the motion estimation to rely on a model which is described through as few parameters as possible, since they have to be transmitted. However, the model still needs to be able to represent as many deformations as possible. In this connection, both block-based and mesh-based models have been identified as appropriate models, and have been largely used in video compression schemes.

1. **Blocks** (see section 2.2.3.1) proved to be very handy regarding the representation of section 2.4.3 motion fields. In addition, they describe the motion field through a small number of parameters: their estimation is particularly **simple**, and the resulting motion information is **compact**. However, they cannot represent any deformation besides simple **translation** (and barely slight deformation through overlapping blocks).
2. **Meshes**, (see section 2.2.3.2) are able to represent a large variety of **deformations**, and naturally provide a **continuous** motion field, but often fail at representing motion discontinuities. In addition, deformable meshes also provide a **continuous** representation of the motion **along the temporal axis**, as several successive warping operation can be applied to the same control grid. They also describe the motion field in a **compact** manner.

This section will explain how their parameters are estimated. Sections 2.4.1 and 2.4.2 will respectively focus on Block Matching Algorithms (BMAs) and CGIs. Section 2.4.3 will then review a few models which hybridize blocks and meshes.

2.4.1 Block matching algorithms

Block Matching Algorithms are motion compensating techniques which rely on the translational block based model seen in section 2.2.3.1. For each block $\mathcal{B}_{cur}$ of the current image partition, its best match $\widehat{\mathcal{B}_{cur}} = \mathcal{B}_{ref}$ is searched into the reference image, within a search area $\mathcal{A}$. $\mathcal{B}$'s motion vector is then given by the displacement pulling apart the positions of $\mathcal{B}$ and $\widehat{\mathcal{B}}$ in the image plane (see figure 2.4.1). A BMA is thus characterized by:

- its **partition** of the image plane into blocks (see 2.2.3.1);
- its **search area** $\mathcal{A}$ which defines the domain within which matching candidates may be found; the maximum displacementsdx_{max} dy_{max} along directions x and $y$are given by $\mathcal{A}$'s dimensions;
- its **search strategy** which drives the way candidates from the search area $\mathcal{A}$ are successively evaluated;
- its **matching criterion** ξ (see 2.3.2) which computes the matching (or distortion) error over the whole domain of $\mathcal{B}$.

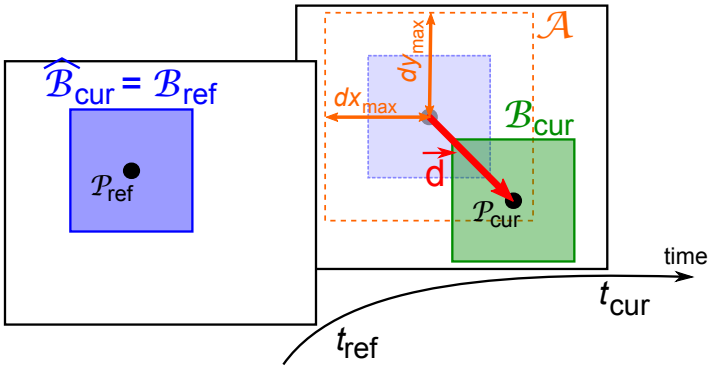


Figure 2.4.1: *BMA: search area and motion vector*

2.4.1.1 Traditional Block Matching Algorithms

a Matching criterion

Traditionally, BMAs partition the image plane into disjoints blocks, thus enabling the estimation to process each block individually. Hence, the motion vector of each block $\mathcal{B}_{cur}$ is obtained by minimizing its matching error ξ with respect to a candidate from the reference image $\widetilde{\mathcal{B}_{ref}}$.

$$\vec{d} = \underset{\vec{d} \in \mathcal{A}}{\arg\min} \quad \underset{(x,y)\in\mathbf{\Omega}_{\mathcal{B}}}{\xi} \left(\mathcal{B}_{cur}(x, y) - \widehat{\mathcal{B}_{ref}}(x + dx, y + dy) \right) \tag{2.25}$$

where $\vec{d} = [dx, dy]^T$ is a displacement vector, such that $-dx_{max} \leq dx \leq dx_{max}$ and $-dy_{max} \leq dy \leq dy_{max}$. Finally, $\mathbf{\Omega}_{\mathcal{B}_{cur}}$ is the support of $\mathcal{B}_{cur}$. Among matching criteria reviewed in 2.3.2), the SAD is very popular for BMAs. Alternatively, SSE and SATD are also employed.

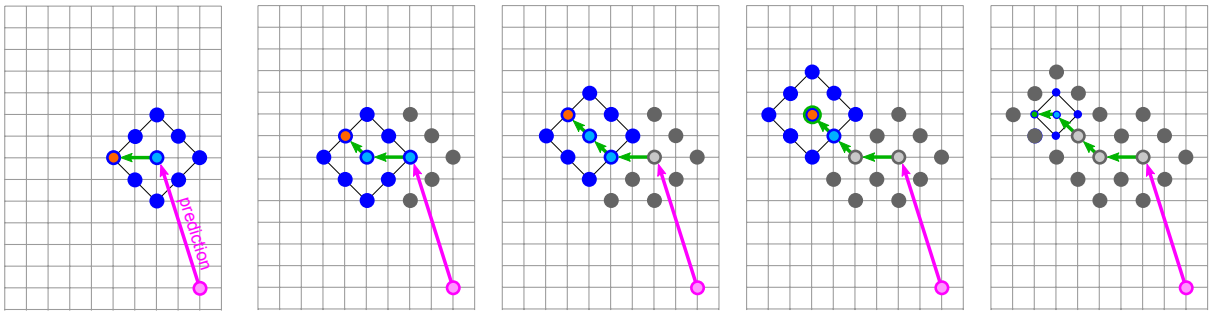
b Search strategies

According to the dimensions of the search area $\mathcal{A}$, the number of candidates to evaluate may be very high, especially if sub-pixellic positions are also considered. Computationally demanding Full Search (FS) evaluates every candidate from $\mathcal{A}$; they are seldom if ever used in practice. Instead, sub-optimal search strategies are typically employed. These have been briefly reviewed in section 2.3.3.2; BMAs, however, have been mostly relying on quadrant-monotonic approaches. The first quadrant monotonic algorithm was proposed by Jain and Jain in [JJ81], within which they introduce the 2-Dimensional Logarithmic (TDL) search which recursively examines the candidates following a cross pattern. Since then, a wide variety of search patterns have been proposed, including:

- **crosses**: TDL search [JJ81] and its variants [Gha90];
- **squares**: n-step searches (Three-Step Search (TSS) [KIH+81], Four-Step Search (FSS) [PM96]);

- **horizontal/vertical directions**: One at a Time Search (OTS), Orthogonal Search Algorithm (OSA) [PHS87];
- **diamonds**: Diamond Search (DS) [ZM97] and its variants [HM99, ZM00, TAL02, CJJ03, UPND07];
- **spirals**: spiral search [ZK96, KDST05].

Currently, state of the art video compression schemes mostly rely on the Enhanced Predictive Zonal Search (EPZS), which is an improvement of the Predictive MVFAST (PMVFAST), itself being a modified version of the Motion Vector Field Adaptive Search Technique (MVFAST). The MVFAST [HM99] initially predicts the motion vector either from a set of predictors including $[0,0]^T$ and available motion vector of spatially adjacent blocks. Then, it performs a modified diamond search using either a small and/or a large diamond. With the PMVFAST [OAL01], an additional set of motion predictors was introduced, including median and temporal predictors. In addition, adaptive early-stopping criteria may further shorten the search. Finally, EPZS [Tou02] further enlarged the set of motion predictors from additional spatio-temporal predictors and acceleration-based predictors. Typically, the EPZS computes the motion vector from 100 times to 5000 times faster than the Full Search.



(a) *Predicted vector and* 1^{st} *DS step* **(b)** *DS* 2^{nd} *step* **(c)** *DS* 3^{rd} *step* **(d)** *DS* 4^{th} *step* **(e)** *DS final step: a small diamond is used*

Figure 2.4.2: *MVFAST: motion prediction mechanism and modified Diamond Search (DS)*

2.4.1.2 Overlapped Block Motion Compensation

Section 2.2.3.1 mentioned that block-based motion representation suffer from blocking effects. As a solution, the OBMC would split the image plane into overlapping blocks, thus removing the blocks boundaries issues.

a Multi-Hypothesis Expectation

In [Sul93, OS94], the OBMC has been formalized as a special case of weighted block motion estimation. Underlying is the Multi-Hypothesis Expectation (MHE) paradigm: it stipulates that a single motion vector cannot fit the motion field of a whole block of pixels.

From a stochastic perspective, the greater the prediction error is, the less accurate the motion vector is likely to be. In [Sul93], Sullivan uses a conditional distribution to model the the displacement error, as an *a posteriori* probability density function $p(x,y;dx,dy|\xi)$ of the position (x,y), the displacement $\overrightarrow{d} = [dx,dy]^T$, and the residual error ξ on the considered pixel. As a consequence, the motion compensated image $\widehat{I_{\text{cur}}}$ can be expressed as

$$\widehat{I_{\text{cur}}}(x,y) = \int_{dx}\int_{dy} p(x,y;dx,dy|\xi) I_{\text{ref}}(x+dx, y+dy) d(dx) d(dy) \tag{2.26}$$

The difference $\widehat{I_{\text{cur}}} - I_{\text{cur}}$ is known as the Multi-Hypothesis DFD (MHDFD). Alternatively, the probability density function $p(x,y;dx,dy|\xi)$ can be interpreted as a regularization term. What is more, it eases the motion coding process which outputs a more compact motion information.

In OBMC, the probability density function $p(x,y;dx,dy|\xi) = p(dx,dy)$ of the displacement error is simply defined as a function of the motion vectors of neighbouring blocks. It does not depend neither on the position (x,y) nor on the residual error ξ. Consequently, the contribution provided by each block depends on its neighbours. OBMC has been notably used in ITU-T H.263 ([ITU04], see annex F), wherein top, bottom, left and right neighbouring blocks influence the current *macroblock*: this corresponds to cross-shaped overlapping patterns.

b Estimation techniques which process the blocks independently from each other

In [NO92], Nogaki and Ohta propose an Overlapped BME (OBME) algorithm which does not take the blocks inter-dependency into account. Similarly to traditional BMAs, they simply look for the motion vector which minimizes the matching error for the current block $\mathcal{B}_{\text{cur}}$. Yet, the matching criterion $\xi_{\text{W, SAD}}$ is modified to take into consideration the OBMC weighting window, and

$$\xi_{\text{w}} = \min_{\overrightarrow{d} \in \mathcal{A}} \sum_{(x,y) \in \mathbf{\Omega}_{\mathcal{B}_{\text{cur}}}} \left| \mathcal{B}_{\text{cur}}(x,y) - \widehat{\mathcal{B}_{\text{ref}}}(x+dx, y+dy) \right| \cdot W(x,y) \tag{2.27}$$

where $W(x,y)$ is the weighting window (see figure 2.2.4c).

c Estimation techniques which account for the blocks interdependency

In [OS94], Orchard and Sullivan introduce an estimation algorithm that takes into account the block overlapping features. The estimation is performed in two steps:

1. the motion field is initialized using Nogaki and Ohta algorithm [NO92]: each block is considered independently from the others;
2. each motion vector is then refined according to its neighbours; this refinement may be iterated. The matching criterion is here given by

$$\xi'_{w,SAD} = \min_{\overrightarrow{d} \in \mathcal{A}} \sum_{(x,y) \in \mathbf{\Omega}_{\mathcal{B}_{\text{cur}}}} \left| \mathcal{B}_{\text{cur}}(x,y) - \sum_{1 \leq i \leq \overline{N}} W_i(x,y) \cdot \widehat{\mathcal{B}_{\text{cur}}}(x+dx_i, y+dy_i) \right| \tag{2.28}$$

   where $\left\{ \overrightarrow{d_0}, \ldots, \overrightarrow{d_{\overline{N}}} \right\}$ are the motion vectors of $\mathcal{B}_{\text{cur}}$'s $\overline{N}$ nearest neighbours $\{\mathcal{B}_0, \ldots, \mathcal{B}_{\overline{N}}\}$, and $\{W_0, \ldots, W_{\overline{N}}\}$ their weighting windows.

Such algorithm guarantees that both estimation and compensation are in phase. In addition, it was proposed to optimize the weighting window along with the motion vectors. Again, this can be interpreted as optimizing the regularization term $p(dx, dy)$.

2.4.2 Control Grid Interpolation

In section 2.2.3.2, control grids were introduced as a way to provide a continuous representation of the motion field, at the expense of a poor representation of discontinuous areas of the motion field. In a block-based model, the influence zone of a motion vector is the block carried by this vector. In control grids, on the contrary, the control points influence zone is spread over all adjacent meshes; influence zones of neighbouring control points are thus overlapping. In the end, the displacement vector of a given control point cannot be estimated without taking into account its neighbours. Two families of algorithms have been dedicated to their estimation:

1. **local approaches** [SB91a, NH91] iteratively estimate the displacement of each control point one after another;
2. **global approaches** [Lec99, Cam04] optimize all the parameters of the control grid at the same time;
3. **semi-global approaches** [AT97a, AT97b, Lec99] consider successive meshes or local sets of meshes and globally optimize their parameters, thus adding a continuity/regularization constraint to the local approach.

In control grids, hierarchical motion estimation techniques have also been proposed [BTZ+99, ARAM03, MJZ06], and robustify the estimation. In any case, several iterations are performed over the whole image plane, until the parameters of the control grid converge to a solution which minimizes the DFD.

2.4.2.1 Local approaches

This iterative approach was first introduced in [SB91a] by Sullivan and Baker. Each control point is processed independently from each other. However, the error functional is minimized over the whole influence zone of the current node, which calls upon neighbouring control points. As a consequence, several iterations are successively performed, during

which the control points are displaced one after the other so that they minimize the error functional with respect to the current positions of neighbouring motion vectors. After several iterations, it is expected that the positions of the control points converge to an optimal solution.

At time instant t_{cur}, the position of the control points $\mathcal{C}(t_{cur})$ are initialized from their position at instant t_{ref}. Each control point $C_i(t_{cur}) \in \mathcal{C}(t_{cur})$ is provided with a flag, `local_flag`, which indicates if the node has reached its local optimum. The algorithm then carries out the following steps:

1. `local_flag` is set to `false` for all nodes.
2. for each node $C_i(t_{cur}) \in \mathcal{C}(t_{cur})$ whose tag `local_flag` equals `false`:
   (a) the algorithm searches for the optimal position of $C_i(t+1)$ within a local neighbourhood, with respect to the current positions of neighbouring control points;
   (b) if the position of $C_i(t_{cur})$ is unchanged and keeps the position estimated at previous estimation, $C_i(t_{cur})$'s tag `local_flag` is set to `true`
3. iterate step 2 until all nodes have converged, *i.e.* `local_flag` $=$ `true` $\forall C_i(t_{cur}) \in \mathcal{C}(t_{cur})$.

In order to transform the meshes according to the displacement of their control points, an interpolation function is required. In [SB91a], Sullivan and Baker perform two bilinear interpolations to obtain the horizontal and the vertical components of the motion vector. Interpolations may end up into sub-pixellic positions; in such cases, the pixellic intensity is interpolated from a third bilinear interpolation of neighbouring pixels. This algorithm has been baptised TBCGI, after the three successive bilinear interpolations.

2.4.2.2 Global approaches

In [Bru90], Brusewitz introduces a global approach to the estimation of the parameters of a deformable control grid $\mathcal{M}$. Inspired by the field of finite elements, global approaches estimate the displacement of all the control points at once. Let $\left\{\overrightarrow{d_{ref\to cur}}(C_0), \ldots, \overrightarrow{d_{ref\to cur}}(C_{N-1})\right\}$ be the displacement vectors they undergone between time instants t_{cur} and t_{ref}. Generally, they perform a gradient descent onto the parameters of the control grid, such that they minimize the DFD error functional ξ, now written as

$$\xi\left(\overrightarrow{d_{ref\to cur}}(C_0), \ldots, \overrightarrow{d_{ref\to cur}}(C_{N-1})\right) = \sum_{P\in\Omega_{cur}} \left[I_{cur}(P) - \widehat{I_{cur}}\left(P + \sum_{j=1}^{\overline{N}} \phi_j\left(P - C_j(t_{ref})\right) \cdot \overrightarrow{d_{ref\to cur}}(C_j)\right)\right] \tag{2.29}$$

where ϕ_j is the inner-mesh interpolation function,$\{C_0, \ldots, C_N\}$ the N control points of $\mathcal{M}$, and $\overline{N}$ the number of those whose influence zone contains point $P(x, y)$.

a A matrix formulation of the control grid parameters and its derivation

Typically, numerical optimization algorithms arranges the control grid parameter set into a $2 \times N$-dimensional vector $\Theta_{\mathcal{M}}$ which concatenates x and y parameters. Similarly, the derivative operator $\overrightarrow{\nabla_{\mathcal{M}}}$ with respect to these parameters is also a $2 \times N$-dimensional vector:

$$\overrightarrow{\Theta_{\mathcal{M}}} = \begin{bmatrix} dx_{ref\to cur}(C_0) \\ \vdots \\ dx_{ref\to cur}(C_N) \\ dy_{ref\to cur}(C_0) \\ \vdots \\ dy_{ref\to cur}(C_N) \end{bmatrix} \quad , \quad \overrightarrow{\nabla_{\mathcal{M}}} = \begin{bmatrix} \partial/\partial dx_{ref\to cur}(C_0) \\ \vdots \\ \partial/\partial dx_{ref\to cur}(C_N) \\ \partial/\partial dy_{ref\to cur}(C_0) \\ \vdots \\ \partial/\partial dy_{ref\to cur}(C_N) \end{bmatrix} \tag{2.30}$$

b First and second order control grid optimization algorithms

Numerous numerical optimization algorithms [BGLS06] can be used to minimize the error functional $\xi\left(\overrightarrow{\Theta_{\mathcal{M}}}\right)$. At each iteration n, they refine the parameters $\overrightarrow{\Theta_{\mathcal{M}}^{(n-1)}}$ output by the previous iteration $n-1$.

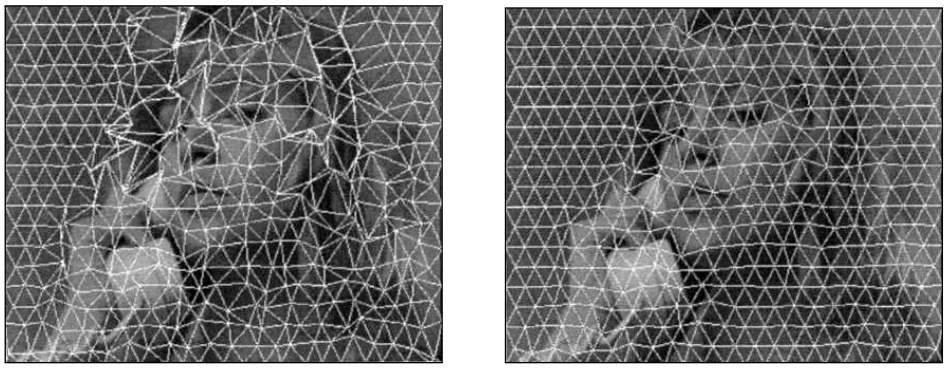
- **First order gradient descents** - $\xi\left(\overrightarrow{\Theta_{\mathcal{M}}^{(n)}}\right) = \xi\left(\overrightarrow{\Theta_{\mathcal{M}}^{(n-1)}}\right) - \lambda \cdot \overrightarrow{\nabla_{\mathcal{M}}}\xi\left(\overrightarrow{\Theta_{\mathcal{M}}^{(n-1)}}\right)$ - often approximate the functional ξ by its first order Taylor series Brusewitz. They are seldom if ever used as their convergence is very slow.
- **Second order gradient descents**, including Newton and quasi Newton techniques, converge much more fast. However, they suffer from the instability of the computation of the second derivatives.
- **The Gauss-Newton method**, finally, is as efficient as second order methods, but avoid the computation of the second order derivatives [LRS98, Lec99]. However, it is highly computationnaly demanding. At each iteration, it needs to invert the Jacobian, which, in this case, is a $2N \times 2N$ positive semi-definite, sparse, and diagonally dominant matrix. It can be performed through classical inversion tools, including Cholesky decomposition [Lec99], QR decomposition, conjugate gradient [Cam04], or Gauss-Seidel.

These approaches provide a better estimation of the control grid deformation in comparison to local approaches, which we reviewed in section 2.4.2.1. Yet, they may end up in solutions which do not respect the non-superposition constraint: some meshes may be overturned.

c A Levenberg-Marquardt regularization to limit meshes overturning

In order to avoid problematic situations like overturned meshes, it was propozed to regularize the estimation through a Levenberg-Marquardt (LM) augmentation. It is motivated by a simple observation: images whose spatial gradient is small will end up into a Jacobian matrix whose diagonal coefficient are small as well ($\ll 1$). Consequently, their inversion may generate large displacements, far from the true motion.

The LM augmentation is an efficient way to regularize an ill-conditioned matrix: it simply adds an offset to the eigenvalues of $\boldsymbol{A}$, to avoid small values. Even better results are obtained with adaptive LM augmentation, which adapts its orientation to the content of the images [Lec99]: it only slows down the deformation of the meshes in the direction whose spatial gradient is the smallest. Figure 2.4.3 shows the effect of the LM augmentation on the resulting deformable control grid.



(a) *Without LM augmentation* **(b)** *With LM augmentation*

Figure 2.4.3: *Effect of the LM augmentation on the estimation of a deformable control grid [Lec99]*

2.4.3 Hybridizing block-based and mesh-based representations

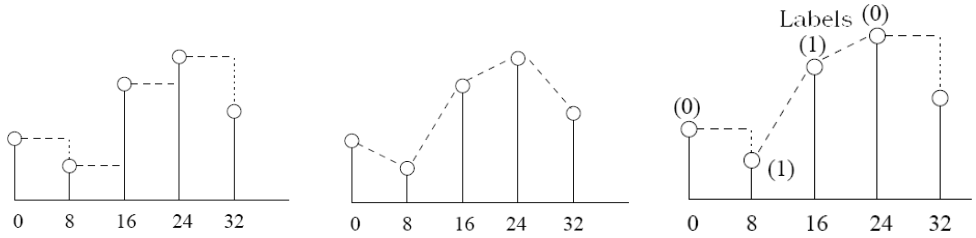
In the introduction of section 2.4, pros and cons of both blocks and meshes were briefly summarized. In particular, their respective abilities to model the motion field in a continuous or a discontinuous fashion was highlighted. Blocks, on one hand, provide a discontinuous representation of the motion, but are limited to the description of translations. However, overlapping the blocks through the OBMC does improve the initial block-based representation. Meshes, on the other hand, provide a continuous representation of the motion, and are able to handle various transformations. However, they can barely represent the ruptures of the motion field.

To overcome these limitations, it was naturally proposed to hybridize both blocks and meshes to benefit from all their respective advantages. Section 2.4.3.1 focuses on the Switched CGI (SCGI), an hybridization of the BMC and the CGI. Then, section 2.4.3.2 focuses on the Switched OBMC (SOBMC), an hybridization of the BMC and the OBMC. Finally, section 2.4.3.3 reviews even more advanced hybrid approaches.

2.4.3.1 Switched Control Grid Interpolation

Ishwar and Moulin introduced in in [IM97, IM00] a variant of the CGI. At first, the image is partitioned into blocks; each of these may then be motion compensated using either BMC or CGI. To this end, the motion vectors are set up at the corners of the blocks, which also play the role of control points. For each of these control points, a label `local_motion` controls the nature of the motion in the corresponding neighbourhood:

1. if $\texttt{local\_motion}(\mathcal{B}_i) = \texttt{false}$, the motion field is described through a translating block, similarly to BMC;
2. if $\texttt{local\_motion}(\mathcal{B}_i) = \texttt{true}$, the motion field is interpolated through CGI.



(a) *BMC* **(b)** *CGI* **(c)** *SCGI*

Figure 2.4.4: *Shapes of different interpolation functions used for motion vector interpolation [IM97]*

This hybrid motion model is illustrated in figure 2.4.4 in the one-dimensional case. As or the estimation algorithm, it is inspired from the local CGI algorithm [SB91a]: all controls point are successively optimized one by one. The optimal parameters of the current control point C_i are given by the pair $\left(\texttt{local\_motion}(C_i), \overrightarrow{d}(C_i)\right)$ that minimizes the error criterion on the reconstructed area. The whole process is iterated several time to ensure convergence.

2.4.3.2 Switched Overlapped Block Motion Compensation

Despite the significant reduction in blocking artefacts brought by the OBMC (see section 2.4.1.2), areas whose motion information quickly varies are blurred. To address this issue, Ishwar and Moulin propose in [IM00] an hybrid technique combining OBMC and BMC: the SOBMC. Similarly to the SCGI, a label `local_motion` controls, for each control point, the nature of the motion in the corresponding neighbourhood:

1. if $\texttt{local\_motion}(\mathcal{B}_i) = \texttt{false}$, the motion field is described through a translating block, similarly to BMC;
2. if $\texttt{local\_motion}(\mathcal{B}_i) = \texttt{true}$, the motion field is interpolated through OBMC.

As for its estimation algorithm, it is, again, inspired from the local CGI algorithm [SB91a]. The optimal parameters of the current block $\mathcal{B}_i$ are given by the pair $\left(\texttt{local\_motion}(\mathcal{B}_i), \overrightarrow{d}(\mathcal{B}_i)\right)$ that minimizes the error criterion on the reconstructed area.

The SOBMC can be seen as a simplified version of the SCGI. In average, though, Ishwar and Moulin show that SCGI provides a better motion field than the OBMC [IM00]. OBMC, however, generally provide a more compact motion information. In the end, SCGI remains worth of interest, since its excellent motion compensation makes up for the transmitted data increase. SOBMC may still be considered as it is less computationally demanding than the SCGI.

2.4.3.3 Advanced hybrid techniques

In further works, it has been proposed to combine OBMC and CGI. In [HMCP01], Helsing and Marpe hybridize those motions models in a similar manner than the SCGI [IM97]: a `local_motion` flag is sent to signal whether to use OBMC or CGI.

In [CHJ+06], Choi *et al.* free themselves from the supplementary labels which were transmitted along with the motion field in previous techniques. In areas whose activity is strong, the motion field is subsampled, by an interpolation function (*e.g.* bilinear), based on neighbouring motion vectors. This boils down to a CGI motion field interpolation. Finally, each sub-block carried by the interpolated motion vectors is then compensated using a local OBMC (see figure 2.4.5).

Choi *et al.* baptized their technique Irregular Grid OBMC (IG-OBMC). Figure 2.4.5 illustrates its principle in the one-dimensional case.

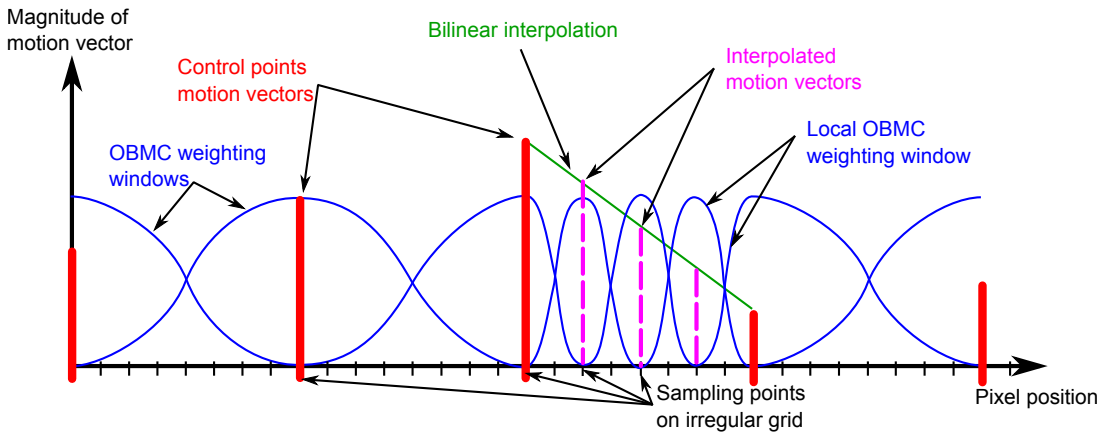


Figure 2.4.5: *IG-OBMC: Sub-sampling and interpolation of the motion field [CHJ+06]*

2.5 Conclusion

In video compression, the ability to motion compensate successive images of an image sequence is crucial. This chapter first described the nature of the motion information, from the real scene being shot, to the image plane on which motion is measured from the optical flow. In order to motion compensate a reference image I_{ref} into the current image I_{cur}, a motion model is used to parametrize the apparent motion. Then, a motion estimation technique is used to evaluate the parameters of the provided model.

From section 2.2, it is easily seen that a large number of motion models are now available, including:

- **dense motion fields**, which provide an exhaustive representation of the motion, and are generally dedicated to video analysis applications, wherein the amount of motion information is not limited;
- **parametric models** employ a smaller set of parameters, but are often limited to the estimation of the parallax component of the motion: they are not suited to the representation of local deformation;
- models based on a set of **deforming geometrical patterns**, which also represent the motion information from a rather small number of parameters. What is more, they are able to locally describe the motion information. For both these reasons, such models are naturally fit to the context of video compression. In particular, blocks and meshes, respectively reviewed in sections 2.2.3.1 and 2.2.3.2, are very popular.

The block-based translational model, on one hand, is particularly simple and still provides a valuable motion compensated prediction. In particular, its ability to represent the discontinuities of the motion field may be its key feature. However, it is less efficient in areas where the motion is continuously varying, and introduce blocking artefacts into the motion compensated prediction. Relying on overlapping blocks, the OBMC proved to be a crude but simple way to avoid these problems, at the expense of a loss in its ability to represent motion discontinuities. It has been extensively used, notably by standardized video coders such as MPEG-x [ISO93, ISO94b, ISO00a] and H.26x [ITU90, ITU94, ITU95, Ric03].

Meshes, on the other hand, are much more complex, but handle a large set of deformations. In addition, they naturally provide a continuous representation of the motion, both in space and time domains. Along the temporal axis, in particular, meshes can be used to exhibit the trajectory of an area: unlike blocks, the mesh-based motion model is not reset at each instant. However, meshes are not able to efficiently represent the ruptures of the motion field.

Naturally, hybrid solutions have been introduced to ally their abilities into switched models; they have been reviewed in section 2.4.3. The corresponding increase in complexity is fairly reasonable, and output motion information is still compact. These models appear to be particularly interesting in regards to our work: an idealistic model should be able to represent both continuities and discontinuities of the motion field, and handle as many deformations as possible, while providing a coherent motion information both in space and time.

Prior to the introduction of our contributions, next chapter will focus on existing video compression schemes. It will explain how decorrelation tools seen in chapter 1 and motion compensation are combined into practical compression schemes.

Chapter 3

Coding image sequences

Correlation, in image sequences, is found under many different forms; most of which were reviewed in chapter 1. Video compression relies on these correlations to extract and remove the redundancies hold by an image sequence, thus reducing the amount of information needed to be transmitted. Most decorrelation techniques introduced in chapter 1 inspired numerous coding designs, with more or less success. Since videos are a succession of images sampled at various instants, correlations are usually considered to be either spatial (within an image) or temporal (between several images). Corresponding terminologies peculiar to video coding are *intra* (spatial) and *inter* (temporal) coding techniques. As for images, they are often referred to as *frames*.

It was soon identified that motion compensation, was essential to an efficient decorrelation along the temporal axis. In other words, it allows the video coder to decorrelate motion and textural information, thus simplifying their processing. In particular, *inter* coding techniques mostly rely on the efficiency of the motion compensation. However, some specific coders such as the Motion JPEG (M-JPEG) and the M-JPEG-2000 (mainly used in Non-Linear Video Editing (NLVE) systems for television and movies post-production) do not use motion compensation, in order to provide a complete random access to the image sequence. However, this comes with a significant reduction in coding efficiency.

As motion plays a key role into our work, this chapter will focus on approaches using motion compensation. As different as they may be, they all consider the video information to be a set of basic spatial, temporal, or spatio-temporal units: pixels, blocks, regions, sub-bands, motion threads... While these units will drive the overall coding process, they may also hold a semantic information; for instance, video objects may be described through region units.

Section 3.1 will give a general overview on different approaches to video coding. Then, section 3.2 will focus on a few widespread coding tools, thus introducing section 3.3 which will give an overview of a state-of-the-art video compression standard: ITU-T H.264/AVC. Finally, section 3.4 will introduce alternative approaches including 3D Wavelet and Analysis-Synthesis (AS) coding schemes, whose disruptive technologies offer additional features and functionalities.

3.1 Introduction to video coders

3.1.1 Image sequences as input data

Spatial and temporal resolutions have a great influence on the quantity of information hold by an image sequence. The sampling rate is typically given as a number of images to be displayed per second, usually around 25 fps (frames per second), though greater sampling rates may also be used (30, 50, 60 fps). The spatial resolution, on the other hand, may greatly vary according to the targeted display. Table 3.1.1a lists several commonly used resolutions. In addition, a video can be sent either in a *progressive* (frame by frame) format, or in an *interlaced* (field by field) format.

The nature of the color information has also an influence on the amount of information needed to be transmitted. Though Red Green Blue (RGB) color space is best known, images are usually converted into the YUV (previously YCbCr) as it better matches the human color perception. Indeed, the human eye is more sensitive to luminance (Y) variations than chrominance (U/Cb - V/Cr) variations. Consequently, it has been proposed to subsample the chrominances, as can be seen in table 3.1.1b. Subsampling modes are denoted $J : a : b$, where J is the horizontal sampling reference (usually 4), a and b respectively indicate the number of chrominance samples kept on the first and the second line of a $J \times 2$ block of original samples.

Format	Width	×	Height	Aspect ratio
QCIF	176	×	144	11:9
CIF	352	×	288	11:9
4CIF	704	×	576	11:9
SD	768	×	576	4:3
VGA	640	×	480	4:3
HD 720	1280	×	720	16:9
HD 1080	1920	×	1080	16:9

(a) *Commonly used resolutions in displaying devices*

Mode	Subsampling		Bandwidth reduction
	Horizontal	Vertical	
4 : 4 : 4	none	none	none
4 : 2 : 2	↓ 2	none	33%
4 : 2 : 0	↓ 2	↓ 2	50%
4 : 1 : 1	↓ 4	↓ 4	62.5%

(b) *Several YUV sampling modes and corresponding subsampling rates*

Table 3.1.1: *Image sequences: common displaying formats and color modes*

While recent displays generally support High-Definition (HD) resolutions, classic displays usually stick to Standard Definition (SD) resolution. As for color information, most video streams are now using either 4 : 2 : 0 YUV mode for low resolutions, and 4 : 2 : 2 mode for higher resolutions requiring a high chroma quality.

3.1.2 Several approaches to video coding

As mentioned earlier, image sequences may be interpreted in various ways. Being a spatio-temporal set of pixels, they can be interpreted and processed as such. On the other hand, they generally result from a real or an artificial shooting process (as described in section 2.1.1 from previous chapter), and can be interpreted as the projection of a scene onto the camera plane, which itself can be in motion. Consequently, various approaches to video coding have been provided, and derive from an interpretation which lies in between a photometric and a semantic understanding. However, they all partition image sequences into basic units (photometric or semantic) which are often processed via similar techniques.

3.1.2.1 A generic approach to video coding

A typical video coding system can be represented as depicted in figure 3.1.1. Besides the *partitioning* step which is highly dependent on the chosen approach, any video coding scheme often performs a few common operations.

- The **motion estimation/compensation** step aims at aligning the video basic units along the temporal axis.
- The **transform** step aims at compacting the information hold by each of these units (transform is to be understood in a broad sense: either a mathematical transform or any compacting description technique).
- The **quantification** step aims at reducing the dynamic of the coefficients to transmit. As this is a non-reversible operation, it is a source for losses and distortions, but controls the output bit-rate.
- The **entropy coding** step further compacts the binary information according to its statistical properties.

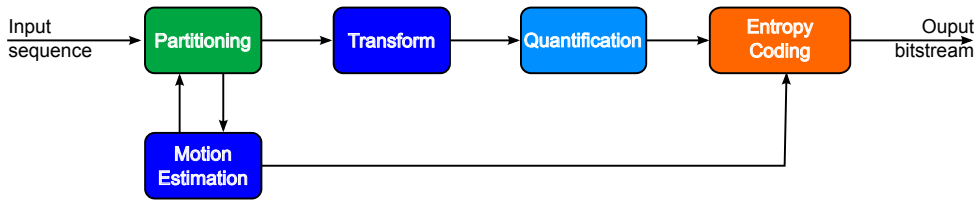


Figure 3.1.1: *A typical video coding system*

3.1.2.2 From photometric to semantic partitions of image sequences

a Photometric coders

Photometric video coding schemes consider image sequences to be a spatio-temporal set of pixels, without any further meaning. Consequently, the partitioning process is adapted to the chosen motion model and transforms, and may

partition each frame into blocks, regions [PL99], ... For instance, the classical approach to video coding partitions an image sequence into Groups Of Pictures (GOPs) and frames. Each frame is then further partitioned into blocks, whose dimensions may be fixed or variable. In any case, the partitioning process is only driven by *rate-distortion* constraints.

b Semantic coders

On the other hand, semantic video coding schemes consider image sequences to be the projection onto the camera plane of a set of objects in motion inside the scene. From then, image sequences are partitioned into objects whose position and shape is tracked across time. Typically, such schemes are known as object-based and model-based video coders [YW94, SGPK94, SMP+97, SM99, XLLZ01]. Some of them are provided as an overlay coding layer, and may be hybridized with a photometric coder (e.g. Motion Picture Expert Group (MPEG)-4 Part 2 [ISO00a]).

The objects being hold by one or several regions, the coder can process each of them individually. Information relative to both the background and the objects are then transmitted to the decoder which recomposes the scene from the received video elements. Both regions and/or 2D/3D models may be used to model the video objects. In practice, however, standardized video compression schemes do not consider this approach, because of two major drawbacks:

- model or shape parametrizations induce an overhead whose cost may be prohibitive, especially at low bitrates;
- both model-based and region-based coders often require the use of a segmentation process, whose computational cost generally prevents from getting real-time implementations.

c Pseudo-semantic coders

In between photometric and semantic approaches lie pseudo-semantic approaches. Though the coding process may still be focused on pixels (hence, photometric), these may be rearranged into semantic sets as it may ease their decorrelation. Indeed, being able to match successive pixels from an object should lead to a highly correlated sequence of pixels.

Like semantic approaches, pseudo-semantic video coding schemes generally do not guarantee a perfect reconstruction of image sequences. Their objective is to provide a reconstruction of the images which is visually acceptable. In particular, Analysis-Synthesis (AS) schemes (see section 3.4) are a typical example of pseudo-semantic coders. In addition, some region-based coders may also be classified into pseudo-semantic approaches; some of them partition the images into regions which may not correspond to video objects [PL99, DBBR07].

3.1.3 Quality assessment

As video information goes through capture, coding and transmission devices, it is subject to various distortions which tend to reduce the perceived video quality. In capture and transmission systems, these distortions result from a processing noise which cannot be controlled. In coding systems, however, lossy compression techniques often reduce the amount of information to be transmitted to such an extent that the quality of the videos is further reduced. Consequently, it is essential for video coders to be able to assess the quality of an image sequence. According to the availability of the original signal, quality metrics are generally classified into:

- **Full Reference (FR) methods:** the original signal is fully available and can be exhaustively compared to the distorted signal;
- **Reduced Reference (RR) methods:** only a set of features from the original signal is available: these features are compared with corresponding ones from the distorted signal [WJP+93];
- **No-Reference (NR) methods:** the original signal is not available at all, forcing the quality measurement to be performed blindly; in such case, it is essential to known which coding artefacts are liable to be found.

In video coding, original images are available, thus FR methods are most often used. Ideally, a good quality metric should match the user-perceived quality, *i.e.* the subjective quality. In subjective tests, users are asked to grade (usually from 0 to 10) the quality of a broadcast video. The Mean Opinion Score (MOS) is then given by the average score. Deblocking filters, though, can be interpreted as NR methods: they are able to blindly identify blocking effects.

A wide variety of objective metrics have been proposed to evaluate the quality. Mathematically convenient metrics such as the Signal-to-Noise Ratio (SNR), and especially the very popular Peak Signal to Noise Ratio (PSNR), ruled the field of quality measurements for a while. However, they proved to be relatively poor matches to subjective measurements. Later, pseudo-subjective metrics were provided as better matches to the subjective quality.

3.1.3.1 Objective metrics

SNR is a metric which measures the power ratio between a signal (the information) and the background noise. Alternatively, it corresponds to the squared ratio of the signal's Root Mean Square (RMS) amplitude to the noise's RMS amplitude. In images, a similar metric is known as the PSNR; it uses the image peak amplitude $\max(I)$ (usually 255 for 8-bits images) instead of the RMS amplitude. As for the amplitude of the noise, it is given by the MSE (see equation (2.20), in section 2.3.2 from previous chapter) between the original image I_{orig} and the distorted image I_{dist}.

$$\text{PSNR}_{\text{dB}} = 10 \cdot \log_{10} \left[\frac{\max(I)^2}{\xi_{\text{MSE}}(I_{\text{orig}}, I_{\text{dist}})} \right] \tag{3.1}$$

However, the Human Visual System (HVS) evaluates visual stimuli based on specific features (edges, textures, smooth regions, ...); consequently, the PSNR is not a good match to subjective scores.

3.1.3.2 Perceptual or pseudo-subjective metrics

In humans, the visual system is, among other things, in charge of assessing the quality of perceived images. Among the numerous psychophysic mechanisms of the HVS, a few are essential to visual perception:

- **a trichromate color perception:** due to the response spectra of the three types of cones -S(hort), M(edium), L(ong)- the eyes are more sensitive to certain wavelengths (*i.e.* colours) than others;
- **the contrast sensitivity** and the **visual masking effect** of the low-level HVS: they determine how sensitive we are to the various frequencies of visual stimuli;
- **the global precedence** of the mid-level HVS: the structural perception is done by integrating edges in a coarse-to-fine scale fashion.

Pseudo-subjective metrics were provided to take into account some of these mechanisms. They are generally compared to subjective evaluations through correlation indexes such as Spearman or Pearson rank correlation factors. The visual acuity, *i.e.* the contrast sensitivity and the masking effect, is generally modelled by a Contrast Sensitivity Function (CSF) [MS74]. Hence, derived metrics generally rely on frequency-based decompositions: [LK97], [Win99], Digital Video Quality (DVQ) [Wat98], JPEG2000's Weighted MSE (WMSE) [TM01], PSNR-HVS and PSNR-HVS-M [EJN+06].

Alternatively, metrics assessing overarching principles such as structural or information extraction have also been proposed: Universal Image Quality (UIQ) index [WB02], Structural Similarity Image Metric (SSIM) [WBSS04], Multi-Scale SSIM (MS-SSIM) [WSB03], Multiscale Modular Similarity (M^2S) and Multiscale Modular Maxima Similarity (M^3S) [ZZYX05], Three-Component PSNR (3-PSNR) and Three-Component SSIM (3-SSIM) [LB10]. Also, some metrics are combining both visual acuity and structural approaches: Threshold vs Contrast (TvC) [TH94], Visual SNR (VSNR) [CH07]. Finally, degradation-based approaches use a degradation model to represent the distortion and evaluate the difference between the model and the distorted image: Weighted SNR (WSNR) and Noise Quality Measure (NQM) [DVKG+00], Visual Information Fidelity (VIF) [SBdV05, SB06].

In image sequences, quality loss may also appear along the temporal axis. Some metrics have been provided to take this aspect into account, and are dedicated to video quality assessment: Perceptual Video Quality Measure (PVQM) [HBL+02], SSIM [WLB04], Video Quality Metric (VQM) [PW04], Perceptual Evaluation of Video Quality (PEVQ) [OPT08]. They may evaluate the video impairments, including: blurring, jerky motion, global noise, block distortion, chrominance distortion, ... Finally, *motion tubes* were also proposed to assess the video quality, and focus on the temporal stability of images [PBC07, Péc08].

Facing such a plethora of pseudo-subjective metrics, it is difficult to identify one of them as a universal metric. Indeed, none of them, so far, proved to be efficient at assessing all kinds of distortions. Consequently, normalization committees still advocate the combined use of PSNR and MOS, though VQM has also been seriously considered.

3.1.4 Coder functionalities

Besides compression abilities, video coders are often required to provide additional functions such as content description (*e.g.* MPEG-4 part 11), subtitling (*e.g.* MPEG-4 part 17), ... and notably scalability and multiple view coding. Both of them are considered to be critical in the development of new video coders. Scalability allows the exact same video stream to be played on a wide variety of terminals. Multiple view coding is essential to the next broadcast generation: Three-Dimensional TV (3DTV).

3.1.4.1 Multiview coding

With growing interest in 3DTV and Free viewpoint TV (FTV), the problem of multiview coding [HO07] has become crucial. Multiview videos are sequences which provide several views of the same scene for each time instant. In 3DTV, two views are required to generate a stereoscopic effect. In FTV, on the other hand, a larger number of views is required to offer to the end-user the possibility to navigate into the scene.

While some coders make do with a separate encoding of each view, thus providing as many video streams as there are angle of views, some more advanced coders use inter-view correlations to reduce the amount of information needed to be transmitted. Among the latter's, some of them may model the scene using a depth map [ZKU+04] or even a global 3D model along with its corresponding textures [GM01, EDM+08, CPML10].

3.1.4.2 Scalability

Throughout the years, more and more transmission channels and devices have been used to distribute and display video information. Nowadays, targeted applications may present strong variations in their abilities to receive, process and display this information. As a consequence, video content providers have to adapt to this heterogeneity. Obviously, it might become unrealistic to propose as many versions of the same video as there are targeted applications. Hence, scalability was proposed as a workaround, and provide a hierarchical video information, enabling each transmission channel and/or device to adapt transmission, reception, decoding or displaying to its maximum capabilities. As this information is hierarchical, the content is viewable whichever the amount of information kept.

Hierarchical coding techniques (*e.g.* the Wavelet transform) were proposed as ways to provide scalability [Sha93, SP96, Tau99]. In terms of generated bitstream, these coding techniques organize the information into *layers*. Typically, scalable bitstreams provide a *base layer* whose binary information can be transmitted, received, decoded and displayed by any of the targeted devices. In addition, a set of successive *enhancement layers* is available for devices of greater capabilities. These enhancements may improve the Quality of Experience (QoE) of the end user in several ways:

- **SNR or quality scalability** successively improves the quality of the received video stream in terms of SNR (typically, PSNR). Each enhancement layer thus provides residual information which reduces the distortion between received and original images;
- **Spatial or resolution scalability** addresses the problem of multiple-sized terminal displays. The base layer provides video information at a low resolution. Each enhancement layer provides additional information enabling the decoder to synthesize the video images at successive higher definitions;
- **Temporal scalability** provides a scalable stream whose temporal frequency adapts to the various refresh rates of terminal displays. Each enhancement layer increases the temporal frequency of the transmitted signal;
- **Motion scalability** provides a representation which successively refines the motion information with each enhancement layer [Cam04]; Indeed, motion information represents a large part of the overall video information
- **View scalability**, in multiview video compression systems, controls the number of views to be transmitted. Each enhancement layer may provide additional views to adapt to the terminal displaying capabilities;
- **Complexity scalability** is also provided as a way to adjust the decoding complexity to the available computational resources of the terminal.

Any of them may also be used to adjust the transmitted signal to the available bandwidth, thus providing a better Quality of Service (QoS). Indeed, transmitting the base layer only, for instance, requires a lot less bandwidth than transmitting all the enhancement layers.

3.2 Classic coding tools

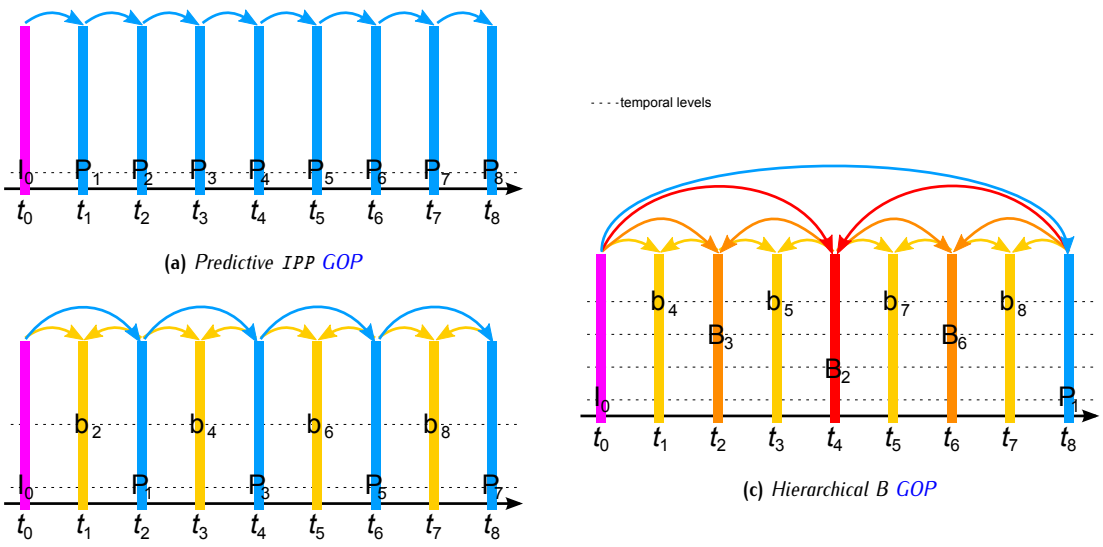
Video coding schemes greatly vary according to their nature and goals. However, some tools are now very popular and spread among a large number of video coders, including Groups Of Pictures (GOPs), *macroblocks* and Rate-Distortion Optimization (RDO).

3.2.1 Groups Of Pictures

Both *intra* and *inter* coding techniques can be used: coding a frame may require previously processed images:

1. **Intra-coded frames (I-frames)** are exclusively coded using spatial decorrelation techniques and do not depend on any other frame. Consequently, they can be decoded independently from any other frame, thus providing random access and limiting the error propagation of alternative coding schemes;
2. **Predicted-frames (P-frames)** are (partially) predicted from a previously coded/decoded frame, thus allowing the use of temporal decorrelation techniques (in particular, motion compensation);
3. **Bidirectional frames (B-frames)** are similar to P-frames, except that they are predicted from two previously coded/decoded frames.

As both P-frames and B-frames rely on inter coding techniques, they are more generally referred to as *inter-frames*. Since they can use both intra and inter coding techniques, they generally achieve further compression than I-frames. The reference frames from which they are predicted may be located both in the past and/or the future. This prevents both coder and decoder to process frames in the displaying order, but forces them to follow a *coding sequence*.



(a) *Predictive IPP GOP*

(b) *Hybrid IPB GOP*

(c) *Hierarchical B GOP*

Figure 3.2.1: *Several GOP structures*

In MPEG-1 and MPEG-2 video compression standards [ISO93, ISO94b], Groups Of Pictures (GOPs) start by an I-frame so that they can be randomly accessed. Their structure consists of a fixed sequence of P-frames and B-frames and their corresponding anchor instants. GOPs are often referred to by two numbers: M tells the distance between two anchor frames, and N tells the distance between two I-frames. For instance, the GOP structure $\mathtt{I}_0\mathtt{b}_2\mathtt{b}_3\,\mathtt{P}_1\mathtt{b}_5\mathtt{b}_6\,\mathtt{P}_4\mathtt{b}_8\mathtt{b}_9\,\mathtt{P}_7\mathtt{b}_{11}\mathtt{b}_{12}\,\mathtt{I}_{10}$ corresponds to $M = 3$ and $N = 12$. Subscript numbers denote coding order indices; uppercase letters denote anchor images and lowercase letters non-anchor images. Some GOP structures are widely used (figure 3.2.1):

- **predictive structures** $\mathtt{I}_0\mathtt{P}_1\mathtt{P}_2\mathtt{P}_3\mathtt{P}_4\ldots$, for which each P-frame is predicted from previous frame;
- **hybrid structures** $\mathtt{I}_0\mathtt{b}_2\mathtt{P}_1\mathtt{b}_4\mathtt{P}_3\ldots$ which incorporate B-frames into a predictive structure. B-frames are predicted from past and future neighbouring P-frames;

Later, H.264/AVC [JVT05] got rid of the notion of GOP and allowed for arbitrary sequences of I, P and B frames to be used, and even changed during the coding process. In particular, **dyadic hierarchical B structures** $\mathtt{I}_0\mathtt{b}_4\mathtt{B}_3\mathtt{b}_5\mathtt{B}_2\mathtt{b}_7\mathtt{B}_6\mathtt{b}_8\mathtt{P}_1\ldots$ rely on several temporal levels of B-frames. This partition of the temporal axis is equivalent to the one provided by the MCTF (see 1.3.3). To be exact, this approach is equivalent to the UMCTF.

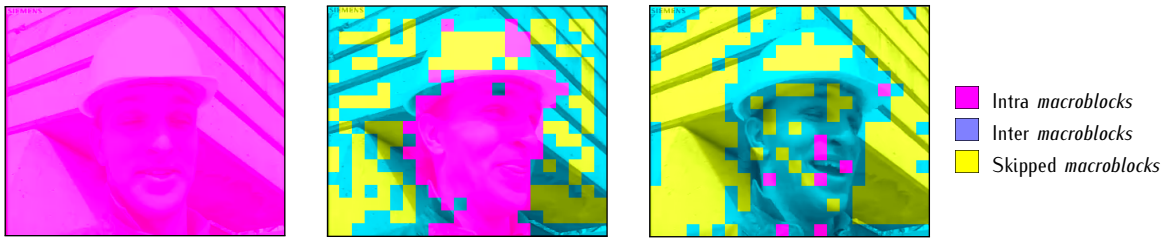
In practice, the length and the structure of the GOP mainly depend on the targeted application. Broadcast applications, for instance, require a key frame (an I-frame) at least every second or so. In addition, hardware or software memory limitations may force the coder to use a restricted set of anchor frames.

3.2.2 Block partitioning

From early video coders such as H.120 [ITU88], it was quickly realized that coding techniques should be adapted according to the local content of the sequence images. Indeed, areas whose temporal correlation is high should be encoded using inter-coding techniques, otherwise intra-coding techniques should be preferred. Hence, it was proposed to partition the images into regions whose coding process is adapted to the nature of their correlations. While a few coders rely on arbitrary shaped regions [SGPK94, SM99, RM04], most of them partition the images into blocks.

Standardized coders, since H.261 [ITU90], then MPEG-1&2 [ISO93, ISO94b] partitioned the images into fixed-size blocks (16×16 pixels), known as *macroblocks*. Later, H.264 [JVT03] introduced variable block size: overarching *macroblock*-size is still 16×16, but each *macroblock* may be further split into smaller *macroblocks* (16×8, 8×16, and 8×8 pixels). Similarly, the Locally Adaptive Resolution (LAR)-Video codec uses adaptive block size based on a quadtree decomposition [FBDC07].

Using blocks, it is then possible to locally adapt the coding technique to be used. Hence, in standardized coders, *macroblocks* are characterized by a coding mode (inter, intra, skip ... see 3.3.1.2 for further information on H.264 modes). Figure 3.2.2 shows the different modes used to code fixed-size *macroblocks* in a standardized coder for several images.



(a) *I-frame* **(b)** *P-frame* **(c)** *B-frame*

Figure 3.2.2: *Mode use on different types of frames on sequence* Foreman

3.2.3 Rate-Distortion Optimization

According to the targeted application, video coders may have to provide compressed image sequences at given quality (in practice, distortion) or bitrate. In any case, the coder should find the optimal trade-off between output *bitrate* $\boldsymbol{R}$ and provided signal *distortion* $\boldsymbol{D}$); this is known as Rate-Distortion Optimization (RDO). Let us assume that the targeted application requires the image sequence to be encoded at bitrate $\boldsymbol{R}_{\text{target}}$. In such case, the optimal rate-distortion tradeoff is obtained by minimizing $\boldsymbol{D}$ under the constraint $\boldsymbol{R} \approx \boldsymbol{R}_{\text{target}}$. As both $\boldsymbol{R}$ and $\boldsymbol{D}$ are functions of the encoding parameters Θ_c, the problem writes as:

$$\begin{aligned} \Theta_c^* &= \arg\min_{\Theta} \quad \boldsymbol{D}(\Theta_c) \\ &\text{subject to} \quad \boldsymbol{R}(\Theta_c) \approx \boldsymbol{R}_{\text{target}} \end{aligned} \tag{3.2}$$

where Θ_c^* are the coder optimal parameters for considered sequence and targeted bitrate. Many constrained optimization techniques are available and can be used to solve such a problem. Usually, however, Lagrangian optimization is used as it provides a relatively easy way to perform such optimization. Using Lagrange multipliers, both the distortion $\boldsymbol{D}(\Theta_c)$ and the bitrate $\boldsymbol{R}(\Theta_c)$ are combined into a Lagrange function $\boldsymbol{J}(\Theta, \lambda)$. The optimal encoding parameters Θ_c^* are now given by the following unconstrained problem:

$$\begin{aligned} \Theta_c^* &= \arg\min_{\Theta_c} \quad \boldsymbol{J}(\Theta_c, \lambda) \\ &= \arg\min_{\Theta_c} \quad \boldsymbol{D}(\Theta_c) + \lambda \boldsymbol{R}(\Theta_c) \\ &\text{such that} \quad \boldsymbol{R}(\Theta_c) \approx \boldsymbol{R}_{\text{target}} \end{aligned} \tag{3.3}$$

where $\lambda \geqslant 0$ is the Lagrange multiplier controlling the relative importance of $\boldsymbol{D}$ and $\boldsymbol{R}$ throughout the optimization process. In the rate-distortion space, the set of reachable $(\boldsymbol{R}, \boldsymbol{D})$ positions are located on the *operational curve* $\boldsymbol{D} = f_{\text{oc}}(\boldsymbol{R})$ (see figure 3.2.3). The optimal Lagrange multiplier λ^* is given by the slope of f_{oc} at the closest operational point $(\boldsymbol{R}, \boldsymbol{D})$ from targeted bitrate $\boldsymbol{R}_{\text{target}}$.

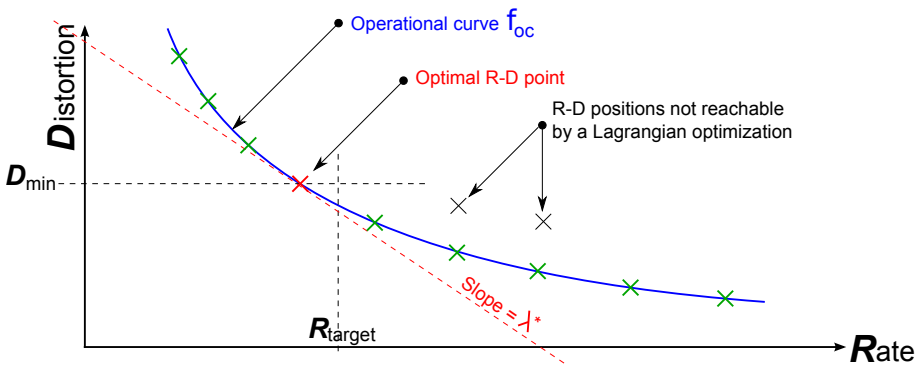


Figure 3.2.3: *Typical rate-distortion curve of a compression system*

3.2.3.1 Identifying optimal λ^*

The value of λ^* may not be *a priori* known; however, numerous works have been trying to provide models for the operational R-D curve. Several interpretations of this problem can be done [LT08], though source coding perspective is the most common one. In particular, a widely accepted model is given by [SW98]:

$$\boldsymbol{R} = f_{oc}^{-1}(\boldsymbol{D}) = \alpha \ln\left(\frac{\sigma^2}{\boldsymbol{D}}\right) \qquad \Longleftrightarrow \qquad \boldsymbol{D} = f_{oc}(\boldsymbol{R}) = \sigma^2 e^{-\boldsymbol{R}/\alpha} \tag{3.4}$$

As the derivative of f_{oc} function with respect to $\boldsymbol{D}$ is equal to the optimal Lagrange multiplier λ^*, the latter is given by

$$\lambda^* = \frac{\partial f_{oc}}{\partial \boldsymbol{R}} = \frac{\boldsymbol{D}}{\alpha}\,. \tag{3.5}$$

In standardized coders, it was shown that the distortion $\boldsymbol{D}$ is linked to the quantization step Q. In H.264/AVC, the quantization step is obtained from the input Quantization Parameter (QP). Finally, λ^* itself is expressed as a function of Q ($\lambda^* \propto Q^2$) [SW98].

3.2.3.2 RDO and block partitioning

As seen in 3.2.2, video compression systems often partition the images into blocks. Assuming that each block $\mathcal{B}_i$ is coded independently from the others, a local Lagrangian RDO can be individually performed on each of them. In each block $\mathcal{B}_i$, a local set of parameters $\Theta_{\mathcal{B}_i}$ is optimized such that they minimize the block distortion $\boldsymbol{D}_{\mathcal{B}_i}$, while allocating an appropriate bitrate $\boldsymbol{R}_{\mathcal{B}_i}$.

In standardized coders, the local RDO process selects the optimal coding mode and its associated parameters. Consequently, λ^* is often referred to as λ_{mode}. This just leaves $\boldsymbol{D}_{\mathcal{B}_i}$ and $\boldsymbol{R}_{\mathcal{B}_i}$ to be evaluated for each coding configuration to be tested. As $\boldsymbol{R}_{\mathcal{B}_i}$'s exact evaluation may be computationally demanding, relations between encoder parameters $\Theta_{\mathcal{B}_i}$ and required bitrate $\boldsymbol{R}_{\mathcal{B}_i}$ have been provided (*i.e.* the *ρ-domain* [HM01]), which spares the encoder from many computations. In the end, the overall distortion and bitrate are given by their sum over all blocks $\mathcal{B}_i$:

$$\begin{cases} \boldsymbol{D}(\Theta) = \sum\limits_i \boldsymbol{D}_{\mathcal{B}_i}(\Theta_i) \\ \boldsymbol{R}(\Theta) = \sum\limits_i \boldsymbol{R}_{\mathcal{B}_i}(\Theta_i) \end{cases} \tag{3.6}$$

3.2.3.3 Optimal motion estimation from a rate-distortion perspective

As motion compensation is a very efficient technique when it comes to decorrelate textures and motion, it still requires the motion information to be sent along with other transmitted data. In recent video compression systems, encoded motion parameters often represent the largest part of the information carried by the encoded bitstream. Hence, it is essential for the motion information to be as compact as possible. In consequence, it was proposed to apply the rate-distortion theory to motion estimation as well.

Motion estimation should find the optimal trade-off between the quality of motion prediction and its coding cost. In particular, it has been proposed to replace the distortion measure of matching criteria reviewed in section 2.3.2 by a rate-distortion Lagrangian [SB91b, Lee95, CKS96, SW98, CW98]. Considering a block-based motion model (see 2.2.3.1) using a single motion vector $\overrightarrow{d_{\mathcal{B}_i}}$ to represent the motion field of block $\mathcal{B}_i$, the Lagrangian matching criterion is given by

$$\begin{aligned}\zeta_{\text{RDO}}(\mathcal{R},\widehat{\mathcal{R}}) = &\quad \boldsymbol{D}_{\mathcal{B}_i} + \lambda_{\text{motion}}\boldsymbol{R}_{\mathcal{B}_i} \\ =&\zeta(\mathcal{B}_i,\widehat{\mathcal{B}}_i) + \lambda_{\text{motion}}\boldsymbol{R}(\overrightarrow{d_{\mathcal{B}_i}})\,,\end{aligned} \tag{3.7}$$

where $\widehat{\mathcal{B}}_i$ is the motion prediction of $\mathcal{B}_i$, and $\zeta(\mathcal{B}_i,\widehat{\mathcal{B}}_i)$ is the distortion measured between $\mathcal{B}_i$ and $\widehat{\mathcal{B}}_i$. Empirical value $\lambda_{\text{motion}} = \sqrt{\lambda_{\text{mode}}}$ has been advocated for a while. In [LT08], Tourapis *et al.* provide a Bayesian interpretation of the RDO, and show that previous empirical value results from several simplifications of a more complex model.

3.2.4 Bitstream generation: signalling flags and entropy coding

In the end, the different coding mechanisms output a large variety of information which need to be transmitted to the decoder. These information are protean, and (notably) include:

- image partitioning parameters (GOP structure, type of frame, block size, regions shape, ...)
- motion compensation parameters (motion predictors, motion vectors)
- transform parameters (pixel residues)
- quantization parameters (quantization step, rate-distortion Lagrangian)

All these information need to be arranged into a binary information and entropy coded to achieve further compression. From the different natures of the information, the statistical properties may greatly vary from one type of data to the other. As a consequence, the entropy coding shall consider each of these information individually. In H.264/AVC, for instance, several statistical contexts are used concurrently to match individual distributions of the different parameters.

To prevent from transmission errors, it is important for the encoded information to be error resilient. To this end, synchronizing markers can be inserted at specific locations of the resulting bitstream. May an error interfere with the decoding process, these markers can be used to check whether the decoding process needs to be interrupted, and started again from another location of the bitstream.

Finally, to each type of information corresponds an order of importance. The image partitioning parameters, for instance, are critical for the decoding process, which will not be able to decode anything if these information are not or badly transmitted. On the contrary, residual informations are less important: any wrong residual information will simply end up in unwanted distortions within the reconstructed images. The overall decoding process, however, is not affected from this malfunction. As a consequence, some information need to be more securely transmitted than others.

3.3 A classic approach to video compression: ITU-T H.264/AVC

Previous sections provided an overview on video compression systems, and focused on a few concepts commonly used in past and modern video coders. Yet, how are they used in practice? This will be explained through a general overlook on last standardized video compression standard: ITU-T H.264/AVC. For further information on H.264/AVC, one may refer to the plethora of publications discussing the standard [Ric03, WS03], and of course the official norm [JVT05, JVT07a, JVT07b, JVT09].

H.264/AVC has been designed by the Joint Video Team (JVT), a group of video coding experts from the International Telecommunication Union (ITU), the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). As both ITU and ISO organizations were engaged in H.264/AVC development, the latter is also known as ISO/IEC 14496-10 MPEG-4 Part 10 Advanced Video Coding (AVC). The JVT was created in 2001 and brings together the:

- **Video Coding Expert Group (VCEG):** ITU Telecommunication Standardization Sector (ITU-T) Study Group 16;
- **Motion Picture Expert Group (MPEG):** ISO/IEC Joint Technical Committee (JTC) 1, Subcommittee (SC) 29, Working Group (WG) 11.

As a result of this collaboration, the compression standard H.264/AVC was first issued in 2003 and provides a detailed description of a *macroblock*-based hybrid photometric decoder. Hence, it performs block motion compensation, block transform, quantization and entropy coding to compact an image sequence (figure 3.3.1). Its compression capabilities are more or less about twice as efficient as those of MPEG-2. It has been designed for a large set of technical applications, including broadcast (over cable, satellite, DSL, terrestrial, ...), storage (Digital Versatile Discs (DVDs), ...), video-conferencing, VOD and multimedia streaming. Its design decouples the compression process from the transmission process, and it is split into two main layers:

1. **the Video Coding Layer (VCL)** is in charge of compacting the image sequence using decorrelation techniques introduced in chapters 1 and 2, along with appropriate binarization and entropy coding;
2. **the Network Abstraction Layer (NAL)** [WS03] facilitates the transmission of H.264/AVC compressed data on various transport layers -Real Time Protocol (RTP), IP-, MPEG-4 file format MP4, MPEG-2 broadcasting protocol, ...) by packetizing the video information into small units of information known as *NAL units*.

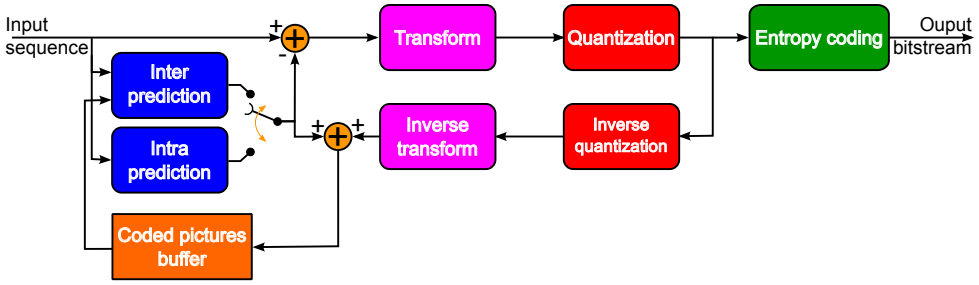


Figure 3.3.1: *H.264/AVC's overall closed-loop scheme*

A *NAL unit* is a packet which contains an integer number of bytes, including a header indicating the nature of the carried information, and a payload which contains the actual information. They may carry two types of information: *VCL units* contain compressed video information, while *non-VCL units* provide side information, such as *parameter sets*.

Decoding parameters are not likely to change very often throughout the sequence. Consequently, *parameter sets* provide control information relative to a large number of VCL units: *sequence parameter sets* apply to a series of consecutive coded pictures, while *picture parameter sets* only apply to the decoding of a single picture.

As several NAL units may be needed to transmit a single image (a *primary coded picture*), they are grouped into *access units*. These are eventually prefixed with an *access unit delimiter* and may also carry additional data such as picture timing information. In addition, the NAL mostly contributes to the error resilience abilities of H.264/AVC compression standard [Wen03, SHW03].

3.3.1 Data partitioning and coding modes

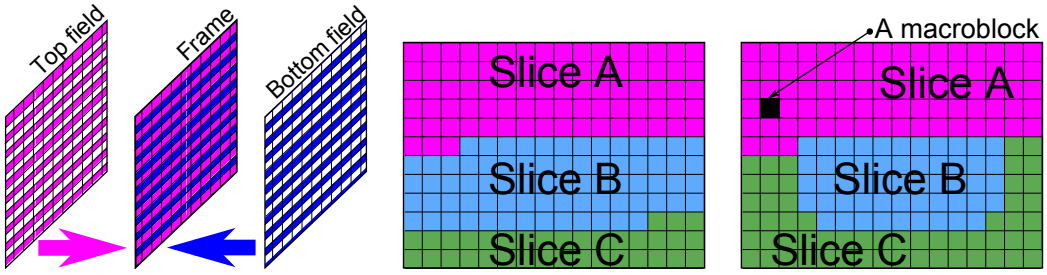
In order to adapt the coding technique to the content of the video, H.264/AVC standard partitions the frames into specific areas: slices and *macroblocks*. According to the type of frame (I-frame, P-frame, B-frame), intra or inter coding techniques may be used in each of these areas. Section 3.3.1.1 will discuss H.264/AVC's partitioning process, and section 3.3.1.2 will discuss the available coding modes.

3.3.1.1 GOPs, frames, fields, slices and *macroblocks*

Though H.264/AVC does not split image sequences into fixed GOPs, but may use ever-changing GOP structures, it still encodes the sequence on a frame-by-frame basis. Both progressive and interlaced frames can be handled through *fields*. Images are split into a *top-field* and a *bottom-field*; the first contains even-numbered rows, while the second contains odd-numbered rows (see figure 3.3.2a). A frame is said to be progressive when both fields are sampled at the same time instant, interlaced when they are not.

In any case, each frame is partitioned into 16×16 *macroblocks*. Each of them are coded using available intra or inter modes. While the overarching *macroblock*-size is fixed, each *macroblock* may be further split (depending on the coding mode) into 16×8, 8×16 or 8×8 sub-blocks, whenever this may increase the compression efficiency.

Slices have been provided as a way to process independently a given set of *macroblocks* from other *macroblocks*. Consequently, a *slice* is a sequence of *macroblocks* (see figure 3.3.2b); using Flexible Macroblock Ordering (FMO), an image may be split into arbitrary-shaped slices (see figure 3.3.2c). This may allow the coder/decoder to independently encode/decode each of these slices. Region Of Interests (ROIs), for instance, can be handled by a set of appropriate slices. In addition, slices may be grouped into *slice groups*, thus providing a hierarchical partitioning of the image.



(a) *Top and bottom fields* **(b)** *Slice partitioning* **(c)** *Arbitrary-shaped slice*

Figure 3.3.2: *Fields, slices and* macroblocks

3.3.1.2 Intra and inter coding modes

Depending on the type of frame (I-frame, P-frame, B-frame), a set of coding modes is evaluated for each *macroblock*. In I-frames, only intra coding modes are enabled; while P-frames and B-frames also enable inter coding modes. In particular, B-frames may use a combination of two motion-compensated predictions. To remove the blocking effects at the *macroblocks* boundaries, a loop-filter is used to smooth the images in those areas.

a Intra-frame prediction

In I-frames, macroblock can be compressed using one of the following modes:

1. **Intra 4×4:** the *macroblock* is split into sixteen independent 4×4 sub-blocks. Each of them are spatially predicted, then transformed using an IntDCT (see 1.2.2.1), and finally quantized.
2. **Intra 8×8:** the *macroblock* is split into four 8×8 sub-blocks. Each of them are predicted, transformed using an 8 × 8 IntDCT (again, see 1.2.2.1) and quantized independently from each other. This mode is available since H.264/AVC's FREXT amendment [JVT05].
3. **Intra 16×16** mode is very similar to *intra 4×4* mode. The prediction pattern, however, is applied to the whole *macroblock*. What is more, resulting DC coefficients are further transformed using another 4×4 IntDCT transform.
4. **Intra Pulse Code Modulation (PCM)** mode bypasses the prediction and transform steps, providing a precise way to encode the *macroblock*, and sets a hard limit to the bit-rate which can be allocated to a *macroblock*.

The spatial prediction is based on previously decoded *macroblocks*. Directional predictions propagate neighbouring luma and chroma samples along the chosen direction in the current *macroblock*. While 8 directions are available for the *intra 4×4* mode, only vertical and horizontal directions are available in the *intra 16×16* mode. Finally, a DC prediction uses the average of adjacent samples and uniformly predict the current *macroblock* with this single value.

b Inter-frame prediction

Besides intra coding modes previously introduced, motion compensation and associated inter coding modes may also be used to code *macroblocks* from P-frames and B-frames. Using inter coding modes, *macroblocks* may be further partitioned, down to 4 × 8, 8 × 4 and 4 × 4 luma samples.

BMC (see 2.4.1) is then performed on each of the sub-blocks, requiring up to 16 motion vectors (for P-frames) and 32 motion vectors (for B-frames) to be transmitted for a single *macroblock*. The motion compensation is accurate and outputs motion vectors in units of one quarter of the distance between two luma samples. They are differentially coded from a median prediction of neighbouring *macroblocks*. Finally, the difference between the motion compensated prediction and the original *macroblock* is transformed, quantized, and transmitted along with the motion vector.

As inter prediction requires both the coder and the decoder to synchronize their lists of reference pictures, Memory Management Operation (MMO) commands are used to control the content of two reference lists: *list 0* and *list 1*. In P-frames, *macroblocks* are motion compensated from a single reference texture: only *list 0* is used as a source of reference images. In B-frames, *macroblocks* are motion compensated from two reference textures: the first one is sourced from an image of *list 0*, while the second is sourced from an image of *list 1*.

In addition, (resp.) *skip* and *direct* modes are provided for (resp.) P-frames and B-frames. In such cases, motion vectors are simply predicted from neighbouring *macroblocks*; no residual information is sent. Hence, both *skip* and *direct* modes provide very compact representations of a *macroblock*.

3.3.2 Entropy coding

Once video information has been compacted using either intra or inter coding techniques, it is further compressed via entropy coding. Video information is first binarized into a stream of symbols; in *macroblocks*, for instance, transform and residual coefficients are ordered following a zig-zag scan. Either way, resulting bitstreams are entropy coded according to the statistics of their symbols. While most syntax elements are coded by an exponential Golomb-Rice dictionary [Gol66], residual and transform coefficients are coded using context-adaptive dictionaries: either Context Adaptive Variable Length Coding (CAVLC) or Context Adaptive Binary Arithmetic Coding (CABAC).

3.3.2.1 Context Adaptive Variable Length Coding

CAVLC is a relatively simple entropy coding technique whose use is suitable when a limited complexity is required. As the zig-zag ordered output of the transform/quantization step generally ends by trailing ones and sequences of zeros, CAVLC has been optimized for such signals.

A codeword table is used to map symbols into a compact low-entropy binary signal. A set of customized codeword tables is available, and adaptivity to the context is provided by a switching mechanism which selects the best codeword table according to previously processed symbols.

3.3.2.2 Context Adaptive Binary Arithmetic Coding

Alternatively, CABAC [MHW03] relies on a more complex approach and usually leads to 10 to 20% in bitrate saving, compared to CAVLC. The information is first binarized according to a codeword table (*e.g.* exponential Golomb-Rice). It is then further compressed by a binary arithmetic coder [Jel68, Ris76, RLJ79, LJ84].

The coding process adapts to the signal statistics by selecting one out of 128 context models provided along with the CABAC. These context models consist of a set of probability states. A state machine is in charge of updating the probability state of the current context according to previously processed bits. Furthermore, as each type of information may be distributed following specific statistics, several contexts are concurrently used by the CABAC, which switches the context according to the type of information to encode.

3.3.3 Extensions

Among amendments to H.264/AVC compression standards, scalable video coding capabilities and multiview coding capabilities have been provided. In 2007, amendment 3 provided the Scalable Video Coding (SVC) extension as annex G [JVT07b]. More recently, amendment 4 provided an Multiview Video Coding (MVC) extension as annex H [JVT09].

3.3.3.1 Annex G: SVC

In SVC extension [JVT07b, SMW07], three types of scalability are provided: spatial, temporal and quality scalabilities (see 3.1.4.2). They may be combined and used altogether (see figure 3.3.3): each enhancement layer, for instance, may then provide both higher resolution and better quality.

In any case, the scalable structure leads to a reduced compression efficiency, compared to a standard single-layered compression. However, inter-layer prediction mechanisms keep the loss of performance fairly reasonable. In particular, they predict the enhancement information of the current layer from previously decoded layers; this includes motion vector prediction, and residue prediction.

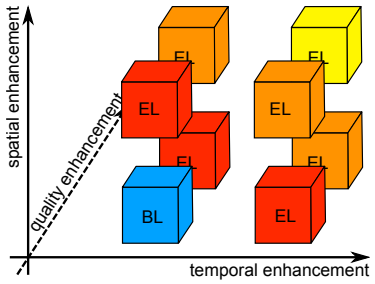


Figure 3.3.3: *SVC and combined scalability: base layer (BL) and enhancement layers (EL)*

a Spatial scalability

SVC provides two types of spatial scalability: *dyadic* and *extended*, each of which providing a pyramid of increasing resolutions, from the base resolution of the base layer to the maximum resolution of the last enhancement layer. With dyadic scalability, each enhancement layer multiplies both width and height of the previous layer's resolution by two.

A more flexible approach is provided through the Extended Spatial Scalability (ESS) which enables arbitrary ratios between successive layers resolutions. Alternatively, it is also capable of successive enlargements of the image support provided by the base layer. Either way, the spatial scalability introduces at least 10% of bit-rate increase compared to classic single-layered H.264/AVC performances.

b Temporal scalability

Temporal scalability is provided via the use of hierarchical GOP structures. As seen in section 3.2.1, dyadic hierarchical B GOPs (figure 3.2.1c) exhibit several temporal layers. It was naturally proposed to split the bitstream such that the base layer and the enhancement layers carry, each of them, the information relative to a given temporal layer. Consequently, temporal scalability can be achieved without any loss in terms of rate-distortion efficiency.

c Quality scalability

With quality scalability, each enhancement layer increases the quality of the decoded sequence. SVC provides a relatively flexible control on the increment in quality, through three different approaches:

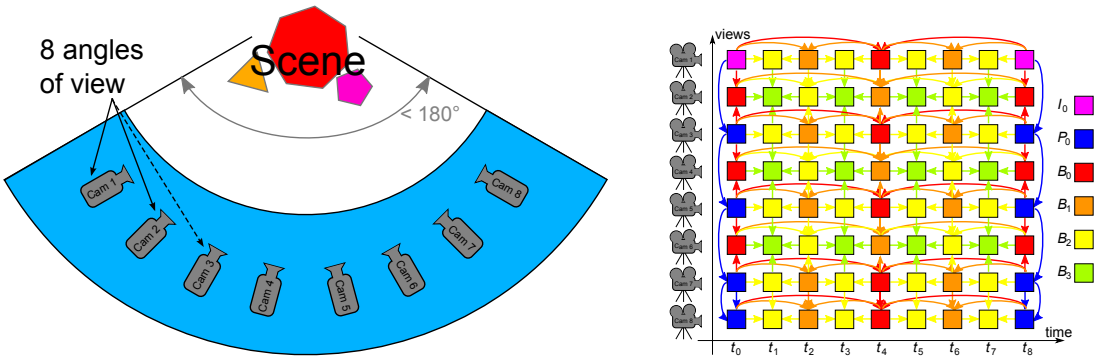
1. **Coarse Grain Scalability (CGS)** refines the quality by the transmission of successive packets. Using such an approach, the number of quality levels is fixed by the encoder;
2. **Medium Grain Scalability (MGS)** lies in between CGS and Fine Grain Scalability (FGS) and provides a finer control on the quality than CGS, while keeping the complexity reasonable.

This type of scalability is especially suited for simulcast applications. When providing a scalable bitstream whose overall bitrate reaches up to twice or three times the bitrate of the base layer, rate-distortion performances are typically within 10% in regards to a single layered bitstream.

3.3.3.2 Annex H: MVC

The MVC extension [JVT09] applies H.264/AVC classic temporal decorrelation and coding techniques to the inter-view dimension. Consequently, images corresponding to successive views of the same time instant are arranged into a GOP structure as can be done along the temporal dimension (figure 3.3.4). A view may then be predicted from both neighbouring views of the same instant and neighbouring instants of the same view.

Compared to simulcast (transmission of as many bitstreams as there are views), MVC is 20 to 30% more efficient for stereoscopic sequences. It is even more efficient with increasing number of views, up to 50% of gain when the number of views is greater then eight. With the recent arrival of High Definition Multimedia Interface (HDMI) 1.4 norm, providing a displaying interface for stereo video signals, MVC has recently been made available in commercial applications (*e.g.* Blu-Ray Disc).



(a) *8-view shooting system*

(b) *MVC: GOP structures along inter-view and temporal dimensions*

Figure 3.3.4: *MVC's approach to FTV*

3.3.4 Profiles

As can be seen from previous section, H.264/AVC provides a large set of coding tools, most of which can be activated or deactivated if required. In order to facilitate their choices, coding *profiles* have been provided and standardize the use of H.264/AVC compression norm:

- **the Baseline Profile (BP):** the simplest profile, it targets applications of limited computational power. In particular, B-frames and CABAC are disabled;
- **the Main Profile (MP):** enables nearly all H.264/AVC's coding tools, with the exception of slice partitioning;
- **the Extended Profile (XP):** provides additional error resilience capabilities and enables all coding tools;
- **the High Profiles (HiP):** they are used for high definition applications -HD Television (HDTV), High Definition DVD (HD-DVD), Blu-Ray, DVB- and support high definitions and 10 bits pixel representation.
- **SVC profiles:** (resp.) scalable baseline, high and high intra profiles respectively target different types of applications: (resp). conversational and mobile applications, streaming/storage and video-conferencing applications, and finally professional applications.
- **MVC profiles:** stereo high profile and multi-view high profile.

3.4 Disruptive approaches to video coding

Besides standardized video coders such as H.264/AVC and its predecessors, various alternative approaches to video coding have been proposed. Ten years ago or so, with growing popularity of the Wavelet transform, the latter was applied along both spatial and temporal dimensions, thus requiring the coder to efficiently decorrelate the motion information from the textural information. Later, it was proposed to adapt the representation of the texture to the nature of the motion, which led to Analysis-Synthesis (AS) video coding schemes. Even more recently, with last progresses in the field of texture synthesis and inpainting, it was also proposed to use these as ways to partially synthesize videos.

3.4.1 Hybrid coding based on 3D-Wavelets and MCTF

As mentioned in section 2.1.5, the motion in an image sequence may be interpreted as a set of trajectories, or *threads*, across time and space. Consequently, alternative video coders based on a transform along these threads were proposed. As can be seen from section 1.3.3, MCTF is a straightforward candidate to the compression of image sequences whose motion trajectories are known. Such open-loop coding schemes (see figure 3.4.1) completely decorrelate the motion information frlom the textural information.

In particular, $t + 2D$ transforms have been relatively popular: a temporal Wavelet is first applied along the motion trajectories, then each temporal subband (*i.e.* filtered images) is then further decomposed with a classic 2D Wavelet. Another interest of such representations lies in the multi-resolution features of the Wavelet transform which naturally provides a scalable framework.

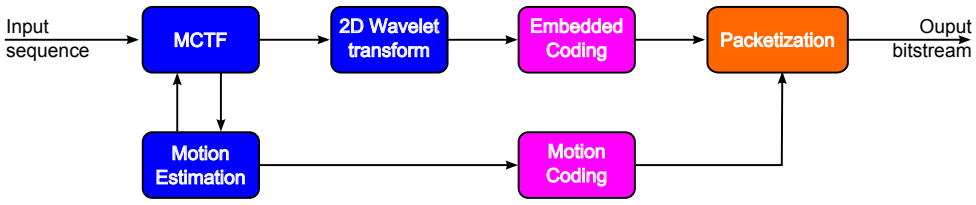


Figure 3.4.1: *Overall open-loop scheme of a video coder based on MCTF and 3D Wavelets*

3.4.1.1 Towards a continuous processing of the motion

Applying a Wavelet transform along a motion trajectory brings several issues to light. In particular, the fact that the motion field is not bijective is a source for serious problems. Assuming that the motion estimation is performed between each image I_{cur} and its predecessor, any pixel $\mathcal{P}$ from the latter may be:

- **single-connected**: it has a single colocated pixel in I_{cur}. This is the most convenient case, as $\mathcal{P}$ belongs to a single motion thread on which the temporal Wavelet transform may be applied;
- **multi-connected**: it has several colocated pixels in I_{cur}. In this case, $\mathcal{P}$ belongs to several motion threads, which is a source for redundancies;
- **non-connected**: it has no colocated pixel in I_{cur}, thus does not belong to any motion thread, though its incorporation into the temporal Wavelet is necessary.

In addition, the problem is even more complex when performing a sub-pixellic motion compensation. Indeed, $\mathcal{P}$ may be predicted by a weighted combination of several reference pixels, such that these reference pixels belong to the same motion trajectory. Above all, the lack of continuity in the processing of the motion prevents from performing an efficient temporal decorrelation.

First attempts at 3D Wavelet coding of image sequences were proposed by Ohm [Ohm94] and Choi [CW99]. As both implementations were prior to practical lifting-based implementations of the Wavelet transform [Swe96], they use Quadrature Mirror Filters (QMFs) to perform the 3D decomposition. In [Ohm94], Ohm decomposes the 2D temporal subbands using a TDAC decomposition [PJB87, Ohm93], then applies an adaptive lattice Vector Quantization (VQ) as an entropy coder. In [CW99], the temporal decomposition is done via a Haar Wavelet, while a separable 9/7 Daubechies Wavelet is used to perform the 2D decomposition; entropy coding is done by a 3D extension of a Subband-Finite State Scalar Quantizer (SB-FSSQ) [NW96]. Both schemes employ a hierarchical VSBMC to motion compensate the images.

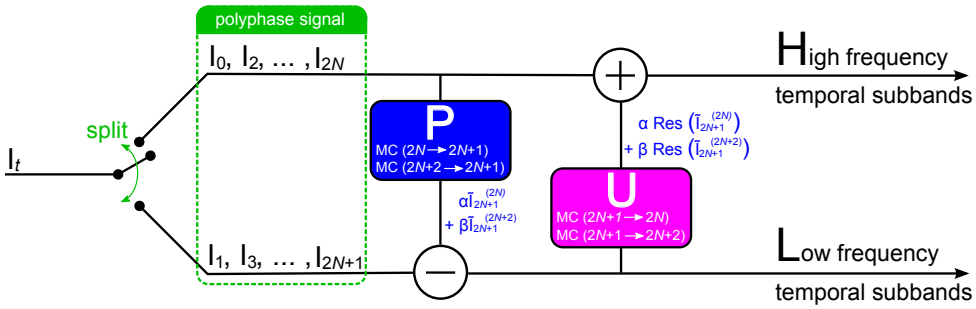


Figure 3.4.2: *Lifting based MCTF*

With the introduction of the lifting scheme into 3D Wavelet video coding [PPB01, ST01, LLL+01], several of the previously mentioned issues were solved. The temporal lifting splits the image sequence into a polyphase signal (odd and even frames, see figure 3.4.2):

- each odd frame is first **predicted** from motion compensated neighbouring even frames,
- then each even frame is **updated** using the prediction residues of neighbouring odd frames.

In [ST01], non-connected pixels are avoided by the use of bi-directional motion fields; however, this increases the side-information needed to be transmitted as twice as much motion information is required than in previous schemes. In [PPB01], the scheme initially proposed in [CW99] is modified into a lifting implementation; what is more, multi-connected pixels are adaptively connected to their best colocated predecessor.

3.4.1.2 Motion threads and Barbell lifting

In [XXLZ01], Xu *et al.* introduce the 3D-Embedded Subband Coding with Optimized Truncation (ESCOT) coding scheme, as a 3D extension of JPEG-2000 Embedded Block Coding with Optimal Truncation (EBCOT) coder. As first 3D Wavelet coding schemes failed at processing and coding the motion efficiently enough, Xu *et al.* used *motion threads* to a continuously process the motion across several time instants. To be more precise, a *motion thread* is a motion trajectory which starts and end with objects appearance and disappearance; they are obtained from the computed motion field. Using such an approach, multi-connected and non-connected pixels are easily handled (see figure 3.4.3):

- **multi-connected pixels** are pixels on which several motion threads are merging; in such case, a single motion thread is continued, the remaining being terminated;
- **non-connected pixels** are pixels which do not belong to any motion thread, and correspond to the beginning of new motion threads.

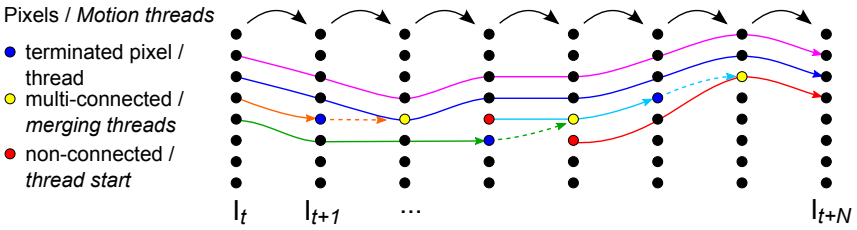


Figure 3.4.3: *Several* motion threads *running through a GOP*

However, motion-threads require motion fields to be pixel-accurate, thus preventing the coding scheme the advantages of sub-pixellic motion compensation. In addition, as motion threads cannot be merged, the number of terminating or starting motion thread may dramatically increase with complex motions. The latter phenomena introduce boundary effects which penalize the coding efficiency.

In [LLL+01, LFWLZ03], motion threads are incorporated into a lifting scheme, and further improved by enabling pixels which were previously terminated to be predicted by both backward and forward matching pixels. As a consequence, motion threads can be merged, thus reducing boundary effects; furthermore, as the lifting scheme guarantees a perfect reconstruction, it is now possible to perform sub-pixel motion compensation. Later, *Barbell lifting* [XWX+04] was provided as a way to predict and update multi-connected pixels using a weighted sum of all its matches. Finally, as was mentioned in section 1.3.3, improved performances may be obtained via UMCTF which deactivates the lifting update step [TvdS02, VGP02, TvdSA+05, Pau06, And07].

3.4.2 Analysis-Synthesis approaches

As seen in previous section, 3D Wavelet coding schemes rely on the motion field to perform the temporal filtering. In other words, the transform is adapted to the temporal *geometry* of the image sequence. However, both motion and textures are intrinsically related, and a losing motion information may have critical repercussions on the textural content of decoded images.

Analysis-Synthesis (AS) approaches break the interdependency between motion and textures. Analysis refers to pre-processing operations which aims at building a compact representation of a signal, while synthesis refers to post-processing operation which reconstructs decoded images. Several AS approaches have been provided in recent years:

- techniques using **global motion compensation**: they project all the images of a GOP onto a single reference instant, thus providing a set of reference textures;
- techniques using **analysis-synthesis of textures** or **inpainting**: textures are generated using dedicated algorithms;
- techniques using **3D models**: they compute and transmit a 3D model of the scene, along with corresponding textures [GM01, GM02, BGM06]; such an approach may be used for multiview coding schemes. They will not be further discussed in this document.

3.4.2.1 Global motion compensation

Coding schemes performing a global motion compensation projecting each image of the GOP into a single system of coordinates (figure 3.4.4). In other words, such an approach generates a motionless GOP whose temporal correlation is obvious and can be easily used to provide a compact representation of the texture across the GOP. As this corresponds to a $2D + t$ approach, there is no need to adapt the decorrelation technique to the motion.

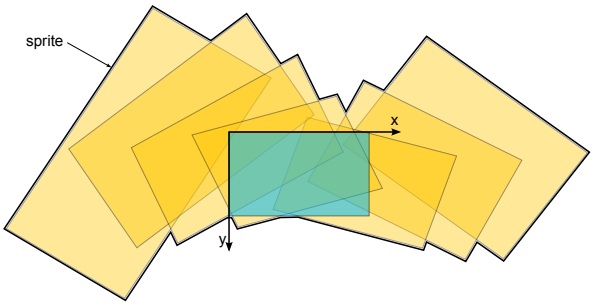


Figure 3.4.4: *Global motion compensation of several images into a single coordinates system and corresponding sprite*

a A generic approach to video coding

In [TZ94], GOPs are motion compensated by BMC, then encoded and transmitted independently from the estimated motion. As translational motion did not prove to be efficient enough to perform a global compensation across a whole GOP, it was proposed in [WXCM99] to use a control grid. Due to scaling motions (zooms, objects getting closer or farther, ...), projecting all the images onto the same reference instant may lead to resolution losses. Consequently, it was proposed, still in [WXCM99], to choose the instant of projection so that it minimizes such effects.

In [Cam04], a $t + 2D$ alternative to global motion compensation is provided. In each GOP, first and last images are used as key images for the base layer, and are encoded using EBCOT. Remaining images, respectively GOP's first half and second half, are projected on first and last key images using active meshes. Each motion compensated image is also linearly interpolated from both key images, and prediction residues are further decomposed into spatio-temporal subbands. Compared to a classic 3D Wavelet coder, this technique requires a lot less motion information to be transmitted. In [LG08], finally, the coding scheme is further modified by adapting the geometry of the motion compensated images into vertical and horizontal features to facilitate the Wavelet decomposition.

b Application to static scenes: sprites

As a special case of global motion compensation, *sprites* may be used to provide a mosaic representation of the scene (figure 3.4.4). It is particularly efficient when the camera is paning and/or zooming, while shooting a static scene. Again, each image from the sequence is projected onto a single system of coordinates. Motion compensated images are then combined into a single image: the *mosaic*, also known as *sprite*. This sprite can then be encoded and transmitted along with estimated camera parameters. Later, it was proposed to represent only the background as a sprite, and to process independently moving objects [WA94, PMCM01, LCL+97]. Sprite-coding is provided in MPEG-4 standard [WJ01].

3.4.2.2 Texture and structure synthesis

While approaches previously reviewed mostly rely on the motion information to compress image sequences, some other attempts at video compression have been focusing on textures and structures hold by successive images. Techniques aiming at synthesizing *textures* or *structures* have been studied for a while; recent progresses even led to techniques whose use may further improve image and video compression schemes.

a Synthesizing regions

Originally, texture and structure synthesis have been mostly used in image compression techniques, either for error resilience [RSB03] or compression [WSWX06, SWL06, LSW07a, LSW+07b] purposes. Typically, textures can be synthesized using parametric approaches [PS00] or template matching algorithms [Ash01, EF05]; while structures are commonly synthesized using inpainting techniques [BSCB00, CS01, CS02, Mas02, DCOY03]. More recently, approaches hybridizing structure and texture synthesis have also been proposed [BVSO02, ESQD05].

b Extension to video compression

As both texture and structure synthesis techniques proved to be quite efficient at compacting images, it was naturally proposed to extend their use to sequences of images. In [DH04], for instance, the provided compression scheme models the texture of the background via a parametric approach. Images are first segmented into uniformly-textured regions; each of them is parametrized using a steerable pyramid [PS00], then removed from the background, now reduced to a a set of color-uniform regions. Both simplified background and texture parameters are encoded and transmitted to the decoder, which regenerates the segmentation from the received background and synthesizes back the textures.

More recently, a few works have been attempting at embedding texture synthesis into H.264/AVC scheme. In [NNSHW05, NNHSW06, NNHW07], regions which do not hold detailed spatial information (structures, contours, ...) are called *detail irrelevant* textures. Corresponding *macroblocks* may then be encoded using rigid or non-rigid texture synthesis. In [GWL08, MADB10], finally, additional intra coding modes based on a set of template matching predictors are provided.

As for inpainting, it has also been used in modified H.264/AVC scheme. In [LSW07a, LSW+07b, ZSWL07a, ZSWL07b], *macroblocks* are classified into *reference* and *non-reference* blocks; the latter are removed and synthesized at the decoder via structure-aware (edge-based) inpainting. Motion threads are used to ensure that synthesized textures are stable across time. A similar approach is proposed in [BW07], wherein a pixel-based inpainting/texture synthesis algorithm is used to synthesize intra *macroblocks*.

3.5 Conclusion

This chapter provided a brief review dedicated to existing video coding schemes. From photometric to semantic approaches, it was seen that the video information can be considered in many different ways. Either way, image sequences are split in basic elements, including blocks, region, spatio-temporal objects, ..., each of which being encoded using decorrelation techniques such as the ones reviewed in chapters 1 and 2.

The classical video coding paradigm provides a closed-loop scheme wherein sequences are split into GOPs, frames, and *macroblocks*. Each *macroblock* may then be encoded using either intra or inter coding techniques. Such an approach proved to be very efficient at compacting sequences, while providing objectively and subjectively proper contents. Naturally, this approach has been standardized into various norms, including MPEG-x, and more recently H.264/AVC. Forthcoming video standard, recently baptised HEVC, will similarly submit to this approach as no disruptive techniques are yet to be integrated.

Disruptive approaches, on the other hand, have been largely proposed. Contrary to H.264/AVC and its predecessors wherein motion is processed on a frame by frame basis, numerous approaches have been focusing on a continuous representation of the motion. Indeed, pixels are moving along specific trajectories (known as *threads*), along which temporal correlation is maximal. 3D Wavelet coders, in particular, perform a temporal Wavelet decomposition along motion threads, and further decompose each temporal subband into spatio-temporal subbands with a 2D Wavelet transform.

Following this idea, analysis-synthesis approaches propose to separate the motion from the texture: both of them can then be processed fully independently. Such approaches include global motion compensation and texture/structure synthesis techniques. As they generally do not provide a perfect reconstruction of the images, objective quality metrics fail at assessing their intrinsic quality: they have not been considered for practical applications.

In the end, the popularity of the classical video coding paradigm results from several key points:

- its high efficiency at compacting image sequences;
- its successive incremental modifications: it did not suffer from any disruptive technology delaying its application;
- objective measures such as the PSNR are relatively fit to its distortions and artifact.

Any disruptive approach willing to take advantage upon H.264/AVC should then take these remarks into account. Ideally, a disruptive approach to video coding should provide a non-negligible increase in coding efficiency, while providing the same benefits as 3D Wavelets and analysis-synthesis techniques do. Also, it should naturally handle additional functionalities such as scalability, multiview coding, or video segmentation. In next chapters, an alternative representation of image sequences is then proposed through *motion tubes*, wherein above considerations are taken into account.

Chapter 4

Towards the notion of motion tubes

Here goes a recurrent observation on image sequences: most of their information (semantic, textures, shapes, contours, ...), persist across time. Based on such assumption, we here introduce a structure that represents a moving patch of texture within an image sequence, used as a basis element to represent an image sequence. We call this structure a *motion tube*.

Section 4.1 starts by identifying the assumption on which all the proposed work is based: the temporal persistence of texture. Then, section 4.2 will define the structure which we call *motion tube*; its role and parameters will be reviewed. Following section 4.3 will explain how such a structure can be used to represent and reconstruct an image sequence. Finally, section 4.4 will tackle the prospects of a representation of image sequences based on *motion tubes*, and section 4.5 will conclude the chapter.

4.1 Temporal persistence of textures

While looking at a sequence of natural images, one can see that a texture is likely to be found in several consecutive frames. Indeed, the textural information is carried by the objects of the scene and its the background: both objects and background are most often persistent through the time. However, due to camera motion and object displacement, any texture may follow a given trajectory and undergo a certain deformation. Figure 4.1.1 shows the evolution of two patches of textures. The green patch trajectory is a simple translation: it is not undergoing any deformation. As for the blue one, its trajectory is a translation as well, but also undergoes a rotation.

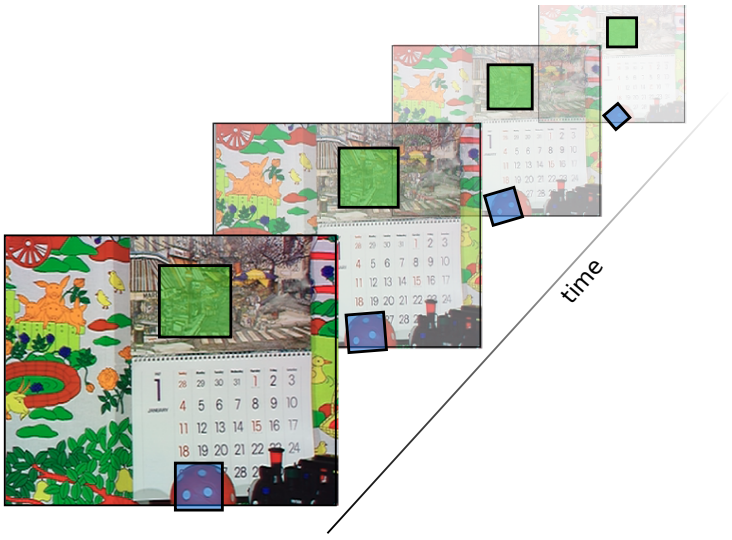
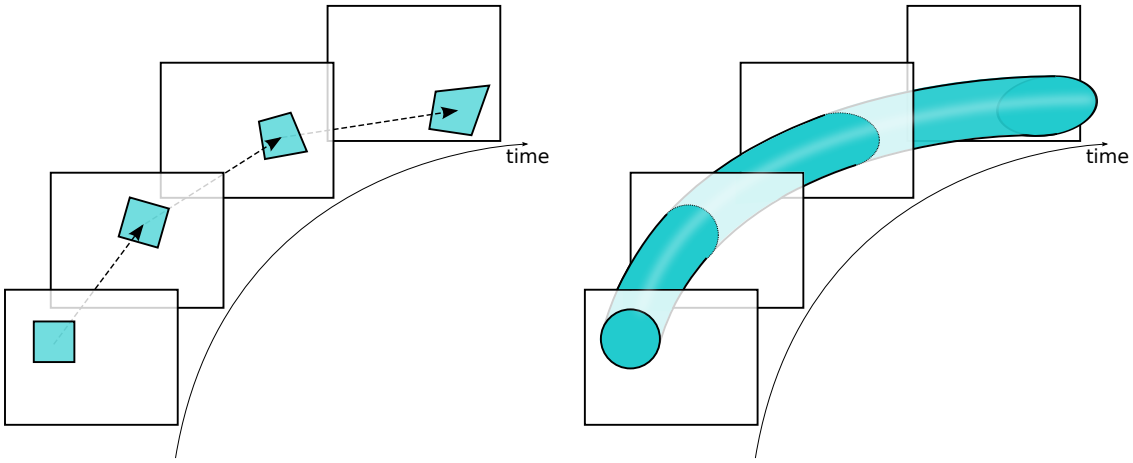


Figure 4.1.1: *Temporal persistence of textures in image sequences*

As a consequence, any sequence of images can generally be considered as a set of moving patches of texture, with respect to a given set of trajectories and deformations. From that perspective, it is reasonable to represent a sequence of images as such, assuming that it is possible to get a correct set of patches along with their trajectories and deformations parameters. Some textures, however, cannot be handled by such an approach: particle objects, liquids and transparencies, to cite a few of them, present such major changes of texture that it might be impossible to track their evolution over time without any specific approach. The current work focuses on sequences which do not present such delicate textures, and aims at finding a set of patches of texture which represent such sequences.

As we can see in figure 4.1.2, the successive shapes of a moving texture patch reminds a tube whose section is deforming. From now on, moving patches of textures will be referred to as *motion tubes*. Such *motion tubes* will be able to track moving texture across time and space, thus exploiting their temporal persistence. Following sections will attempt at giving a formal definition of *motion tubes*, then explain how they can represent a sequence of images.



(a) *A moving patch of texture* **(b)** *A crude representation of a* motion tube

Figure 4.1.2: *From a moving patch of texture towards the* motion tube

4.2 *Motion tube*: a definition

Let $\mathcal{M}_\mathcal{T}$ be a *motion tube*. It attempts to describe a patch of texture along with its evolution across time and space, through a sequence of images. Hence, $\mathcal{M}_\mathcal{T}$ will be characterized by 3 types of information: its texture $\mathcal{T}$, its lifespan $\mathcal{L}$, and its deformation parameters $\mathcal{W}$:

$$\mathcal{M}_\mathcal{T} = \{\mathcal{T}, \mathcal{L}, \mathcal{W}\} \ . \tag{4.1}$$

Following subsections will detail the contents and roles of $\mathcal{T}$, $\mathcal{L}$, and $\mathcal{W}$.

4.2.1 Lifespan of a motion tube

A *motion tube* starts at an instant t_{start} and ends at an instant t_{end}. This time interval defines its lifespan $\mathcal{L}$:

$$\mathcal{L} = [t_{\text{start}}, t_{\text{end}}] \tag{4.2}$$

4.2.2 Deformation of a motion tube

In order to cope with texture displacements and deformations, an appropriate deformation model has to be set up. The deformation of a texture may be described as the result of a transform whose transfer function is a *warping* operator w. A set of warping operators $\mathcal{W} = \{w_{i\to i+1}\}$, $t_i \in [t_{\text{start}}, t_{\text{end}}[$ are provided with $\mathcal{M}_\mathcal{T}$. The deformation transform is supposed to be invertible, and

$$w_{j\to i} = w_{i\to j}^{-1} \tag{4.3}$$

This hypothesis implies that the shape of the patch of texture is consistent. In particular, it is considered that neither occlusion nor disocclusion may impact the patch. $\mathcal{W}$ describes the deformation of a *motion tubes* as a succession of consecutives transforms from t to $t+1$. However, the deformation between 2 non adjacent time instants might also be described. To ease their manipulations, warping operators $\mathcal{W}$ can be composed, such that

$$w_{i\to k} \circ w_{k\to j} = w_{i\to j}\,. \tag{4.4}$$

Let $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t)$ be the support of the *motion tube* $\mathcal{M}_\mathcal{T}$ in the image $I(t)$ at time instant t, $t_{\text{start}} \leq t \leq t_{\text{end}}$. $\mathcal{M}_\mathcal{T}$ is a 2D+t volume whose sections are $\{\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t)\,|t_{\text{start}} \leq t \leq t_{\text{end}}\}$. The deformation of the shape of a *motion tube* between time instants t_i and t_j is then given by

$$\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_j) = w_{i\to j}\left(\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_i)\right)\,. \tag{4.5}$$

The notion of trajectory is crucial to the notion of *motion tubes*. Indeed, the temporal persistence of textures will be captured only if motion tubes naturally exhibit a deformation along a trajectory $T(t)$. A luminance sample $\mathcal{P}$ of the patch of texture, due to the displacement and the deformation of the texture, will follow a trajectory $T_\mathcal{P}(t)$. Let $\mathcal{G}_{\mathcal{M}_\mathcal{T}}(t)$ be the center of gravity of the *motion tube*'s support $\mathbf{\Omega}_t(\mathcal{M}_\mathcal{T})$ at time instant t. The trajectory $T_{\mathcal{M}_\mathcal{T}}(t)$ of a motion tube will be given by the trajectory of the centre of gravity $\mathcal{G}_{\mathcal{M}_\mathcal{T}}(t)$ of the patch at each time instant t, such that $\mathcal{G}_{\mathcal{M}_\mathcal{T}}(t) = T_{\mathcal{M}_\mathcal{T}}(t)$ for $t_{\text{start}} \leq t \leq t_{\text{end}}$. Figure 4.2.1 illustrates this principle.

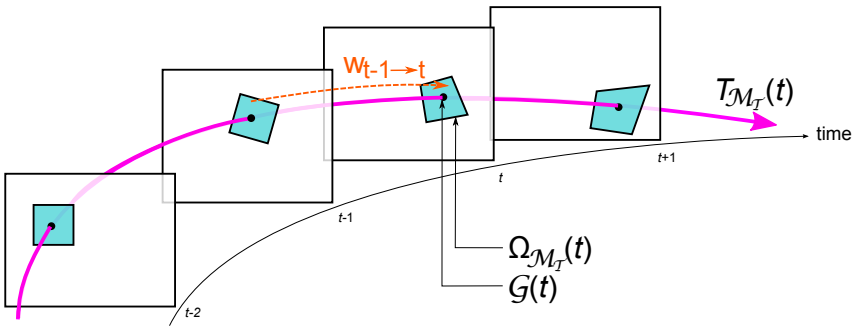


Figure 4.2.1: *Trajectory and deformation of a* motion tube

4.2.3 Textural information

A *motion tube* being a moving patch of texture, it obviously needs to incorporate a textural information. This texture will be able, using an appropriate deformation, to reconstruct the textural information at any instant t ($t_{\text{start}} \leq t \leq t_{\text{end}}$). However, due to resolution losses, illumination changes, noise or any natural change, the texture itself may vary over time. Thus, the textural information $\mathcal{T}(t)$ is defined as a function of time, and reflects the contents and the changes of the texture across time:

$$\mathcal{T}(t) = \begin{cases} \mathcal{T}(t) & \text{if the texture changes across time} \\ \mathcal{T} & \text{if the texture is static} \end{cases} \tag{4.6}$$

4.2.4 Coding motion tubes

In the context of video coding, *motion tubes* will need to be transmitted, hence encoded. $\mathcal{T}$, $\mathcal{L}$ and $\mathcal{W}$then need to be encoded. Resulting information is then written as follows:

$$\mathcal{C}_{\mathcal{M}_\mathcal{T}}(\mathcal{M}_\mathcal{T}) = \left\{\mathcal{C}_\mathcal{L}(\mathcal{L}),\quad \mathcal{C}_\mathcal{T}(\mathcal{T}),\quad \mathcal{C}_\mathcal{W}(\mathcal{W})\right\} \tag{4.7}$$

where $\mathcal{C}_{\mathcal{M}_\mathcal{T}}$ is the *motion tube* encoding operator, $\mathcal{C}_\mathcal{L}$ is the lifespan encoding operator, $\mathcal{C}_\mathcal{T}$ is the textural encoding operator, and $\mathcal{C}_\mathcal{W}$ is the deformation encoding operator. Obviously, the coding process of a *motion tube* is not independent from the coding process of other *motion tubes* to encode. Consequently, (4.7) can be rewritten as

$$\mathcal{C}_{\mathcal{M}_\mathcal{T}}(\mathcal{M}_\mathcal{T}) = \left\{\mathcal{C}_\mathcal{L}(\mathcal{L},\{\mathcal{M}_{\mathcal{T}i}\}),\quad \mathcal{C}_\mathcal{T}(\mathcal{T},\{\mathcal{M}_{\mathcal{T}i}\}),\quad \mathcal{C}_\mathcal{W}(\mathcal{W},\{\mathcal{M}_{\mathcal{T}i}\})\right\} \tag{4.8}$$

where $\{\mathcal{M}_{\mathcal{T}i}\}$ is a set of *motion tubes* are transmitted along with $\mathcal{M}_\mathcal{T}$. $\{\mathcal{M}_{\mathcal{T}i}\}$ might be a set of neighbouring *motion tubes*, or any other set of *motion tubes* whose encoding parameters may influence the coding of $\mathcal{M}_\mathcal{T}$.

4.3 An image sequence representation based on motion tubes

As explained in the previous sections, a *motion tube* aims at tracking a moving patch of texture over time. Now that such a structure has been carefully defined, building a representation for image sequences upon *motion tubes* is quite straightforward. It consists in building a set of *motion tubes*, i.e. a set of moving patch of textures, which will be able to represent the whole sequence or its largest part. Indeed, section 4.4.2 will explain how *motion tubes* may not be able to entirely reconstruct a sequence. Due to the use of *motion tubes*, this representation naturally exhibits the temporal persistence of textures, thus optimizes the re-use factor of textures to transmit.

4.3.1 An image sequence as a set of motion tubes

Figure 4.3.1 illustrates a *motion tube* based representation. Five tubes have been initialized at $t_{t_{start}} = t_2$, and their textures have been tracked from t_0 to t_4. Assuming the motion field is uniform on areas reconstructed by *motion tubes* 1, 2 and 3, we can see that they are kept together through the whole GOP. Tube 4, on the other hand, behaves differently to map a discontinuity of the motion field. Tubes 1, 2, 3 and 4 share the same start ($t_{t_{start}} = t_0$) and end ($t_{t_{end}} = t_4$) instants, and their lifespan $\mathcal{L}$ is $[t_0, t_4]$. Finally, tube 5 does not appear at t_0 nor t_4 because the texture it carries is not present at those instants: its lifespan is then $[t_1, t_3]$.

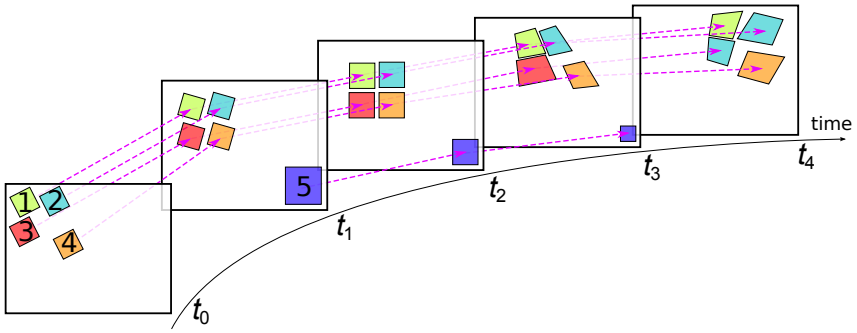


Figure 4.3.1: *An image sequence partially reconstructed from a few* motion tubes

This being said, the main problematic of this representation comes down to find a right set of *motion tubes*. Let I_t be an image from the sequence $\mathcal{S}_\mathcal{I}$ at time instant t. It is assumed that $\mathcal{S}_\mathcal{I}$ can be represented by a set of *motion tubes* $S = \{\mathcal{M}_{\mathcal{T}_i}\}$ where $N = \text{Card}(\mathcal{S})$ is a finite number. The synthesized image $\overline{I}_t$ at time instant t is given by

$$\overline{I}_t = \bigcup_{i=1}^{N} \mathcal{R}\left(\mathcal{M}_{\mathcal{T}_i}, w_{t_{ref} \to t}\right) \tag{4.9}$$

where $\mathcal{R}$ is the rendering operator of the *motion tube*: it aims at synthesizing the tracked texture according to its parameters at current time instant. While the existence of a matching set of *motion tubes* S can be easily assumed, there is no uniqueness for such a representation.

4.3.2 A concrete approach to motion tubes

A *motion tube* is driven by a large set of parameters (lifespan, shape, trajectory, texture, ...). In practice, its optimization may prove to be a difficult task. In order to simplify this problem, it is proposed to create groups of *motion tubes* which, for now, will share the same temporal parameters. These groups are called *families* of *motion tubes*.

4.3.2.1 A family of motion tubes

A *family* of *motion tubes* $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ is a set of N tubes whose members share the same reference instant t_{ref}

$$\begin{cases} \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}}) = \{\mathcal{M}_{\mathcal{T}_i}\}, \quad i \in [0, N-1], \\ Card\left(\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})\right) = N \end{cases} \tag{4.10}$$

$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ can be updated according to any received control data: removal or temporary deactivation of a *motion tube*. Control data might also modify the properties or the behaviour of any given *motion tube* from the set. At time instant t, $\overline{I_t}$ is synthesized by rendering all the tubes from $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$, such that:

$$\overline{I_t} = \mathcal{R}\left(\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}}), t\right) = \bigcup_{i=1}^{N} \mathcal{R}\left(\mathcal{M}_{\mathcal{T}_i}, w_{t_{ref}\rightarrow t}\right) \tag{4.11}$$

where $\overline{I_t}$ is the reconstruction of I_t, and t_{ref} the time instant when $\mathbf{T}_i$ has been initialized. $\mathcal{R}(\mathcal{M}_\mathcal{T})$ operator aims at rendering $\mathcal{M}_\mathcal{T}$ using appropriate weightings. $\bigcup$ operator is the composition operator: it combines all the tubes from $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$. When one or more *motion tubes* are available to reconstruct an area, $\bigcup$ will chose which of them should be used. When no tubes are available, $\bigcup$ might also reconstruct some of the unpredicted areas and may use surroundings *motion tubes*. These operators will be tackled in section 4.4.2.

4.3.2.2 Several families to reconstruct the sequence

A sequence of images represents a scene within which, due to camera motion or objects displacements, background may change and objects may appear or disappear. Likewise, the availability of the textures will also vary across time. A family of motion tubes sources the textural information from a common reference instant. As a consequence, it will not be able to register the entire textural information: a single family cannot entirely represent a complex sequence.

Therefore, several families of *motion tubes* will be required to provide an appropriate representation. These families may overlap temporally and/or spatially. Figure 4.3.2 illustrates the use of three families of motion tubes. Family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ is referenced at instant t_0 and its lifespan is $\mathcal{L} = [t_0; t_3]$. Family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_2)$ is referenced at instant t_2 and its lifespan is $\mathcal{L} = [t_0; t_4]$. Family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$ is referenced at instant t_4 and its lifespan is $\mathcal{L} = [t_2; t_4]$.

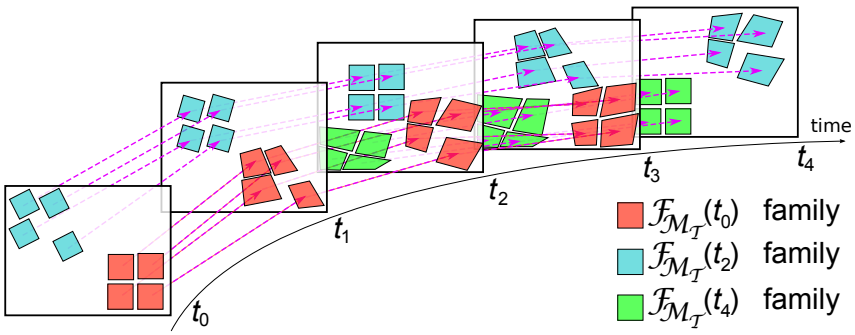


Figure 4.3.2: *Several families to reconstruct the sequence.*

A convenient way to instantiate several *motion tube* families is to split the sequence into GOPs and create a family for each of them. Each family is initialized at the GOP start instant. This particular solution is illustrated in figure 4.3.3 using GOPs of 8 time instants. The first GOP is reconstructed with the red family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ whose tubes end at t_8, the second GOP by the blue family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ whose tubes end at t_{16}, and the third GOP is reconstructed by the green family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{16})$ which ends at t_{24}. However, camera motion and objects displacement may force us to create several families within the same GOP. Chapter 6 will further investigate this possibility.

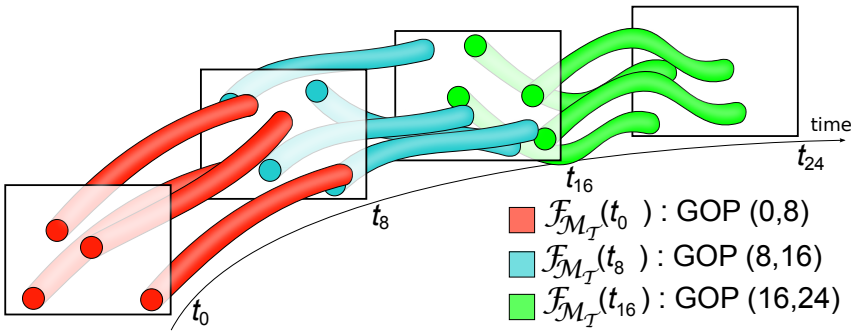


Figure 4.3.3: *GOP paradigm in the context of* motion tubes

4.3.3 Motion tubes: an Analysis-Synthesis approach to video coding

The representation of the sequence can be split into two main steps: an analysis step during which the textures are tracked across time, and a synthesis step where the sequence is reconstructed by a set of tracked textures. Therefore, a coder based on a *motion tube* representation of the sequence is an AS-like video coder. Figure 4.3.4 represents the associated block diagram.

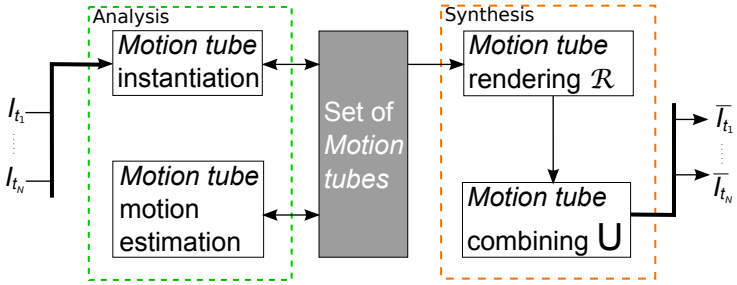


Figure 4.3.4: *Analysis-Synthesis schematization of a* motion tube *based coder*

From an image sequence coding perspective, *motion tubes* present two types of data to encode:

1. the textural information $\mathcal{T}$, which can be encoded by any traditional intra coding technique, along with prediction schemes to code its evolution across time,
2. (resp.) temporal and deformation information (resp. $\mathcal{L}$ and $\mathcal{W}$), which will need dedicated coding techniques.

Motion tubes, by their nature, can benefit from the temporal persistence of moving textures: their tracking is simplified. In particular, the motion estimation of a *motion tube*, at a given instant, can be guided by its trajectory at previous or next instants. It will tend to reduce the discrepancies of the motion field, and the motion coding cost. It will also maximize the tube's lifespan, due to an enhanced tracking, thus minimizing the amount of textural information to be sent. Furthermore, *motion tubes* can start and end at any time instant, hence fit appropriately the instants of apparition and disappearance of the tracked textures.

They can be either dependent or independent from each other (neighbouring *motion tubes* undergo the same changes), *connected* or *disconnected* (neighbouring *motion tubes* may be joint or disjoint). Keeping *motion tubes* connected will constraint the motion field to be continuous, while disconnections will be able to represent the ruptures of the motion field. Again, this will lead to an efficient representation.

4.4 Prospects of the representation

In this chapter, we have introduced the *motion tube* paradigm and its application to the representation of image sequences. By tracking patches of texture through time and space, one can exploit the temporal persistence of textures. In image sequences, this persistence is responsible for most of the redundancy; *motion tubes* should hence be able to identify then remove this redundancy.

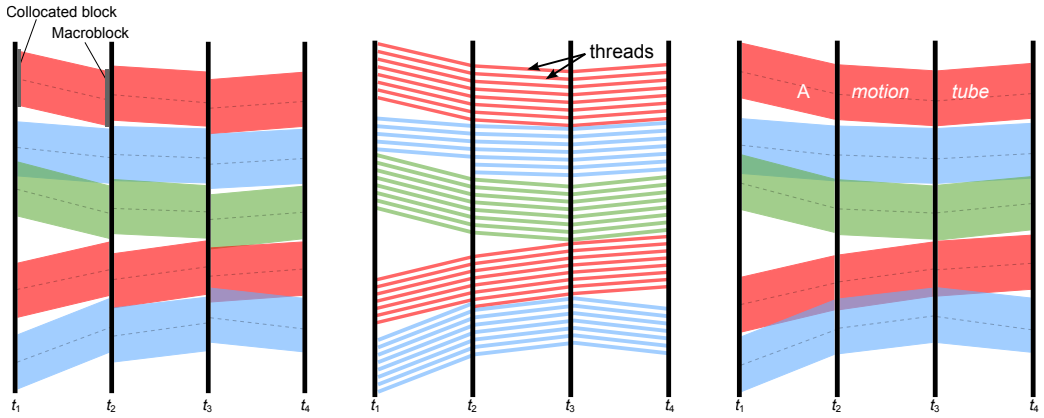
4.4.1 Features provided by the representation

4.4.1.1 A temporally continuous block-based representation

The popularity of the classical representation, as it is used in ITU-T H.264/AVC standard, is notably due to the use of blocks of pixels as basis representation units. However, it resets the image partition and the motion grid at each time instant, thus providing a discontinuous description of the images contents along the temporal axis.

On the other hand, MCTF, Motion threads, and Barbell lifting [XWX+04] showed how much important it is for a representation to describe the motion information as continuously as possible. Yet, theses representations are much more complex than the block-based representation, and describe the temporal evolution on a pixel-by-pixel basis.

With *motion tubes*, the advantages of both approaches are kept as they may be easily used to track blocks of texture in a continuous fashion. Figure 4.4.1 shows a sectional view of the motion field along several time instants, as it is described by the classical representation, the motion threads, and the proposed *motion tubes*.



(a) *Block-based representation* **(b)** *Thread-based representation* **(c)** *Tube-based representation*

Figure 4.4.1: *Continuous vs discontinuous representations of the motion field: sectional view)*

4.4.1.2 The ability to represent both continuous and discontinuous displacements and deformations

The traditional block-based representation easily captures the discontinuities of the motion field, but fail at precisely describing its smooth variations. Conversely, mesh-based representations [Bru90, Dud96, Lec99, LG08] naturally handle the continuities of the motion field, but cannot represent its ruptures (or discontinuities). Even advanced techniques that have introduced solutions to disconnect neighbouring meshes [Mar00, Cam04] failed to provide an efficient workaround. On the other hand, *motion tubes* naturally offer the ability to represent both continuities and discontinuities of the motion field. Indeed, each *motion tube* can be considered as an individual mesh undergoing specific displacements and deformations.

4.4.1.3 An invaluable spatio-temporal information

In region-based representations [YW94, SGPK94, SMP+97, SM99, XLLZ01], it is proposed to track the evolution of regions from an image sequence. However, most of them suffer from the complexity of the deformations objects can undergo. The displacements and the deformations of the object or the background are often quite complex, and may require parametric models.

On the other hand, motion tubes focus on smaller patches of texture, hence aim at representing a local motion. Local motions are far more easy to capture and describe; and more simple motion models may be used (see figure 4.4.2). Even though *motion tubes* do not account for the spatio-temporal contents on a semantic level, they still provde a relatively high level description of the sequences, which can be easily processed to extract semantic informations such as objects shapes and locations.

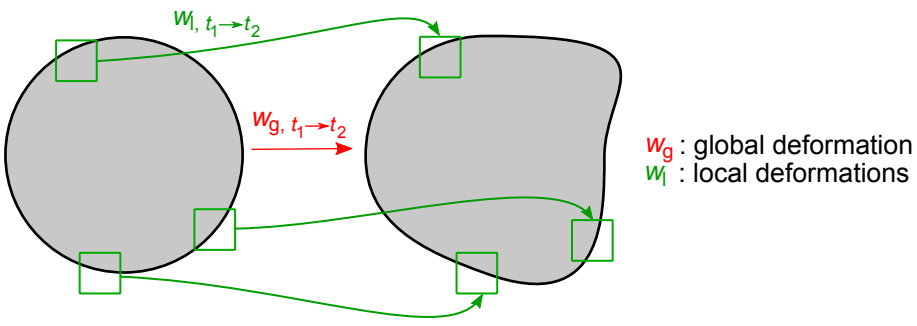


Figure 4.4.2: *How a complex deformation may be seen as a set of simple local deformations*

Motion tubes have already been proposed by Péchard *et al.* in [PBC07, Péc08] to assess the quality of ITU-T H.264/AVC HDTV image sequences. Their use is strictly limited to quality assessment, and does not concern the field of video representation and compression. Still, the provided analysis further confirms the fact that *motion tubes* may be very interesting in terms of spatio-temporal representation abilities.

4.4.2 Problems raised by the representation

As a preliminary example, figure 4.4.3 represents the reconstruction of *Foreman* sequence from instant 0 to 8. A single family $\mathcal{L}_0$ of *motion tubes* has been instantiated at t_0, and is tracked towards t_8. As this point, *motion tubes* deformation model has not yet been defined, therefore they undergo simple translations defined by their trajectory $\mathcal{T}(t)$.

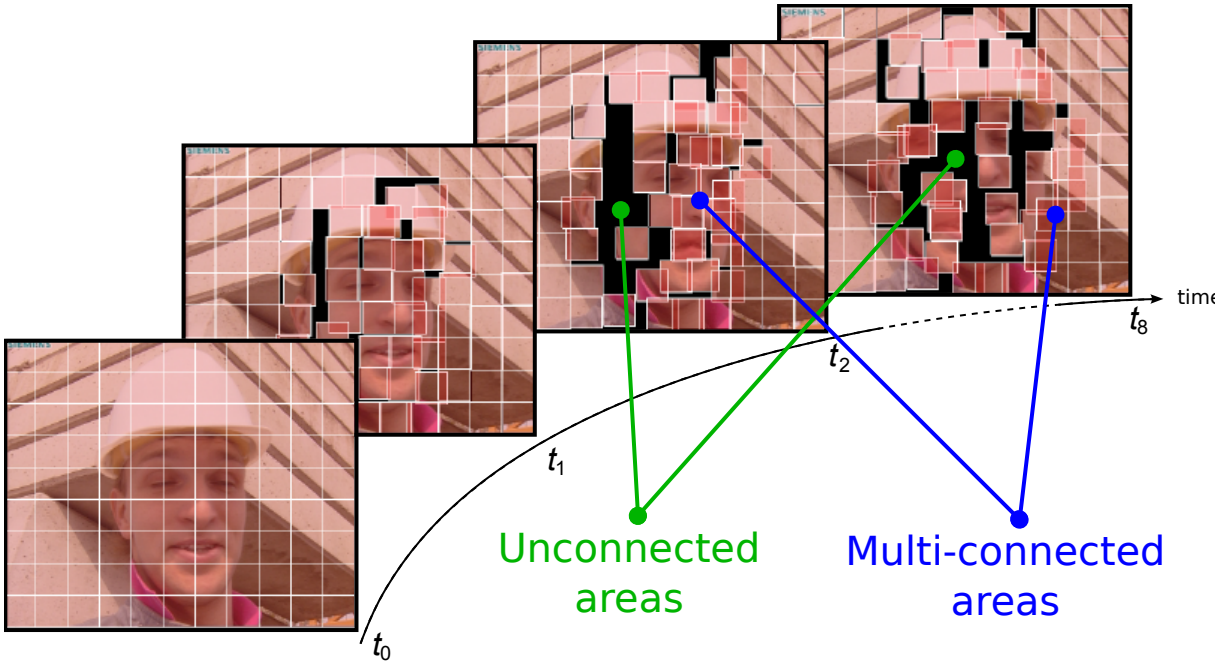


Figure 4.4.3: *A preliminary example of* motion tube *based reconstruction of sequence* Foreman

4.4.2.1 Which deformation model should be employed?

As mentioned earlier, no deformation model was used to represent *Foreman* sequence in figure 4.4.3. Designing an appropriate deformation model will be a major issue: which deformations have to be handled, how can we offer a compact representation of these deformations?

4.4.2.2 An incomplete reconstruction

As mentioned earlier, sole *motion tubes* might not be able to entirely represent a sequence of images. In that, they may be seen as a synthetic representation as they were seen in 1.1.2. In particular, complex scenes on which the tracking of textures is difficult or even impossible won't be entirely reconstructed by any reasonable set of *motion tubes*.

In figure 4.4.3, unconnected areas correspond to areas which are not reconstructed by any *motion tube*. Dedicated mechanism will then need to be proposed to handle these areas. The composition $\bigcup$ of the *motion tubes* should be able to fill these holes, using surrounding *motion tubes* (*e.g.* with inpainting).

4.4.2.3 Inter-tube redundancies

In *multi-connected areas*, several *motion tubes* contribute to the reconstruction. Again, the composition $\bigcup$ will be in charge of dealing with these regions, and avoid as much redundancies as possible, while ensuring the reconstruction to be accurate enough. The fewer *motion tubes*, the more compact the representation is likely to be.

4.4.3 Life and death of motion tubes

Another major issue lays in the estimation of the temporal parameters of a *motion tube*. Indeed, it is necessary to fit its lifespan to the content of the sequence. Therefore, a quality assessor dedicated to *motion tubes* will have to be built: which criteria have to be used? Later on, a decision mechanism has to be designed; it will set the *motion tubes* lifespan.

4.4.4 Extension of motion tubes to other applications

Motion tubes offer a very flexible way to represent a sequence, and allow for numerous possibilities. Among them, multiple description is inherent to the representation. Indeed, multiple description send several different versions of the same signal; the receiver will reconstruct the signal by using all the contributions he received. This reconstruction may be, or not, guided by the sender. *Motion tubes* or families of *motion tubes* might represent the same part of the sequence with more than one texture: this is a case of multiple description.

As a special case of multiple description, scalability is indeed naturally handled by this representation. Spatial and quality scalabilities may be brought through several families of *motion tubes* of different sizes, resolutions or qualities, predicting the same area.

The segmentation of image sequences has been always and is still a complex task. Since *motion tubes* track a given texture across time, the semantic information it carries is stable across time: a *motion tube* tracks either a part of an object, or a part of the background. Hence, they offer a compact representation, while giving crucial information on the semantic content of the sequence. Due to the inherent temporal coherence of the representation, *motion tubes* can be used to efficiently segment an image sequence. Indeed, neighbouring *motion tubes* sharing the same lifespan and/or motion parameters may be merged into a semantic region.

Even further, *motion tubes* can be applied to 3D video: if textures can be tracked across time, they can also be tracked from a view to another view. One can think of 4-D version of a *motion tube* which represent the deformation of a patch of texture across time and also across the different views [CPML10].

4.5 Conclusion

This chapter started with the observation that, in image sequences, textures are found in many successive images, whether they have been translated or deformed in any way we could think of. This led to a structure called *motion tube* that aims at tracking a patch of texture across time. This structure is composed of several information: a temporal information that indicates when the patch of texture is found, a deformation information that describes its evolution across time, and finally a textural information which carries the texture itself and its eventual updates.

The *motion tube* paradigm has been introduced: an image sequence is considered as a set of moving textures, hence as a set of *motion tubes*. The main issue of this work will thus be to find an appropriate set of *motion tubes* to reconstruct the sequence. To ease this problem's formulation and resolution, *families* of *motion tubes* have been introduced: they consist in a set of *motion tubes* initialized at the same reference instant. Since then, an image sequence is seen as a set of families of *motion tubes*. Using such families, one could define the concept of GOP in a *motion tube* based representation.

Later on, the proposed representation was studied: its major strength would be its ability to capture the redundancies of image sequences by exploiting the temporal persistence of the texture. From a coding perspective, *motion tubes* seem to be good candidates as a basis element to a new video coder. However, the representation might lead to unconnected and/or multi-connected areas. The latter phenomenons revealed themselves to be critical problems for the representation. Also, the estimation of the temporal parameters seemed to be far from trivial; in particular, designing a quality measure dedicated to the *motion tubes* would consist in a critical step. Finally, the deformation model of *motion tubes* proved itself to be another field of investigations.

In the end, *motion tubes* are made from three types of information: their motion $\mathcal{W}$, their texture $\mathcal{T}$ and their lifespan $\mathcal{L}$. Each of them, and their influence on the synthesized sequences, will be studied in dedicated chapters:

- **chapter 5** will provide a relatively simple **deformation model** which exhibits the notion of trajectory, and allows for various deformations to be described;
- then, **chapter 6** will focus on the synthesized **textures** and provide mechanisms which guarantee the reconstructed images to be complete;
- finally, **chapter 7** will provide various **life and death mechanisms** which will be used to adapt the **lifespan** of the *motion tubes* to the images contents, and remove inefficient or redundant ones.

Perspectives of *motion tubes* seem to be pretty various. Not only they seem to be able to represent and encode 2D image sequences, they might also help with their spatio-temporal segmentation. Due to their nature, they naturally handle multiple description, in particular any kind of scalability. Even further, they might also be able to represent 3D image sequences which propose several views for the same time instant.

Chapter 5

Modelling and transmitting motion tubes deformations

Coping with the displacements and the deformations of a patch of texture across time is one of the greatest challenges towards an tube-based image sequence representation. Motion compensation, as seen in chapter 2, provides a wide variety of techniques dedicated to the representation of the motion, hence the textural deformation between images. However, it was also seen in chapter 3 that video compression brings additional constraints: the compactness of the parameters describing the motion information. Moreover, practical applications further limit the choice to models of limited complexity. Taking these considerations into account, this chapter provides a motion model to the *motion tubes*. The proposed model offers a realistic trade-off between reconstruction quality, required bitrate and complexity:

- it is a good match to a large set of common deformations,
- it requires relatively few parameters to be transmitted,
- and it does not require a large amount of computations to be estimated nor applied.

Building on the literature from chapters 2 and 3, section 5.1 will first investigate which features an idealistic motion model should provide. Then, section 5.2 will provide a global insight on the proposed motion model. Section 5.3 will detail the chosen hybrid motion model and section 5.4 will investigate the basic features of the motion model. Then, section 5.5 will incorporate an essential feature to the motion model: the notion of trajectory. Section 5.6 will introduce a mechanism in charge providing *motion tubes* of variable sizes. Finally, 5.7 will give a quick overview of the entropy coding scheme used to compact the motion information into a binary bitstream, while section 5.8 will conclude the chapter.

5.1 Identifying desirable features for an idealistic motion model

5.1.1 Block-based and mesh-based models in the literature: a summary

Chapter 2 highlighted several characteristics an idealistic motion model should grant. At first, it should be able to describe as many deformations as possible. In practice, translations and slightly more complex warpings are efficient enough in most cases. In addition, it should be able to represent both continuities and discontinuities of the motion field. In an eye to compression, finally, models based on geometrical patterns seem to provide a relatively compact representation of the motion field. In addition, their complexity is fairly reasonable, provided that the geometrical patterns are independent enough from each other.

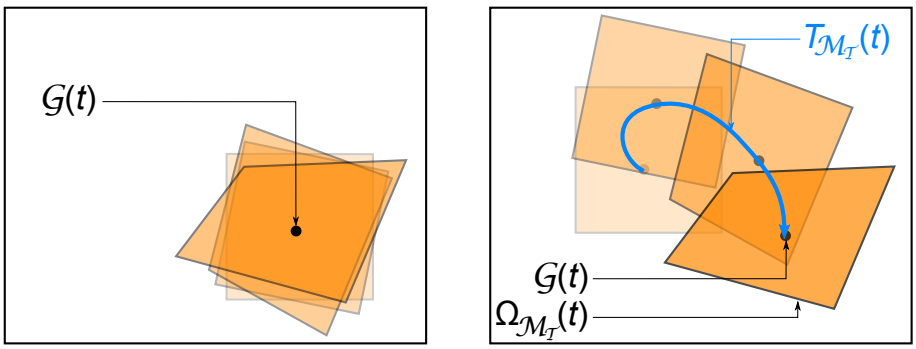
Among pattern-based models, block-based and mesh-based models (respectively reviewed in sections 2.2.3.1 and 2.2.3.2) are very popular in video compression applications. Historically, however, blocks are generally preferred to meshes as they are used in standardized video compression schemes. Yet, numerous techniques employing blocks and/or meshes have been provided (see section 2.4). Table 5.1.1 summarizes the ability of several models (including BMC, OBMC, CGI and a few hybridized variants) to fulfill a set of features which we consider to be critical regarding our work. From table 5.1.1, it can be seen that hybrid CGI-based approaches generally provide the best abilities regarding motion representation. However, they do not fulfill our complexity requirements. In the end, SOBMC provides the best trade-off between modelling abilities and complexity.

Motion models	BMC	CGI	OBMC	SCGI	SOBMC	CGI-OBMC
Translating motions	✓	✓	✓	✓	✓	✓
Complex motions	×	✓	~	✓	~	✓
Motion continuities	×	✓	✓	✓	✓	✓
Motion discontinuities	✓	×	×	✓	✓	✓
Time-continuous motion representation	×	✓	×	×	×	×
Block-based	✓	×	✓	~	✓	~
No blocking effects	×	✓	~	✓	~	✓
Pattern independence	✓	×	~	×	~	×
Computational simplicity	✓	×	✓	×	✓	×

Table 5.1.1: *Comparisons of several features of existing motion models*

5.1.2 Naturally exhibiting the spatio-temporal trajectory of the motion tubes

As a *motion tube* is moving and deforming across several time instants, its motion model $\mathcal{W}$ consists of a succession of warping operations $w_{i\rightarrow i+1}$, describing its deformation from an instant to the next one (figure 5.1.1a). However, *motion tubes* may need to be temporally processed into different fashions (time-increasing, time-decreasing, hierarchical, etc): an ideal motion model should then be able to describe the deformation of a *motion tube* from any arbitrary time instant to any other arbitrary one. As a consequence, an idealistic motion model should be able to compose operators $w_{i\rightarrow i+1}$ in any thinkable way. Finally, an ideal motion model should clearly exhibit the trajectory $T_{\mathcal{M}_T}(t)$ of a *motion tube* (figure 5.1.1b); the latter may even be encouraged to follow a trajectory as regular as possible.



(a) *Deformations around the gravity centre* $\mathcal{G}_{\mathcal{M}_T}(t)$ **(b)** *Overall displacement and deformation*

Figure 5.1.1: *Sectional view of a* motion tube*: projection of* $\mathbf{\Omega}_{\mathcal{M}_T}(t)$ *onto a single image plane*

5.2 An overall insight on the proposed motion model

In order to establish an appropriate motion model for the *motion tubes*, previous section identified the SOBMC as a sound candidate since its abilities match a large number of our requirements. However, the SOBMC has been only used to model the deformation between a couple of images, and does not provide (unlike active meshes, for instance), an intrinsic ability to describe the deformation of a whole GOP in a continuous manner. As a consequence, the provided motion model will extend the abilities of the SOBMC to account for the spatio-temporal nature of the *motion tubes*. This section will now outline the proposed motion model, and highlight several improvement axis:

- a motion model describing the spatio-temporal evolution of a **family** of *motion tubes* – section 5.2.1;
- an **hybrid** deformation model capable of describing **continuous** and **discontinuous** deformations – section 5.2.2;
- the ability to **adapt** the **dimensions** of the *motion tubes* to the images contents – section 5.2.3;
- several **regularization** mechanisms limiting the amount of motion discrepancies – section 5.2.4;
- a **simple** motion estimation process which evaluates *motion tubes* **one-at-a-time** – section 5.2.5.

5.2.1 Describing the spatio-temporal information through a single family of motion tubes

In chapter 4, it was proposed to split image sequences into GOPs of limited duration. In each of them, it was proposed to use a family (or several families) of *motion tubes* to exhibit the spatio-temporal textural persistence. In this chapter, it will be assumed that each GOP $\{I_0, \ldots, I_G\}$ is described through a single family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ of *motion tubes*.

Among the guidelines listed within the introduction of this chapter, it was mentioned that relying on a block-based partition of the image plane would greatly simplify the design of a competitive video compression scheme. In addition, this prevents our tube-based representation to be too much disruptive in regards to the standard block-based representation. Keeping this in mind, it is proposed to partition the reference image I_0 into disjoint and regular square blocks $\mathcal{B}_i$. Each of them are then used to initialize a *motion tube* $\mathcal{M}_{\mathcal{T}i}$ whose texture $\mathcal{T}_i$ is sourced from $\mathcal{B}_i$.

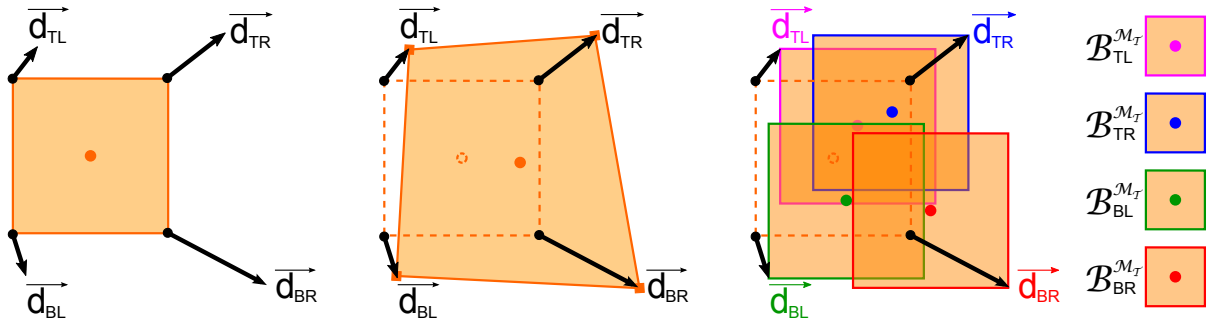
$$\forall \mathcal{M}_{\mathcal{T}_i} \in \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0) \quad \left\{ \begin{array}{rcl} \mathcal{L}_i & = & [t_0, t_G] \\ \mathbf{\Omega}_{\mathcal{M}_{\mathcal{T}i}}(t_0) & = & \mathbf{\Omega}_{\mathcal{B}_i} \\ \mathcal{T}_i & = & I_0(\mathbf{\Omega}_{\mathcal{B}_i}) \end{array} \right. \tag{5.1}$$

where $\mathbf{\Omega}_{\mathcal{B}_i}$ is the support of block $\mathcal{B}_i$ and $I_{\text{ref}}(\mathbf{\Omega}_{\mathcal{B}_i})$ is the patch of texture hold by block $\mathcal{B}_i$. In the preliminary illustration provided by figure 4.4.3 in chapter 4, the first image of sequence *Foreman* was similarly split into 32×32 square blocks which were used to instantiate a family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ of *motion tubes*.

5.2.2 In between blocks and meshes: a modified Switched OBMC motion model

In order to represent a wide enough variety of deformations, the motion of each *motion tube* $\mathcal{M}_\mathcal{T}$ is described through four motion vectors. To each corner of the quadrilateral patch of texture being tracked is associated a motion vector which describes its displacement in regards to its initial position in the reference image. At time instant t_n:

1. $\overrightarrow{d_{\text{TL}}}(t_n)$ describes the displacement of the **top-left** corner of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$;
2. $\overrightarrow{d_{\text{TR}}}(t_n)$ describes the displacement of the **top-right** corner of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$;
3. $\overrightarrow{d_{\text{BL}}}(t_n)$ describes the displacement of the **bottom-left** corner of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$;
4. $\overrightarrow{d_{\text{BR}}}(t_n)$ describes the displacement of the **bottom-right** corner of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$;



(a) *Initial block of texture* $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{ref})$ **(b)** *Idealistic forward CGI warping* **(c)** *OBMC-like warping approximation*

Figure 5.2.1: *Forward motion compensation of a* motion tube*: in between OBMC and CGI*

The proposed deformation model is illustrated in figure 5.2.1. Figure 5.2.1a shows the initial block of texture sourced from I_0 along with its motion vectors. As each motion vector describes the local displacement of a single corner, the proposed model can be compared to the CGI, wherein all four corners are behaving as control points: the original square block can then be seen as a mesh whose deformations motion compensate the corresponding patch of texture. Figure 5.2.1b illustrates such an idealistic deformation.

5.2.2.1 Simplifying the CGI into a modified OBMC

In practice however, the CGI does not match our requirements as corresponding interpolation operations are too much complex. Instead, it is proposed to approximate the CGI with a variant of the OBMC which simply relies on blocks

and translations. At any time instant t_n, the support $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$ of $\mathcal{M}_\mathcal{T}$ is given by the union $\mathcal{B}_{\text{TL}}^{\mathcal{M}_\mathcal{T}}(t_n) \cup \mathcal{B}_{\text{TR}}^{\mathcal{M}_\mathcal{T}}(t_n) \cup \mathcal{B}_{\text{BL}}^{\mathcal{M}_\mathcal{T}}(t_n) \cup \mathcal{B}_{\text{BR}}^{\mathcal{M}_\mathcal{T}}(t_n)$ of four square blocks:

1. **block** $\mathcal{B}_{\text{TL}}^{\mathcal{M}_\mathcal{T}}(t_n)$ is provided with vector $\overrightarrow{d_{\text{TL}}}(t_n)$ in its **top-left** corner;
2. **block** $\mathcal{B}_{\text{TR}}^{\mathcal{M}_\mathcal{T}}(t_n)$ is provided with vector $\overrightarrow{d_{\text{TR}}}(t_n)$ in its **top-right** corner;
3. **block** $\mathcal{B}_{\text{BL}}^{\mathcal{M}_\mathcal{T}}(t_n)$ is provided with vector $\overrightarrow{d_{\text{BL}}}(t_n)$ in its **bottom-left** corner;
4. **block** $\mathcal{B}_{\text{BR}}^{\mathcal{M}_\mathcal{T}}(t_n)$ is provided with vector $\overrightarrow{d_{\text{BR}}}(t_n)$ in its **bottom-right** corner.

According to the directions and the amplitudes of the four motion vectors, these blocks may be disposed in various ways. A *motion tube* in translation, for instance, will have four equal motion vectors: all four blocks $\mathcal{B}_{\text{TL}}^{\mathcal{M}_\mathcal{T}}(t_n)$, $\mathcal{B}_{\text{TR}}^{\mathcal{M}_\mathcal{T}}(t_n)$, $\mathcal{B}_{\text{BL}}^{\mathcal{M}_\mathcal{T}}(t_n)$ and $\mathcal{B}_{\text{BR}}^{\mathcal{M}_\mathcal{T}}(t_n)$ will be superimposed. Otherwise, they may be partially overlapping, thus crudely describing various shapes. This is illustrated in figure 5.2.1.

In the end, each pixel may be motion compensated from up to four contributions. In this connection, the provided motion model is highly inspired from the Barbell lifting [IM97], wherein OBMC weighting windows play the role of the Barbell weights. The forward motion compensation is illustrated in figure 5.2.1c. This modified version of the OBMC is called Overlapped Tube Motion Compensation (OTMC).

From the four motion vectors which describe the deformation of the quadrilateral patch of texture being tracked, a wide variety of deformations and displacements can be handled. In particular, translational and slight warpings were identified as a minimal set of deformations *motion tubes* should be able to undergo. In addition, chapter 4 highlighted the importance of the ability for the *motion tubes* to be connected or disconnected, such that continuities and discontinuities of the motion field could be matched. Taking previous considerations into account, following subsections will introduce two motion models:

1. the **Tube Motion Compensation (TMC)** model describes **translational** deformations which require the *motion tubes* to be **disconnected** from their neighbours.
2. the **Overlapped Tube Motion Compensation (OTMC)** model describes **more complex** deformations by keeping neighbouring *motion tubes* **connected** to each other.

5.2.2.2 A time-continuous representation of the deformation

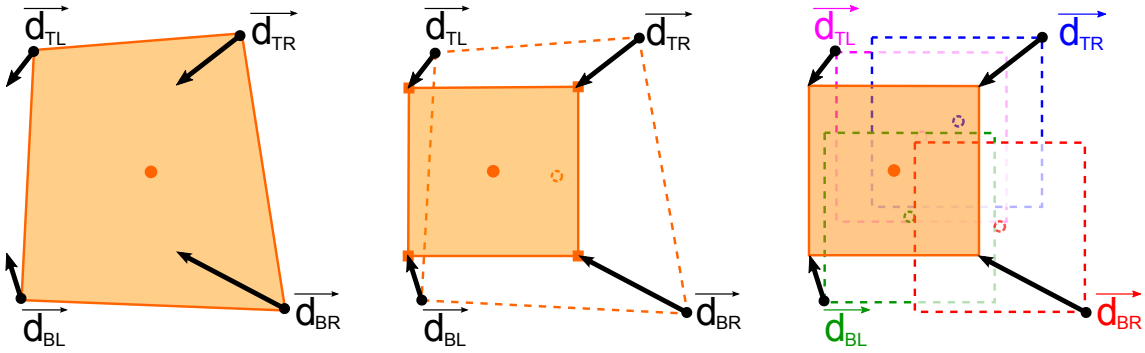
As mentioned earlier, one of the most attractive features of deformable control grids is probably their ability to describe successive deformations undergone by a patch of texture in a continuous fashion (table 5.1.1). On the other hand, block-based representations, including OBMC, generally reinitialize the motion grid at each time instant.

Motion tubes intrinsically being time-continuous objects, their deformation model should be time-continuous as well. As a consequence, equations (4.3) and (4.4) respectively require the *motion tubes* warping operators $\mathcal{W} = \{w_{i\to i+1}\}_{i0,\dots,G}$ to be invertible and the ability to be composed to each other. To this end, the both deformation modes Tube Motion Compensation (TMC) and OTMC can be inverted and successively applied to the same patch of texture.

a Inverting the warping operations

Figure 5.2.1 illustrates the forward motion compensation process: an initial block of texture is translated onto four different locations. In each pixel, the prediction is then given by a weighted sum of the different contributions. This can be interpreted as a low computational version of the CGI; this particular topic is further investigated in appendix A. To this end, the inversion of the OBMC is approximated by four translations:

- the **forward motion compensation** individually translates the blocks $\mathcal{B}_{\text{TL}}^{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$, $\mathcal{B}_{\text{TR}}^{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$, $\mathcal{B}_{\text{BL}}^{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ and $\mathcal{B}_{\text{BR}}^{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ by respective displacement vectors $\overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{BL}}}(t_{\text{cur}})$ and $\overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$.
- conversely, the **backward motion compensation** is simply obtained by applying opposite translations. Hence, it individually translates the blocks $\mathcal{B}_{\text{TL}}^{\mathcal{M}_\mathcal{T}}(t_{\text{cur}})$, $\mathcal{B}_{\text{TR}}^{\mathcal{M}_\mathcal{T}}(t_{\text{cur}})$, $\mathcal{B}_{\text{BL}}^{\mathcal{M}_\mathcal{T}}(t_{\text{cur}})$ and $\mathcal{B}_{\text{BR}}^{\mathcal{M}_\mathcal{T}}(t_{\text{cur}})$ by respective displacement vectors $-\overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$, $-\overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $-\overrightarrow{d_{\text{BL}}}(t_{\text{cur}})$ and $-\overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$, back onto their initial position. The backward motion compensation is illustrated in figure 5.2.2.



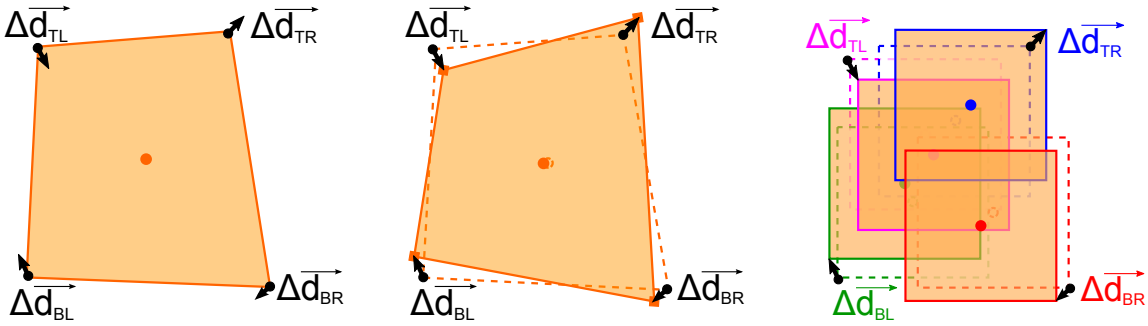
(a) *Current block* $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{cur})$ (b) *Idealistic CGI warping* (c) *OBMC-like warping approximation*

Figure 5.2.2: *Inversion of the OTMC: backward motion compensation*

b Composing the warping operations

In order to provide a time-continuous description of the motion information, warping operations need to be successively applied to an initial shape. Let $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{prev})$ be the support of a *motion tube* $\mathcal{M}_\mathcal{T}$ at time instant t_{prev} as shown in figure 5.2.3a. $\mathcal{M}_\mathcal{T}$'s support $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{cur})$ at instant t_{cur} can be described as a deformation of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{prev})$. The deformation between instants t_{prev} and t_{cur} is then given by the four differences between the motion vectors of t_{prev} and those of t_{cur}:

1. $\Delta\overrightarrow{d_{TL}}(t_{prev}, t_{cur}) = \overrightarrow{d_{TL}}(t_{cur}) - \overrightarrow{d_{TL}}(t_{prev})$
2. $\Delta\overrightarrow{d_{TR}}(t_{prev}, t_{cur}) = \overrightarrow{d_{TR}}(t_{cur}) - \overrightarrow{d_{TR}}(t_{prev})$
3. $\Delta\overrightarrow{d_{BL}}(t_{prev}, t_{cur}) = \overrightarrow{d_{BL}}(t_{cur}) - \overrightarrow{d_{BL}}(t_{prev})$
4. $\Delta\overrightarrow{d_{BR}}(t_{prev}, t_{cur}) = \overrightarrow{d_{BR}}(t_{cur}) - \overrightarrow{d_{BR}}(t_{prev})$



(a) *Previously warped block of texture* $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{prev})$ (b) *From* $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{prev})$ *to* $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{cur})$*: idealistic deformation* (c) *OBMC-like warping approximation*

Figure 5.2.3: *Applying successive deformations to a* motion tube*: composing the warping operations*

Consequently, the motion compensation of $\mathcal{M}_\mathcal{T}$ from t_{prev} to t_{cur} consists of individual translations of blocks $\mathcal{B}_{TL}^{\mathcal{M}_\mathcal{T}}(t_{cur})$, $\mathcal{B}_{TR}^{\mathcal{M}_\mathcal{T}}(t_{cur})$, $\mathcal{B}_{BL}^{\mathcal{M}_\mathcal{T}}(t_{cur})$ and $\mathcal{B}_{BR}^{\mathcal{M}_\mathcal{T}}(t_{cur})$ by respective displacement vectors $\Delta\overrightarrow{d_{TL}}(t_{prev}, t_{cur})$, $\Delta\overrightarrow{d_{TR}}(t_{prev}, t_{cur})$, $\Delta\overrightarrow{d_{BL}}(t_{prev}, t_{cur})$ and $\Delta\overrightarrow{d_{BR}}(t_{prev}, t_{cur})$. This is illustrated in figures 5.2.3b and 5.2.3c. Thus, motion vectors can be interpreted as the sum of successive relative displacements $\Delta\overrightarrow{d}(t_i, t_j)$, enabling the warping operations to be successively composed. If $t_{cur} > t_{ref}$,

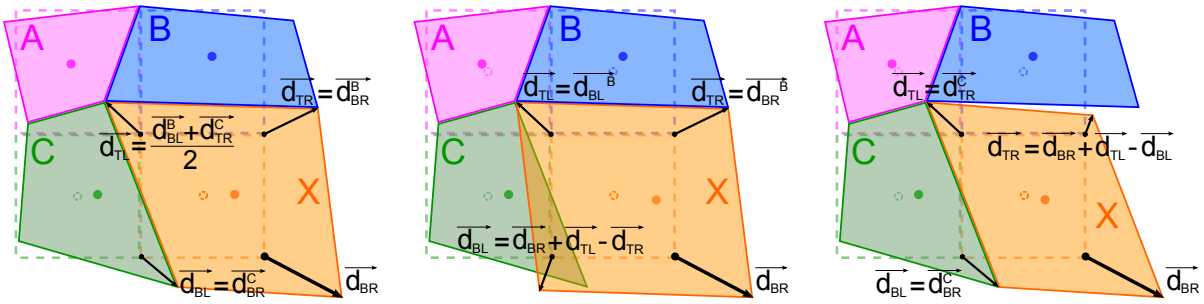
$$w_{ref\to cur} = w_{ref\to(ref+1)} \circ \ldots \circ w_{(cur-1)\to cur} \iff \begin{cases} \overrightarrow{d_{TL}} = \sum_{i=ref}^{cur-1} \Delta\overrightarrow{d_{TL}}(t_i, t_{i+1}) & , \quad \overrightarrow{d_{TR}} = \sum_{i=ref}^{cur-1} \Delta\overrightarrow{d_{TR}}(t_i, t_{i+1}) \\ \overrightarrow{d_{BL}} = \sum_{i=ref}^{cur-1} \Delta\overrightarrow{d_{BL}}(t_i, t_{i+1}) & , \quad \overrightarrow{d_{BR}} = \sum_{i=ref}^{cur-1} \Delta\overrightarrow{d_{BR}}(t_i, t_{i+1}) \end{cases} \tag{5.2}$$

Conversely, if $t_{cur} < t_{ref}$,

$$w_{ref \to cur} = w_{ref \to (ref-1)} \circ \ldots \circ w_{(cur+1) \to cur} \iff \begin{cases} \overrightarrow{d_{TL}} = \sum_{i=cur}^{ref-1} \Delta\overrightarrow{d_{TL}}(t_i, t_{i+1}) & , \quad \overrightarrow{d_{TR}} = \sum_{i=cur}^{ref-1} \Delta\overrightarrow{d_{TR}}(t_i, t_{i+1}) \\ \overrightarrow{d_{BL}} = \sum_{i=cur}^{ref-1} \Delta\overrightarrow{d_{BL}}(t_i, t_{i+1}) & , \quad \overrightarrow{d_{BR}} = \sum_{i=cur}^{ref-1} \Delta\overrightarrow{d_{BR}}(t_i, t_{i+1}) \end{cases} \tag{5.3}$$

5.2.2.3 OTMC: connected motion tubes undergoing limited deformations

In order to provide a continuous representation of the motion field, it is crucial for the *motion tubes* to be able to remain connected to each other, and for their motion model, to handle corresponding deformations (warpings). Let **X** be the current *motion tube*. Let **A**, **B** and **C** be respectively the top-left, top, and left causal neighbours of **X** (figure 5.2.4); it is assumed that their deformation has been previously estimated.



(a) *Full-connected OTMC* **(b)** *Top-connected OTMC* **(c)** *Left-connected OTMC*

Figure 5.2.4: *Idealistic representation of the deformation for the three connected modes*

The proposed OTMC model keeps **X**'s corners connected to those of its causal neighbours. In practice, however, it has been chosen to connect **X** to **B** and **C** only: vertical and horizontal connection directions are only considered. Figure 5.2.4 illustrates the OTMC motion model. In order to keep the schematics as simple as possible, the deformations are represented as idealistic CGI warpings. Three connection modes are provided:

1. a **full-connected OTMC** mode: **X**'s top-left corner is connected to the average position between **B**'s bottom-left corner and **C**'s top-right corner, **X**'s top-right corner is connected to **B**'s bottom-right corner, **X**'s bottom-left corner is connected to **C**'s bottom-right corner, and **X**'s bottom-right corner is disconnected (figure 5.2.4a);
2. a **top-connected OTMC** mode: **X**'s top corners (respectively top-left and top-right) are connected to **B**'s bottom corners (respectively bottom-left and bottom-right), and **X**'s bottom corners are disconnected. The position of **X**'s bottom-left corner is adjusted such that the overall shape of $\mathbf{\Omega}_X(t)$ is a parallelogram (figure 5.2.4b);
3. a **left-connected OTMC** mode: **X**'s left corners (respectively top-left and bottom-left) are connected to **C**'s right corners (respectively top-right and bottom-right), and **X**'s right corners are disconnected. The position of **X**'s top-right corner is adjusted such that the overall shape of $\mathbf{\Omega}_X(t)$ is a parallelogram (figure 5.2.4c).

Forcing $\mathbf{\Omega}_X(t)$'s shape to be a parallelogram, whenever a single direction of connection is used, encourages the representation of slight rotations. Whichever way, a single motion vector needs to be estimated, remaining motion vectors being retrieved from those of **B**, and **C**. From then, a large set of continuous deformations can be described. As the deformation is approximated by an OBMC-like projection, however, only simple deformations can be effectively represented.

5.2.2.4 TMC: disconnected motion tubes in translation

Despite its crudeness, the translational BMC motion model proved to be quite efficient in classical approaches to motion compensation, in terms of compression. In particular, its ability to compactly represent the discontinuities of the motion field have been largely appreciated. Similarly, it is crucial for the *motion tubes* to be able to simply translate, regardless

to the deformation of neighbouring patches of textures. In such case, a single motion vector (in practice, $\overrightarrow{d_{BR}}$) is used to describe the translation undergone by the *motion tube*, and

$$\overrightarrow{d_{TL}}(t_n) = \overrightarrow{d_{TR}}(t_n) = \overrightarrow{d_{BL}}(t_n) = \overrightarrow{d_{BR}}(t_n) \quad . \tag{5.4}$$

In figure 5.2.5, the current *motion tube* **X** is undergoing a simple translation, regardless of the nature of the deformations of neighbouring patches of texture. As can be seen, **X** is not connected to any of its causal neighbours **A**, **B**, or **C**. For now, it is assumed that **A**, **B**, and **C** are also undergoing simple translations. Such a motion model is baptized TMC and requires a single translational projection to be performed. Consequently, a single motion vector needs to be estimated and transmitted: $\overrightarrow{d_{BR}}(t_n)$. The TMC motion mode is illustrated in figure 5.2.5.

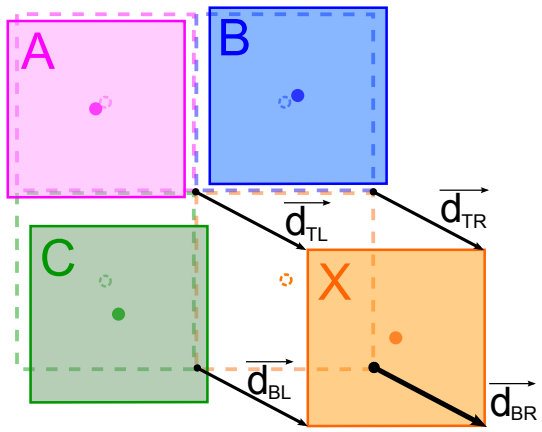


Figure 5.2.5: *A disconnected* motion tube *in translation*

5.2.2.5 Hybridizing TMC and OTMC motion models into a Switched OTMC model

Similarly to the SOBMC and other hybrid block-mesh approaches [IM97, IM00, HMCP01, CHJ+06], the final motion model hybridizes the different motion modes previously introduced. Four connection modes are now available: full connection, left connection, right connection and disconnection. Table 5.2.1 summarizes the different connection modes, along with corresponding motion vectors. The motion vectors of **B**, and **C** are respectively written $\overrightarrow{d^{B}_{corner}}(t_n)$, and $\overrightarrow{d^{C}_{corner}}(t_n)$.

Corner	Full OTMC is/isn't connected to	Top OTMC is/isn't connected to	Left OTMC is/isn't connected to	TMC is/isn't connected to
Top-Left	~ both **B** and **C** $\overrightarrow{d_{TL}} = \frac{\overrightarrow{d^{B}_{BL}} + \overrightarrow{d^{C}_{TR}}}{2}$	✓ **B**'s BL corner $\overrightarrow{d_{TL}} = \overrightarrow{d^{B}_{BL}}$	✓ **C**'s TR corner $\overrightarrow{d_{TL}} = \overrightarrow{d^{C}_{TR}}$	✗ none $\overrightarrow{d_{TL}} = \overrightarrow{d_{BR}}$
Top-Right	✓ **B**'s BR corner $\overrightarrow{d_{TR}} = \overrightarrow{d^{B}_{BR}}$	✓ **B**'s BR corner $\overrightarrow{d_{TR}} = \overrightarrow{d^{B}_{BR}}$	✗ none $\overrightarrow{d_{TR}} = \overrightarrow{d_{BR}} + \overrightarrow{d_{TL}} - \overrightarrow{d_{BL}}$	✗ none $\overrightarrow{d_{TR}} = \overrightarrow{d_{BR}}$
Bottom Left	✓ **C**'s BR corner $\overrightarrow{d_{BL}} = \overrightarrow{d^{C}_{BR}}$	✗ none $\overrightarrow{d_{BL}} = \overrightarrow{d_{BR}} + \overrightarrow{d_{TL}} - \overrightarrow{d_{TR}}$	✓ **C**'s BR corner $\overrightarrow{d_{BL}} = \overrightarrow{d^{C}_{BR}}$	✗ none $\overrightarrow{d_{BL}} = \overrightarrow{d_{BR}}$
Bottom-Right	✗ none $\overrightarrow{d_{BR}}$	✗ none $\overrightarrow{d_{BR}}$	✗ none $\overrightarrow{d_{BR}}$	✗ none $\overrightarrow{d_{BR}}$

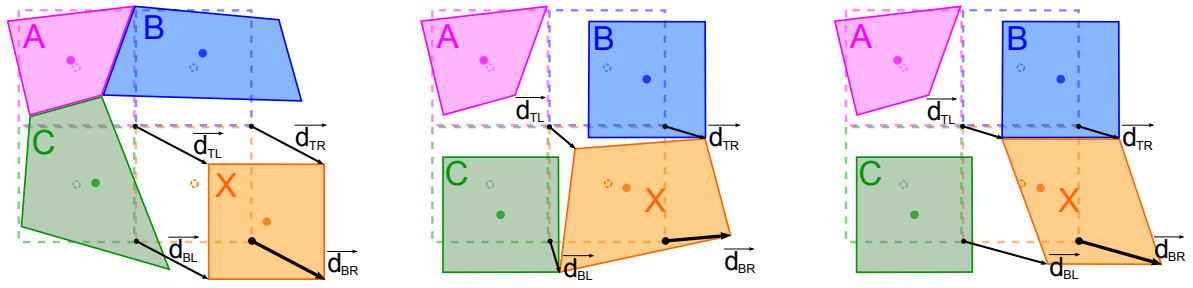
Table 5.2.1: *Available motion modes: corresponding connections and motion vectors*

Whenever the deformation parameters of the current *motion tube* X are estimated, all four motion modes are competing. In each case, only the motion vector $\overrightarrow{d_{BR}}$ is optimized in regards to the considered motion model. From the distortion ξ_{Dist} and an eventual regularity criterion ξ_{Regul}, the motion mode which minimizes the matching error is kept, such that

$$\begin{aligned}\left\{\text{mode}^*,\ \overrightarrow{d^*_{BR}}\right\} &= \underset{\text{mode},\ \overrightarrow{d_{BR}}}{\arg\min}\quad \xi_{Tube}\left(\mathcal{T}_X\ ,\ \overline{\mathcal{T}_X}\right)\\ &= \underset{\text{mode},\ \overrightarrow{d_{BR}}}{\arg\min}\quad \xi_{Dist}\left(\overrightarrow{d^{\text{mode}}_{BR}}\ ,\ \text{mode}\right)\left(\mathcal{T}_X\ ,\ \overline{\mathcal{T}_X}\right)\\ &\quad \text{subject to}\quad \xi_{Regul}\left(\overrightarrow{d^{\text{mode}}_{BR}}\ ,\ \text{mode}\right) \geq \xi^{min}_{Regul}\quad ,\end{aligned} \tag{5.5}$$

where mode^* and $\overrightarrow{d^*_{BR}}$ are respectively the optimal motion mode and the optimal bottom-right motion vector. $\mathcal{T}_X$ and $\overline{\mathcal{T}_X}$ are respectively the original patch of texture and its motion compensated prediction. ξ^{max}_{Regul} is the minimum value of the regularity criterion under which the motion field is not to considered to be regular enough.

Thus, the different motion modes can be hybrized in various ways to accurately represent the local variations of the motion field. Figure 5.2.6 illustrates their hybridization with several connection patterns which may appear. The deformations are symbolized by idealistic CGI warping to increase the schematics readability. Annex B provides additional figures which further illustrate the possibilities of the proposed motion model.



(a) *A, B and C: OTMC; X: TMC* (b) *A and X: OTMC; B and C: TMC* (c) *A: OTMC; B and C: TMC; X: top OTMC*

Figure 5.2.6: *Various hybridizations of the different motion models*

5.2.3 Towards content-adaptive variable size motion tubes

While block-based representations were proved to be a simple and efficient way to partition image sequences with an eye to compression, they often suffer from their inability to adapt to the geometry of the spatial contents. Consequently, several still image and video compression schemes using variable block size have also been proposed, including the LAR still image coder [DBBR07] and the ITU-T H.264/AVC video compression standard.

So far, *motion tubes* were initialized from a regular partition of the reference image into fixed-size blocks. Consequently, the measured motion field may not be accurate enough with regard to the images contents. It is now proposed to allow *motion tubes* to be split into sub-tubes, each of which undergoing specific displacements and deformations. A hierarchical structure, optimized though a rate-distortion approach, will then be provided, thus ensuring that the size of the *motion tubes* are fit to the images contents, while controlling the increase in motion information bit-rate.

5.2.4 A spatio-temporal regularization of the trajectories of the motion tubes

Both the deformation model and the motion estimation process should integrate some regularity constraints to encourage the *motion tubes* to undergo spatio-temporal displacements and deformations as smooth as possible. To this end, several mechanisms may be activated to improve the overall consistency of the provided spatio-temporal representation:

1. a **locally adaptive** variant of the OTMC can be used to regularize the deformations;
2. a **multigrid** approach can be used to regularize the motion field along its spatial dimensions;

3. a **rate-distortion** matching criterion can be used to evaluate a set of motion parameters. Along with an appropriate set of motion predictors, this will naturally regularize the spatio-temporal motion information. In addition, the estimation process will then be able to accurately control the amount of motion information to be transmitted;
4. finally, a **trajectory-based** prediction mechanism can be used to encourage the *motion tubes* to undergo spatio-temporal displacements and deformations as smooth as possible.

Mechanisms 1 and 4 are handled by the motion model and provides a description of the motion information which is intrinsically regularized. Mechanisms 2 and 3 are performed by the estimation process. In the end, all these mechanisms contribute to the reduction of motion discrepancies, and significantly improve the quality of the synthesized images.

5.2.5 A low-computational tube-independent motion estimation

5.2.5.1 Optimizing the backward motion compensation

As mentioned in chapter 4, projecting a set of *motion tubes* across a GOP is very likely to produce multi-connected and disconnected areas. Both these phenomenons raise critical issues regarding the motion estimation process.

1. How are **disconnected areas** supposed to be taken into account by the estimation? Should their occurrences be minimized, or should they simply not be taken into account?
2. In **multi-connected areas**, how each contribution should be assessed in regards to the other ones? Should the estimation limit such phenomenons?

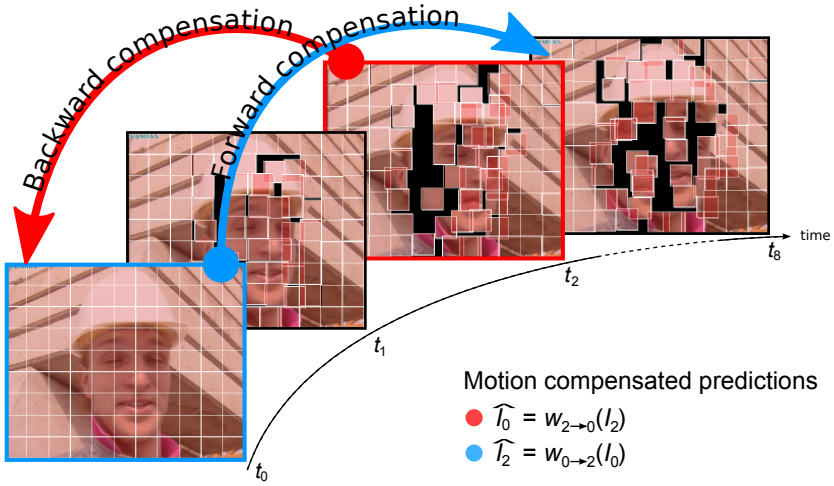


Figure 5.2.7: *Backward and forward motion compensation within* motion tube *paradigm*

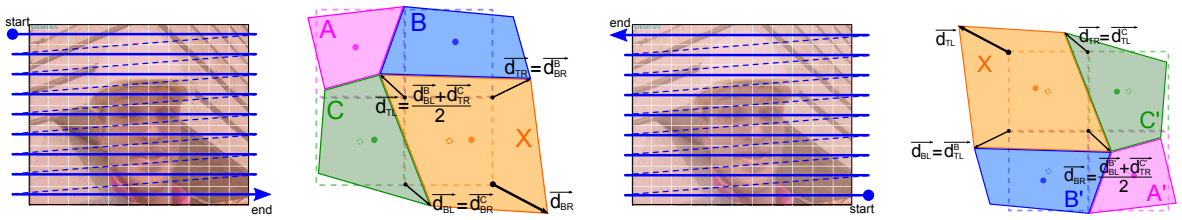
From these questions, it clearly appears that directly optimizing the synthesized images (*i.e.* the *forward* motion compensated images) is a very delicate task. To overcome this issue, it is naturally proposed to optimize the *backward* projection $\widehat{I_{ref}}$ of the current image I_{cur} on the reference image I_{ref}. Indeed, the reference image being split into disjoint blocks (as explained in section 5.2.1), the *motion tubes* are not overlapping and do not introduce neither multi-connected nor disconnected areas. As a consequence, both *backward* and *forward* motion compensation steps will be required (figure 5.2.7):

1. several **backward motion compensation** steps will be performed by the **motion estimation** to optimize $\widehat{I_{ref}}$ from I_{cur};
2. a **forward motion compensation** step will be performed to generate the final **synthesized image** $\widehat{I_{cur}}$ from I_{ref}, by inverting the estimated backward motion parameters.

5.2.5.2 Processing the motion tubes following a raster scan order

Despite the fact that *motion tubes* are not overlapping at the reference instant, their optimization shall take into account the interdependency which results from their ability to be connected. Indeed, it was seen in section 5.2.2.5 that connected *motion tubes* inherit some of their motion vectors from their neighbours.

In order to take this interdependency into account, it is proposed to use a sub-optimal estimation process which simply processes the *motion tubes* following a causal raster-scan order illustrated in figure 5.2.8a. This is why, in section 5.2.2.5, it was proposed to connect the current *motion tube* **X** to its causal neighbours (in practice, **B** and **C**) only: their deformation is already known when processing **X**. The causal connection model is illustrated in figure 5.2.8b.



(a) *Causal motion estimation step*

(b) *Causal connection with top-left, top and left neighbours*

(c) *Anticausal motion estimation step*

(d) *Anticausal connection with right, bottom and bottom-right neighbours*

Figure 5.2.8: *Causal and anticausal motion estimation: effect on the connection directions*

Such an approach guarantees causal connections to be optimal in regards to the considered neighbourhood (*motion tubes* **B** and **C**). However, this is not optimal when considering all the conceivable connection scenarios. In order to improve this sub-optimal approach to motion estimation, it is proposed to perform the estimation in two steps:

1. an **anticausal** motion estimation step following a **reversed raster scan** order (figure 5.2.8c);
2. a **causal** motion estimation step following a **raster scan** order (figure 5.2.8a);

The anticausal step requires the connection patterns to be put upside-down: the current *motion tube* **X** can now be connected to its anticausal neighbours **B'** (bottom) and **C'** (right), $\mathcal{M}_{\mathcal{T}_B}$ (bottom) and $\mathcal{M}_{\mathcal{T}_R}$ (right). The modified connection pattern is shown in figure 5.2.8d. Consequently, the motion estimation process now considers both causal and anticausal directions of connection to estimate the displacement and the deformations of the *motion tubes*. In the end, however, the causal connection model is used to predict and encode the motion information. Though complexity is not our main concern within the scope of this study, future experiments will show that the increase in complexity is compensated by a significantly better motion estimation.

5.3 Warping motion tubes across time: in between block-based and mesh-based model

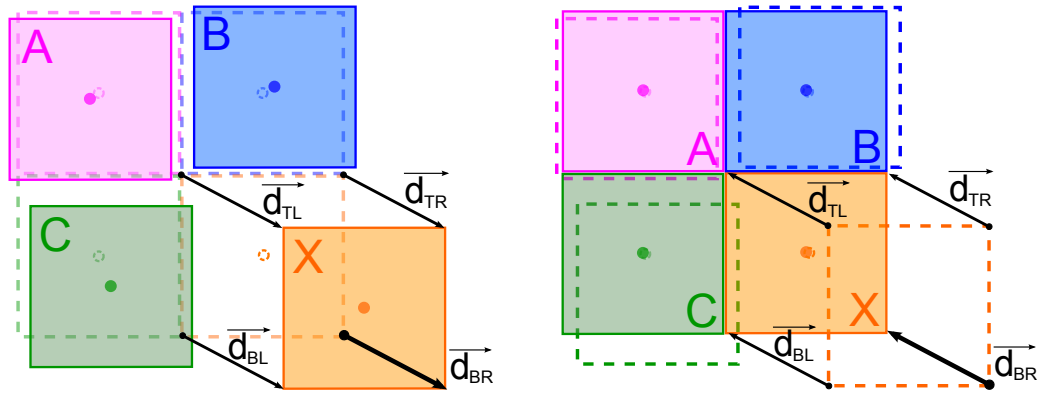
Previous section introduced four different motion models: the disconnected translational TMC model, and the three connected deformation models (full, top and left OTMC). This section will detail how forward and backward motion compensations are performed in each case. In addition, it will compare the respective performances of each motion model and show that their hybridization ends up into a simple and effective motion model.

5.3.1 Disconnected motion tubes: the translating TMC mode

Let $\{I_0, \ldots, I_G\}$ be a GOP of G images. Let **X** be a *motion tube* belonging to the family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ defined in section 5.2.1: each *motion tube* is initialized from a block of texture sourced in image $I_{\text{ref}} = I_0$. Let **A**, **B** and **C** respectively be its top-left, top and right causal neighbours. With the TMC motion mode, **X** is simply undergoing a translation $\overrightarrow{d} = [dx, dy]^T$, such that

$$\overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) = \overrightarrow{d_{\text{TR}}}(t_{\text{cur}}) = \overrightarrow{d_{\text{BL}}}(t_{\text{cur}}) = \overrightarrow{d_{\text{BR}}}(t_{\text{cur}}) = \overrightarrow{d} \tag{5.6}$$

As a consequence, all four blocks $\mathcal{B}^{\text{X}}_{\text{TL}}(t_{\text{cur}})$, $\mathcal{B}^{\text{X}}_{\text{TR}}(t_{\text{cur}})$, $\mathcal{B}^{\text{X}}_{\text{BL}}(t_{\text{cur}})$ and $\mathcal{B}^{\text{X}}_{\text{BR}}(t_{\text{cur}})$ are superimposed: a single translation operation is required to perform both forward and backward motion compensation operations.



(a) *Forward motion compensation* **(b)** *Backward motion compensation*

Figure 5.3.1: *TMC motion mode: a disconnected* motion tube *translating by vector* $\overrightarrow{d}$

5.3.1.1 Forward motion compensation

At time instant t_{cur}, the *forward* motion compensation translates the reference texture $\mathcal{T}$ onto $\mathcal{M}_{\mathcal{T}}$'s support $\mathbf{\Omega}_X(t_{cur})$. $\mathbf{\Omega}_X(t_{cur})$ is obtained by translating $\mathbf{\Omega}_X(t_{ref})$

$$\mathbf{\Omega}_X(t_{cur}) = t_{\overrightarrow{d}}(\mathbf{\Omega}_X(t_{ref})) \tag{5.7}$$

where $t_{\overrightarrow{d}}$ is the translation operator by vector $\overrightarrow{d}$. As a consequence, the synthesized texture $\overline{\mathcal{T}}(t_{cur})$ is simply given by the original patch of texture (figure 5.3.1a). In practice, it comes down to copy the collocated texture from the reference image I_{ref}, and $\forall (x,y) \in \mathbf{\Omega}_X(t_{cur})$:

$$\overline{\mathcal{T}}(t_{cur}) = I_{ref}(\mathbf{\Omega}_X(t_{ref})) \quad \Longleftrightarrow \quad \overline{\mathcal{T}}(t_{cur})(x,y) = I_{t_{ref}}(x+dx, y+dy) \tag{5.8}$$

5.3.1.2 Backward motion compensation

Conversely, the *backward* motion compensation translates the current texture $\mathcal{T}$ onto $\mathbf{\Omega}_X(t_{ref})$, support of $\mathcal{M}_{\mathcal{T}}$ at time instant t_{ref}. $\mathbf{\Omega}_X(t_{ref})$ is obtained by translating $\mathbf{\Omega}_X(t_{cur})$ by $\overrightarrow{d}$'s opposite vector, and:

$$\mathbf{\Omega}_X(t_{ref}) = t_{-\overrightarrow{d}}(\mathbf{\Omega}_X(t_{cur})) \tag{5.9}$$

As a consequence, the texture synthesized at instant t_{ref}, $\overline{\mathcal{T}}(t_{ref})$, is given by the patch of texture registered by the *motion tube* $\mathcal{M}_{\mathcal{T}}$ at the current instant t_{cur} (figure 5.3.1b). In practice, it comes down to copy the collocated texture from the current image I_{cur}. $\forall (x,y) \in \mathbf{\Omega}_X(t_{ref})$

$$\overline{\mathcal{T}}(t_{ref}) = I_{cur}(\mathbf{\Omega}_X(t_{cur})) \quad \Longleftrightarrow \quad \overline{\mathcal{T}}(t_{ref})(x,y) = I_{cur}(x-dx, y-dy) \tag{5.10}$$

5.3.1.3 Building the matching criterion

In TMC motion mode, the matching criterion $\xi_{\text{Dist-TMC}}$ of a *motion tube* is given by the distance between the reference texture $\mathcal{T}(t_{ref})$ and the synthesized version of the reference texture $\overline{\mathcal{T}}(t_{ref})$, predicted from the current image I_{cur}. Both MAE and MSE have been considered, and $\xi_{\text{Dist-TMC}}$ may write as

$$\begin{aligned}\xi_{\text{Dist-TMC}}\left(\mathcal{T}(t_{ref}),\ \overline{\mathcal{T}}(t_{ref})\right) &= \xi_{\text{Dist-TMC}}\Big(\underbrace{I_{ref}(\mathbf{\Omega}_X(t_{ref}))}_{\text{original}},\ \underbrace{I_{cur}(\mathbf{\Omega}_X(t_{cur}))}_{\text{synthesized}}\Big) \\ &= \begin{cases} \dfrac{1}{MN} \cdot \displaystyle\sum_{(x,y)\in\mathbf{\Omega}_X(t_{ref})} \left| I_{ref}(x,y) - I_{cur}(x-dx_1, y-dy_1) \right| & \text{with the MAE} \\ \dfrac{1}{MN} \cdot \displaystyle\sum_{(x,y)\in\mathbf{\Omega}_X(t_{ref})} \left(I_{ref}(x,y) - I_{cur}(x-dx_1, y-dy_1) \right)^2 & \text{with the MSE} \end{cases}\end{aligned} \tag{5.11}$$

where M and N are respectively the width and height, in pixels, of the patch of texture at the reference instant t_{ref}.

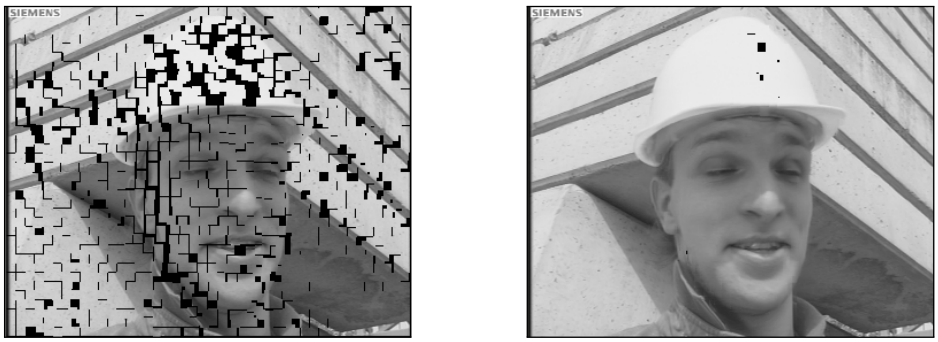
5.3.2 Towards more complex deformations: OTMC motion mode

The OTMC motion mode accounts for both motion field continuities and geometrical deformations. In particular, it relies on the overlapping features of the OBMC. In this regard, our motivation is twofold:

- the OBMC acts as a **simplified** version of the **CGI** (see appendix A);
- the OBMC significantly **reduces** the amount of **blocking artefacts** from which the TMC motion mode may suffer.

5.3.2.1 A preliminary observation: how overlapping the motion can significantly improve the representation

As a preliminary observation, figure 5.3.2 shows how much overlapped *motion tubes* can improve the representation. In figure 5.3.2a, the reference image is partitioned into disjoint tubes. In figure 5.3.2a, it is partitioned into overlapping tubes twice as large as those of the first case. A weighting window is used to smooth the transition between the tubes. Figure 5.3.2 speak for itself: overlapping the *motion tubes* increases the coherence of the reconstruction. Blocking and staircase effects are reduced, at the expense of a notable blurring effect: overlapping should only be used when it effectively improves the reconstruction.



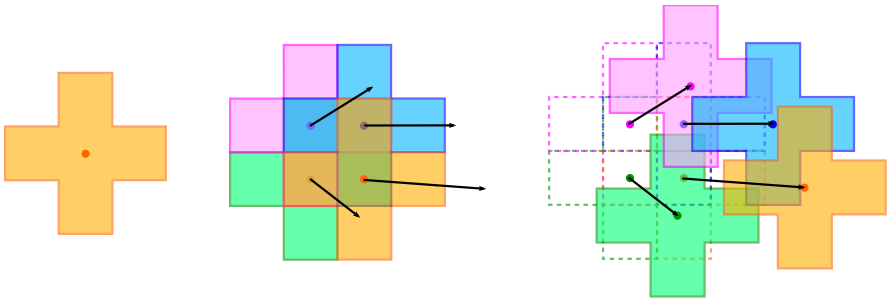
(a) *Non-overlapping* motion tubes **(b)** *Overlapping* motion tubes

Figure 5.3.2: *How overlapping the textures of the* motion tubes *may improve the synthesized textures*

5.3.2.2 Towards a modified OBMC

a Classical conception of the OBMC

Typically, OBMC partitions the image plane into overlapping patterns. A weighting window is used to play down the importance of the contributions brought by the patterns periphery. This smoothes the motion field and avoids sharp motion field transitions from one pattern to the next. In ITU-T H.263 video compression standard, for instance, a cross-shaped pattern is employed (figure 5.3.3). As it performs a *backward* motion compensation, figure 5.3.3c is projected into figure 5.3.3b. Conversely, *motion tubes* may perform the inverse operation, and project figure 5.3.3c into figure 5.3.3b.



(a) *Cross pattern* **(b)** *Initially overlapping crosses* **(c)** *Resulting forward motion compensation*

Figure 5.3.3: *Typical understanding of OBMC: overlapping basis patterns in ITU-T H.263 standard*

b How OTMC can be interpreted as special case of OBMC

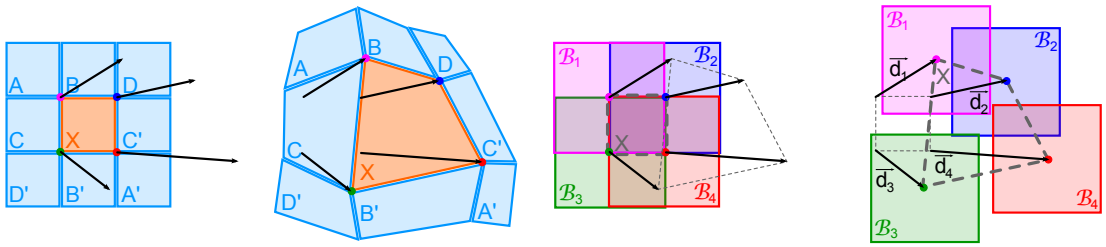
Section 5.2.2.1 explained that the proposed motion model relies on the translation of four blocks $\mathcal{B}^{X}_{TL}$, $\mathcal{B}^{X}_{TR}$, $\mathcal{B}^{X}_{BL}$ and $\mathcal{B}^{X}_{BR}$ by respective vectors $\overrightarrow{d_{TL}}$, $\overrightarrow{d_{TR}}$, $\overrightarrow{d_{BL}}$ and $\overrightarrow{d_{BR}}$. With the OTMC motion mode, it was provided the ability to connect a *motion tube* **X** to its top and left neighbours **B** and **C**. By cascading the connections, it is possible to connect as many *motion tubes* as required. Let **A**, **B**, **C**, **D**, **X**, **A'**, **B'**, **C'** and **D'** be a set of nine adjacent *motion tubes* which are kept connected to each other. Figures 5.3.4a and 5.3.4b respectively illustrate such a set of *motion tubes* and their connected deformation. **A**, **B**, **C** and **X**, in particular are respectively connected by their bottom-right, bottom-left, top-right and top-left corners. Corresponding motion vectors are equal and:

$$\overrightarrow{d^{A}_{BR}} = \overrightarrow{d^{B}_{BL}} = \overrightarrow{d^{C}_{TR}} = \overrightarrow{d^{X}_{TL}} = \overrightarrow{d_1} \tag{5.12}$$

Corresponding blocks, respectively $\mathcal{B}^{A}_{BR}$, $\mathcal{B}^{B}_{BL}$, $\mathcal{B}^{C}_{TR}$ and $\mathcal{B}^{X}_{TL}$ are adjacent and undergoing the same translation $\overrightarrow{d}$. The super-block $\mathcal{B}_1 = \mathcal{B}^{A}_{BR} \cup \mathcal{B}^{B}_{BL} \cup \mathcal{B}^{C}_{TR} \cup \mathcal{B}^{X}_{TL}$ can then be considered as a single pattern in translation by vector $\overrightarrow{d_1}$ (figures 5.3.4c and 5.3.4d). Just like the OBMC, it is now proposed to provide this super-block with a weighting window $W_1(x, y)$ centred around **X**'s top-left corner (*i.e* the centre of the super-block). Following the same idea, and under similar assumptions, three other super-blocks can be extracted:

- from **X**'s **top-right corner**, super-block $\mathcal{B}_2 = \mathcal{B}^{B}_{BR} \cup \mathcal{B}^{D}_{BL} \cup \mathcal{B}^{X}_{TR} \cup \mathcal{B}^{C'}_{TL}$ is translating by vector $\overrightarrow{d_2} = \overrightarrow{d^{B}_{BR}} = \overrightarrow{d^{D}_{BL}} = \overrightarrow{d^{X}_{TR}} = \overrightarrow{d^{C'}_{TL}}$; it is provided with a weighting window $W_2(x, y)$ centred around **X**'s top-right corner;
- from **X**'s **bottom-left corner**, super-block $\mathcal{B}_3 = \mathcal{B}^{C}_{BR} \cup \mathcal{B}^{X}_{BL} \cup \mathcal{B}^{D'}_{TR} \cup \mathcal{B}^{B}_{TL}$ is translating by vector $\overrightarrow{d_3} = \overrightarrow{d^{C}_{BR}} = \overrightarrow{d^{X}_{BL}} = \overrightarrow{d^{D'}_{TR}} = \overrightarrow{d^{B}_{TL}}$; it is provided with a weighting window $W_3(x, y)$ centred around **X**'s bottom-left corner;
- from **X**'s **top-right corner**, super-block $\mathcal{B}_4 = \mathcal{B}^{X}_{BR} \cup \mathcal{B}^{C'}_{BL} \cup \mathcal{B}^{B'}_{TR} \cup \mathcal{B}^{A'}_{TL}$ is translating by vector $\overrightarrow{d_4} = \overrightarrow{d^{X}_{BR}} = \overrightarrow{d^{C'}_{BL}} = \overrightarrow{d^{B'}_{TR}} = \overrightarrow{d^{A'}_{3}}$; it is provided with a weighting window $W_4(x, y)$ centred around **X**'s bottom-right corner.

These four super-blocks are overlapping, weighted and undergoing individual translations $\overrightarrow{d_1}$, $\overrightarrow{d_2}$, $\overrightarrow{d_3}$ and $\overrightarrow{d_4}$ (figures 5.3.4c and 5.3.4d). Hence, they can be interpreted as classic square OBMC overlapping patterns: OTMC is thus a special case of OBMC which relies on square overlapping patterns. However, OTMC and TMC motion modes have the additional ability to control whether to use one, two, three or all four quadrants of the super-blocks to motion compensation the *motion tubes*. As such, the proposed motion model uses locally shape-adaptive overlapping patterns.



(a) *A set of nine adjacent and connected* motion tubes ... **(b)** ... *and their deformation* **(c)** *OBMC super-blocks overlapped on* motion tube ***X*** **(d)** *Translated super-blocks*

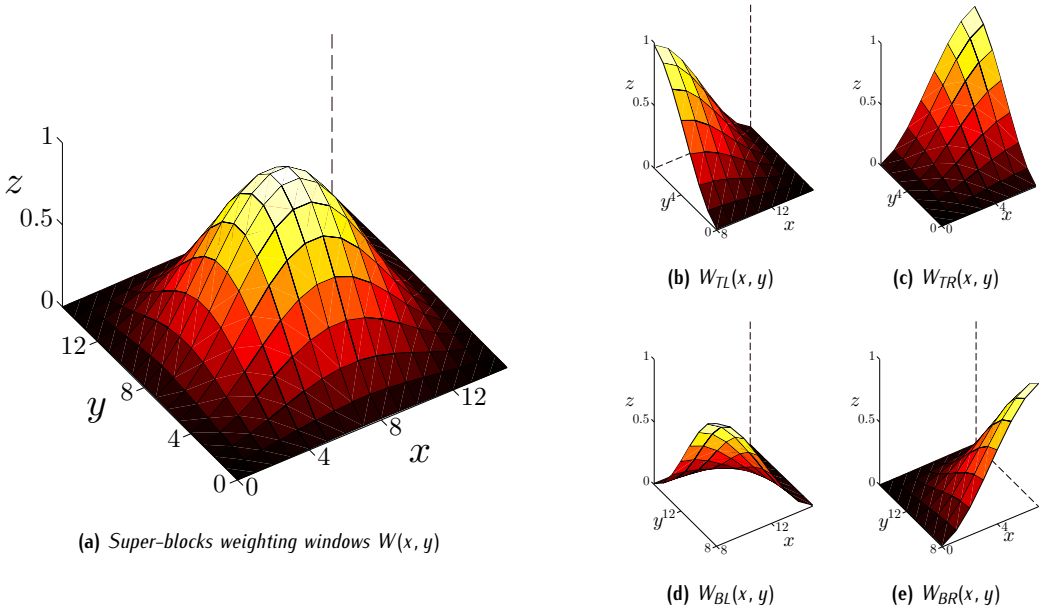
Figure 5.3.4: *How OTMC can be interpreted as OBMC: equivalent super-blocks*

c Building a set of appropriate OBMC weighting windows

The identification of the OTMC to the OBMC led to the construction of four super-blocks overlapping on the current *motion tube* **X** and its adjacent neighbours. Each of these super-blocks were weighted using a window centred on its centre, playing down the importance of its periphery. Such a weighting window is shown in figure 5.3.5a.

In particular, the four blocks $\mathcal{B}^{X}_{TL}$, $\mathcal{B}^{X}_{TR}$, $\mathcal{B}^{X}_{BL}$ and $\mathcal{B}^{X}_{BR}$ respectively correspond to a single quadrant of super-blocks $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ and $\mathcal{B}_4$. As a consequence, they will be weighted by corresponding quadrants of weighting windows $W_1(x, y)$, $W_2(x, y)$, $W_3(x, y)$, $W_4(x, y)$.

- $\mathcal{B}^{X}_{TL}$ is $\mathcal{B}_1$'s bottom-right quadrant; it will be weighted by $W_{TL}(x,y)$, $W_1(x,y)$'s bottom-right quadrant.(fig. 5.3.5b)
- $\mathcal{B}^{X}_{TR}$ is $\mathcal{B}_2$'s bottom-left quadrant; it will be weighted by $W_{TR}(x,y)$, $W_2(x,y)$'s bottom-left quadrant. (fig. 5.3.5c)
- $\mathcal{B}^{X}_{BL}$ is $\mathcal{B}_3$'s top-right quadrant; it will be weighted by $W_{BL}(x,y)$, $W_3(x,y)$'s top-right quadrant. (fig. 5.3.5d)
- $\mathcal{B}^{X}_{BR}$ is $\mathcal{B}_4$'s top-left quadrant; it will be weighted by $W_{BR}(x,y)$, $W_4(x,y)$'s top-left quadrant. (fig. 5.3.5e)



(a) *Super-blocks weighting windows $W(x,y)$*

(b) $W_{TL}(x,y)$ (c) $W_{TR}(x,y)$

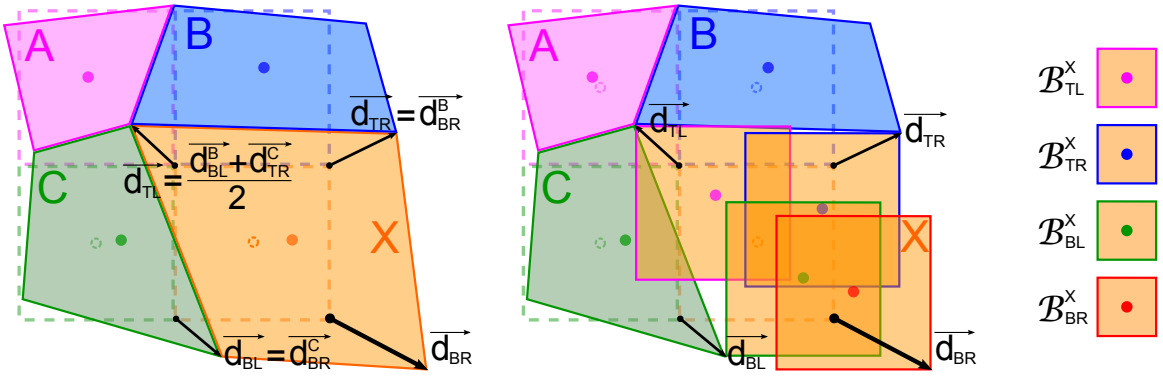
(d) $W_{BL}(x,y)$ (e) $W_{BR}(x,y)$

Figure 5.3.5: *Partition of the OBMC weighting window into quadrants*

Provided that the super-blocks are motion compensated using the same symmetric weighting window $W(x,y)$ (as shown in figure 5.3.5a), successively centred around each of their centre, the summation of the four weights, in each position (x,y) from X's support is constant and can be normalized:

$$\forall (x,y) \in \mathbf{\Omega}_X(t)\,, \quad W_{TL}(x,y) + W_{TR}(x,y) + W_{BL}(x,y) + W_{BR}(x,y) = 1\,. \tag{5.13}$$

5.3.2.3 Forward motion compensation



(a) *Idealistic CGI forward motion compensation*

(b) *OBMC-like approximation of the deformation*

Figure 5.3.6: *OTMC motion mode: forward motion compensation*

With OTMC, X's top-left, top-right and bottom left corners are kept connected to its top and left causal neighbours B and C. At time instant t_{cur}, the *forward* motion compensation translates the reference texture $\mathcal{T}$ onto four locations. Indeed,

$\mathbf{\Omega}_X(t_{cur})$ is given by the union of the translations of blocks $\mathcal{B}^X_{TL}(t_{ref})$, $\mathcal{B}^X_{TR}(t_{ref})$, $\mathcal{B}^X_{BL}(t_{ref})$, and $\mathcal{B}^X_{BR}(t_{ref})$ by respective vectors $\overrightarrow{d_{TL}}(t_{cur})$, $\overrightarrow{d_{TR}}(t_{cur})$, $\overrightarrow{d_{BL}}(t_{cur})$ and $\overrightarrow{d_{BR}}(t_{cur})$. To be more precise,

$$\begin{aligned}\mathbf{\Omega}_X(t_{cur}) =& \mathcal{B}^X_{TL}(t_{cur}) \cup \mathcal{B}^X_{TL}(t_{cur}) \cup \mathcal{B}^X_{TL}(t_{cur}) \cup \mathcal{B}^X_{TL}(t_{cur}) \\ =& t_{\overrightarrow{d_{TL}}(t_{cur})}\left(\mathcal{B}^X_{TL}(t_{ref})\right) \cup t_{\overrightarrow{d_{TR}}(t_{cur})}\left(\mathcal{B}^X_{TL}(t_{ref})\right) \cup t_{\overrightarrow{d_{BL}}(t_{cur})}\left(\mathcal{B}^X_{TL}(t_{ref})\right) \cup t_{\overrightarrow{d_{BR}}(t_{cur})}\left(\mathcal{B}^X_{TL}(t_{ref})\right) ,\end{aligned} \tag{5.14}$$

where $t_{\overrightarrow{d}}$ is the translation operator by vector $\overrightarrow{d}$. From OTMC's connection pattern (see table 5.2.1), three out of the four motion vectors are sourced from neighbouring *motion tubes* **B** and **C**, and:

- the **top-left** motion vector $\overrightarrow{d_{TL}}(t_{cur})$ is the mean of **B**'s bottom-left and **C**'s top-right vectors $\overrightarrow{d^B_{BL}}(t_{cur})$ and $\overrightarrow{d^C_{TR}}(t_{cur})$;
- the **top-right** motion vector $\overrightarrow{d_{TR}}(t_{cur})$ is given by **B**'s bottom-right motion vector $\overrightarrow{d^B_{BR}}(t_{cur})$;
- the **bottom-left** motion vector $\overrightarrow{d_{BL}}(t_{cur})$ is given by **C**'s bottom-right motion vector $\overrightarrow{d^C_{BR}}(t_{cur})$;
- the **bottom-right** motion vector $\overrightarrow{d_{BR}}(t_{cur})$ is specific to **X**.

$$\begin{cases} \overrightarrow{d_{TL}}(t_{cur}) = [dx_{TL}, dy_{TL}]^T = \dfrac{\overrightarrow{d^B_{BL}}(t_{cur}) + \overrightarrow{d^C_{TR}}(t_{cur})}{2} & , \quad \overrightarrow{d_{TR}}(t_{cur}) = [dx_{TR}, dy_{TR}]^T = \overrightarrow{d^B_{BR}}(t_{cur}) \\ \overrightarrow{d_{BL}}(t_{cur}) = [dx_{BL}, dy_{BL}]^T = \overrightarrow{d^C_{BR}}(t_{cur}) & , \quad \overrightarrow{d_{BR}}(t_{cur}) = [dx_{BR}, dy_{BR}]^T \end{cases} \tag{5.15}$$

As a consequence, the synthesized texture $\overline{\mathcal{T}}^*(t_{cur})$ results from a weighted sum of a variable number of contributions (from one to four). $\forall (x,y) \in \mathbf{\Omega}(t_{cur})$, $\overline{\mathcal{T}}^*(t_{cur})$ is given by

$$\begin{aligned}\overline{\mathcal{T}}^*(t_{cur})(x,y) = & \quad \mathbf{1}_{\mathcal{B}^X_{TL}(t_{cur})}(x,y) \cdot W_{TL}\left(x - x_{ref} - dx_{TL} \ , \ y - y_{ref} - dy_{TL}\right) \cdot I_{ref}\left(x - dx_{TL} \ , \ y - dy_{TL}\right) \\ & + \mathbf{1}_{\mathcal{B}^X_{TR}(t_{cur})}(x,y) \cdot W_{TR}\left(x - x_{ref} - dx_{TR} \ , \ y - y_{ref} - dy_{TR}\right) \cdot I_{ref}\left(x - dx_{TR} \ , \ y - dy_{TR}\right) \\ & + \mathbf{1}_{\mathcal{B}^X_{BL}(t_{cur})}(x,y) \cdot W_{BL}\left(x - x_{ref} - dx_{BL} \ , \ y - y_{ref} - dy_{BL}\right) \cdot I_{ref}\left(x - dx_{BL} \ , \ y - dy_{BL}\right) \\ & + \mathbf{1}_{\mathcal{B}^X_{BR}(t_{cur})}(x,y) \cdot W_{BR}\left(x - x_{ref} - dx_{BR} \ , \ y - y_{ref} - dy_{BR}\right) \cdot I_{ref}\left(x - dx_{BR} \ , \ y - dy_{BR}\right)\end{aligned} \tag{5.16}$$

where (x_{ref}, y_{ref}) are the coordinates of the top-left corner of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_0)$. Hence, $(x - x_{ref} - dx \ , \ y - y_{ref} - dy)$ are the relative coordinates of position (x,y) into $t_{\overrightarrow{d}}\left(\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{ref})\right)$, translation of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{ref})$ by vector $\overrightarrow{d} = [dx, dy]^T$. $\mathbf{1}_\mathbf{\Omega}(x,y)$ is the indicator function defined by

$$\mathbf{1}_\mathbf{\Omega}(x,y) = \begin{cases} 1 & \text{if} \quad (x,y) \in \mathbf{\Omega} \\ 0 & \text{elsewhere} \end{cases} \tag{5.17}$$

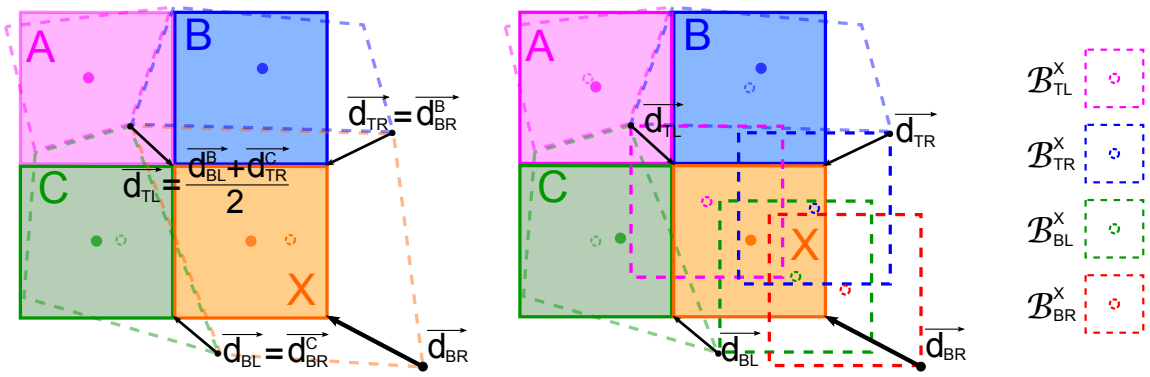
As the weights sum may vary from one position to another, the synthesized texture $\overline{\mathcal{T}}^*(t_{cur})$ is not normalized with respect to the OBMC weighting windows. Consequently, $\overline{\mathcal{T}}^*(t_{cur})$ is normalized into $\overline{\mathcal{T}}(t_{cur})$ by the following computation: $\forall (x,y) \in \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_1)$, it is given by

$$\overline{\mathcal{T}}(t_1)(x,y) = \frac{\overline{\mathcal{T}}^*(t_1)(x,y)}{N(x,y)} \tag{5.18}$$

where $N(x,y)$ is the normalization factor

$$\begin{aligned}N(x,y) = & \quad \mathbf{1}_{\mathcal{B}^X_{TL}(t_{cur})}(x,y) \cdot W_{TL}\left(x - x_{ref} - dx_{TL} \ , \ y - y_{ref} - dy_{TL}\right) \\ & + \mathbf{1}_{\mathcal{B}^X_{TR}(t_{cur})}(x,y) \cdot W_{TR}\left(x - x_{ref} - dx_{TR} \ , \ y - y_{ref} - dy_{TR}\right) \\ & + \mathbf{1}_{\mathcal{B}^X_{BL}(t_{cur})}(x,y) \cdot W_{BL}\left(x - x_{ref} - dx_{BL} \ , \ y - y_{ref} - dy_{BL}\right) \\ & + \mathbf{1}_{\mathcal{B}^X_{BR}(t_{cur})}(x,y) \cdot W_{BR}\left(x - x_{ref} - dx_{BR} \ , \ y - y_{ref} - dy_{BR}\right)\end{aligned} \tag{5.19}$$

The normalization is often source for additional complexity; however, it is only required by the forward motion compensation. Indeed, *backward* motion compensation will not require any normalization step. In practice, the normalization step is performed once all the *motion tubes* have been motion compensated, since several *motion tubes* may also overlap. In the end, OTMC motion compensation is rather similar to the simple TMC motion compensation from an implementation perspective: it simply consists in a series of classic block-based weighted projections.



(a) *Idealistic CGI backward motion compensation* **(b)** *OBMC-like approximation of the deformation*

Figure 5.3.7: *OTMC motion mode: backward motion compensation*

5.3.2.4 Backward motion compensation

Conversely, the *backward* motion compensation translates four blocks of texture of the current image I_{cur} - $\mathcal{B}^X_{TL}(t_{cur})$, $\mathcal{B}^X_{TR}(t_{cur})$, $\mathcal{B}^X_{BL}(t_{cur})$ and $\mathcal{B}^X_{BR}(t_{cur})$ - onto a single location $\mathbf{\Omega}_X(t_{ref})$, by respective vectors $-\overrightarrow{d_{TL}}(t_{cur})$, $-\overrightarrow{d_{TR}}(t_{cur})$, $-\overrightarrow{d_{BL}}(t_{cur})$ and $-\overrightarrow{d_{BR}}(t_{cur})$. At time instant t_{ref}, the patch of texture $\overline{\mathcal{T}}(t_{ref})$ synthesized from the original textural information found in X's support $\mathbf{\Omega}_X(t_{cur})$ at instant t_{cur} is then given by

$$\begin{aligned}\overline{\mathcal{T}}(t_{ref})(x,y) = \quad & W_{TL}(x-x_{ref}, y-y_{ref}) \cdot I_{cur}(x-dx_{TL}, y-dy_{TL}) \\ + \quad & W_{TR}(x-x_{ref}, y-y_{ref}) \cdot I_{cur}(x-dx_{TR}, y-dy_{TR}) \\ + \quad & W_{BL}(x-x_{ref}, y-y_{ref}) \cdot I_{cur}(x-dx_{BL}, y-dy_{BL}) \\ + \quad & W_{BR}(x-x_{ref}, y-y_{ref}) \cdot I_{cur}(x-dx_{BR}, y-dy_{BR})\end{aligned}$$

$\forall (x,y) \in \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_0)$. Again, (x_{ref}, y_{ref}) are the coordinates of the top-left corner of $\mathbf{\Omega}_X(t_{ref})$. As weighting coefficients are normalized - see equation (5.13) -, no normalization step is required, which spares the estimation loop from a large number of computations.

5.3.2.5 Building the matching criterion

Similarly to the TMC motion mode and its matching criterion introduced in section 5.3.2.5 - see equation (5.11) -, the OTMC is evaluated by a matching criterion which computes a pixel-based distance between the original reference texture $\mathcal{T}(t_{ref})$ and its reconstruction $\overline{\mathcal{T}}(t_{ref})$. Again, both MAE and MSE can be used. With the MAE, in particular, the matching criterion $\xi_{\text{Dist-OTMC}}$ can be expressed as

$$\begin{aligned}\xi_{\text{OTMC-MAE}}\left(\mathcal{T}(t_0)\ ,\ \overline{\mathcal{T}}(t_0)\right) = & \frac{1}{MN} \cdot \sum_{(x,y)\in\mathcal{B}^X_{TL}(t_{cur})} W_{TL}(x-x_{ref}, y-y_{ref}) \cdot \left| I_{ref}(x,y) - I_1(x+dx_{TL}, y+dy_{TL}) \right| \\ + & \frac{1}{MN} \cdot \sum_{(x,y)\in\mathcal{B}^X_{TR}(t_{cur})} W_{TR}(x-x_{ref}, y-y_{ref}) \cdot \left| I_{ref}(x,y) - I_1(x+dx_{TR}, y+dy_{TR}) \right| \\ + & \frac{1}{MN} \cdot \sum_{(x,y)\in\mathcal{B}^X_{BL}(t_{cur})} W_{BL}(x-x_{ref}, y-y_{ref}) \cdot \left| I_{ref}(x,y) - I_1(x+dx_{BL}, y+dy_{BL}) \right| \quad (5.20) \\ & \frac{1}{MN} \cdot \sum_{(x,y)\in\mathcal{B}^X_{BR}(t_{cur})} W_{BR}(x-x_{ref}, y-y_{ref}) \cdot \left| I_{ref}(x,y) - I_1(x+dx_{BR}, y+dy_{BR}) \right| \quad (5.21)\end{aligned}$$

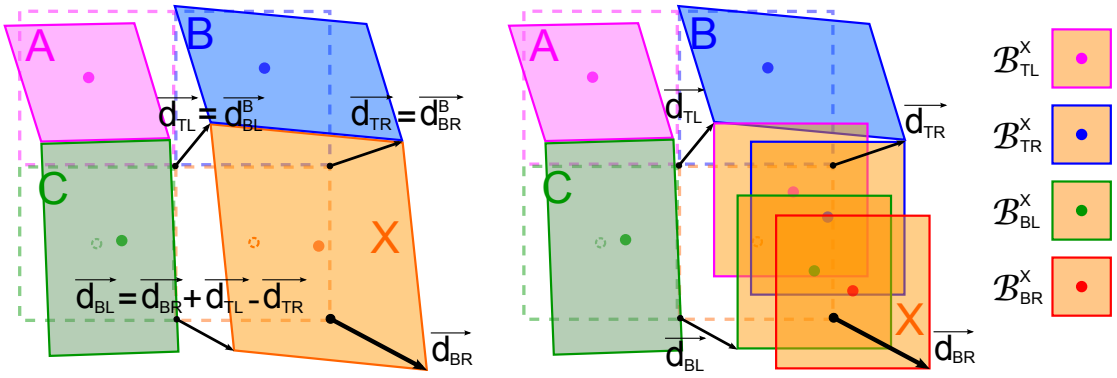
where M and N are respectively the width and height, in pixels, of the (square or rectangular - see section 5.6 -) patch of texture at the reference instant t_{ref}.

5.3.3 In between TMC and OTMC: alternative intermediate motion modes

With TMC and OTMC motion modes, neighbouring *motion tubes* can be either fully connected, or completely disconnected. At some point, however, it may be interesting to partially connect a *motion tube* to only one of its neighbours. As a solution, left OTMC and top OTMC are provided two intermediate motion modes lying in between the disconnected TMC and the connected OTMC motion modes:

- the **Top OTMC** motion mode keeps the current *motion tube* X connected to its top neighbour B;
- the **Left OTMC** motion mode keeps the current *motion tube* X connected to its left neighbour C.

5.3.3.1 Connection to the top: Top OTMC



(a) *Idealistic CGI's warping* **(b)** *Simplified block-based projection*

Figure 5.3.8: *Top OTMC mode: the current* motion tube ***X*** *is connected to its top causal neighbour* ***B***

From Top OTMC's connection pattern (see table 5.2.1), three out of the four motion vectors are sourced from the top neighbour B of the current *motion tube* X, and:

- the **top-left** motion vector $\overrightarrow{d_{TL}}(t_{cur})$ is given by B's bottom-left vector $\overrightarrow{d^{B}_{BL}}(t_{cur})$;
- the **top-right** motion vector $\overrightarrow{d_{TR}}(t_{cur})$ is given by B's bottom-right motion vector $\overrightarrow{d^{B}_{BR}}(t_{cur})$;
- the **bottom-right** motion vector $\overrightarrow{d_{BR}}(t_{cur})$ is specific to X;
- the **bottom-left** motion vector $\overrightarrow{d_{BL}}(t_{cur})$ is adjusted, in function of the other three motion vectors, to force X's section $\Omega_X(t_{cur})$ to be in the form of a parallelogram.

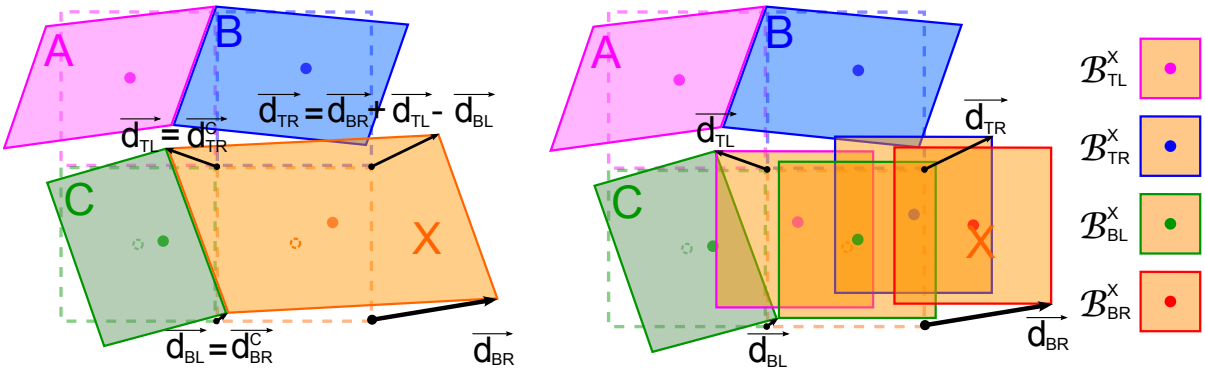
$$\begin{cases} \overrightarrow{d_{TL}}(t_{cur}) = \overrightarrow{d^{B}_{BL}}(t_{cur}) & , \quad \overrightarrow{d_{TR}}(t_{cur}) = \overrightarrow{d^{B}_{BR}}(t_{cur}) \\ \overrightarrow{d_{BL}}(t_{cur}) = \overrightarrow{d^{C}_{BR}}(t_{cur}) + \overrightarrow{d_{TL}}(t_{cur}) - \overrightarrow{d_{TR}}(t_{cur}) & , \quad \overrightarrow{d_{BR}}(t_{cur}) = \overrightarrow{d} \end{cases} \tag{5.22}$$

Both the connection pattern and the deformation abilities are illustrated in figure 5.3.8. Forward and backward motion compensations are identical to the compensation processes performed by the Full OTMC mode. Consequently, equations (5.16) and (5.20) respectively describe the Top OTMC forward and backward motion compensation processes as well.

5.3.3.2 Connection to the left: Left OTMC

Similarly, Top OTMC's connection pattern (see table 5.2.1) sources three out of the four motion vectors from the left neighbour C of the current *motion tube* X, and:

- the **top-left** motion vector $\overrightarrow{d_{TL}}(t_{cur})$ is given by C's top-right vector $\overrightarrow{d^{C}_{TR}}(t_{cur})$;
- the **bottom-left** motion vector $\overrightarrow{d_{BL}}(t_{cur})$ is given by C's bottom-right motion vector $\overrightarrow{d^{C}_{BR}}(t_{cur})$;



(a) *Idealistic CGI's warping* **(b)** *Simplified block-based projection*

Figure 5.3.9: *Left OTMC mode: the current* motion tube ***X*** *is connected to its left causal neighbour* ***C***

- the **bottom-right** motion vector $\overrightarrow{d_{BR}}(t_{cur})$ is specific to **X**;
- the **top-right** motion vector $\overrightarrow{d_{TR}}(t_{cur})$ is adjusted, in function of the other three motion vectors, to force **X**'s section $\mathbf{\Omega}_X(t_{cur})$ to be in the form of a parallelogram.

$$\begin{cases} \overrightarrow{d_{TL}}(t_{cur}) = \overrightarrow{d^{C}_{TR}}(t_{cur}) & , \quad \overrightarrow{d_{TR}}(t_{cur}) = \overrightarrow{d_{BR}}(t_{cur}) + \overrightarrow{d_{TL}}(t_{cur}) - \overrightarrow{d_{BL}}(t_{cur}) \\ \overrightarrow{d_{BL}}(t_{cur}) = \overrightarrow{d^{C}_{BR}}(t_{cur}) & , \quad \overrightarrow{d_{BR}}(t_{cur})\,\overrightarrow{d} \end{cases} \tag{5.23}$$

Both the connection pattern and the deformation abilities are illustrated in figure 5.3.9. Forward and backward motion compensations are identical to the compensation processes performed by the Full OTMC mode. Consequently, equations (5.16) and (5.20) respectively describe the Left OTMC forward and backward motion compensation processes as well.

5.3.4 Regularizing the motion discrepancies: Locally-Adaptive OTMC

While OTMC provides the ability to connect neighbouring *motion tubes*, it represents the deformations in a very crude way: any geometric shape is reduced to a set of four overlapping blocks $\mathcal{B}^{X}_{TL}$, $\mathcal{B}^{X}_{TR}$, $\mathcal{B}^{X}_{BL}$ and $\mathcal{B}^{X}_{BR}$. This crude representation may not be appropriate to any kind of deformation Two scenarios can be distinguished:

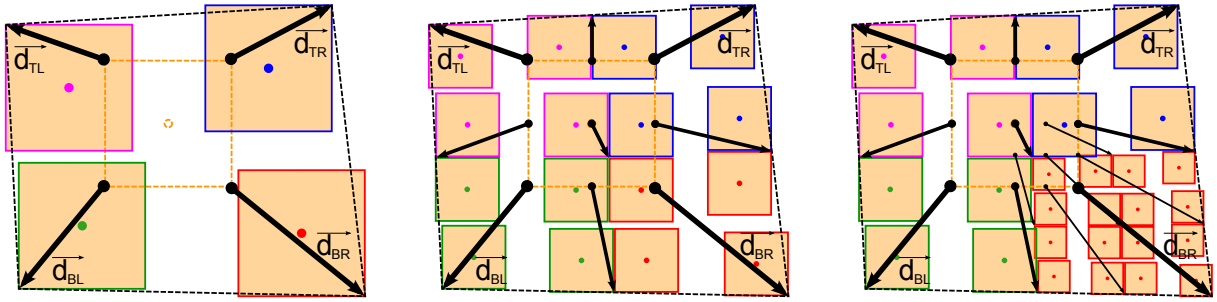
- either the **deformation** is relatively **small**, in which case its OTMC approximation is acceptable. This corresponds to scenarios where the four motion vectors $\overrightarrow{d_{TL}}$, $\overrightarrow{d_{TR}}$, $\overrightarrow{d_{BL}}$ and $\overrightarrow{d_{BR}}$ are close to each other;
- or the **deformation** is **larger**, in which case its OTMC approximation may not be acceptable. This results into motion discrepancies which could be avoided with a finer motion model.

As a solution, it is proposed to recursively split these four blocks into sub-blocks. Their displacements are interpolated from the four original motion vectors $\overrightarrow{d_{TL}}$, $\overrightarrow{d_{TR}}$, $\overrightarrow{d_{BL}}$ and $\overrightarrow{d_{BR}}$. This process is iterated until the representation of the deformation is accurate enough. Such a motion compensation process is called Locally Adaptive OTMC (LAOTMC), and is illustrated in figure 5.3.10.

1. Start with the four initial blocks $\mathcal{B}^{X}_{TL}$, $\mathcal{B}^{X}_{TR}$, $\mathcal{B}^{X}_{BL}$ and $\mathcal{B}^{X}_{BR}$.
2. Check whether the four corresponding motion vectors are close enough to each other -see equation (5.24) -.
   - If **yes**: the four (sub-)blocks are translated by their corresponding vectors.
   - If **no**: continue.
3. On each of the four (sub-)blocks:
   - partition the (sub-)block into four half-sized (sub-)blocks;
   - interpolate the motion vector of each sub-block from available motion vectors;
   - recursively iterate step 2 on the four sub-blocks.

The decision whether to split or not to split a (sub-)block relies on its four displacement difference between consecutive corners. If all the differences are under a coherence threshold δ_{Th}, the partitioning operation is not required, and the (sub-)block can be directly motion compensated:

$$\begin{cases} \left\| \overrightarrow{d_{TR}} - \overrightarrow{d_{TL}} \right\| < \delta_{Th} \quad \text{and} \quad \left\| \overrightarrow{d_{BR}} - \overrightarrow{d_{BL}} \right\| < \delta_{Th} \\ \left\| \overrightarrow{d_{BR}} - \overrightarrow{d_{TR}} \right\| < \delta_{Th} \quad \text{and} \quad \left\| \overrightarrow{d_{BL}} - \overrightarrow{d_{TL}} \right\| < \delta_{Th} \end{cases} \tag{5.24}$$



(a) *Initial OBMC blocks* **(b)** *First level partitioning* **(c)** *Second level partitioning*

Figure 5.3.10: *LAOTMC motion mode: automatic recursive partitioning of a* motion tube

As can be seen from figure 5.3.10, the accuracy of the deformation representation is locally adapted to the nature of the motion field. In particular, it can be seen from figure 5.3.10c that the shape's approximation is much more closer to the idealistic shape (represented by the dashed quadrilateral) than its initial crude approximation was. Using the LAOTMC, a larger set of deformations can be handled.

All in all, the LAOTMC provides a motion model lying in between the OTMC and the CGI, to the cost of a little extra computations: the interpolations simply consist in successive averages of two motion vectors. Provided that blocks are recursively split down to pixels, the LAOTMC can be interpreted as a CGI warping which only projects the images samples available at the reference instant. Finally, as all additional motion vectors are interpolated from original $\overrightarrow{d_{TL}}$, $\overrightarrow{d_{TR}}$, $\overrightarrow{d_{BL}}$ and $\overrightarrow{d_{BR}}$, no additional information needs to be sent.

5.3.5 TMC, OTMC and LAOTMC motion modes: compared performances

A disconnected motion mode (the TMC) and three (partially or not) connected motion modes (full, left and top OTMC) can now be applied to *motion tubes*. This section will now investigate their performances on real life sequences, and produce both objective and subjective elements to assess their abilities to track the deformation and the displacement of a set of *motion tubes*.

5.3.5.1 Hybridizing performances of connected and disconnected motion modes

a Specifications of the experiments

In order to measure and assess the performances of the different motion modes, a set of six Common Intermediate Format (CIF) sequences have been considered: *Bus*, *Football*, *Foreman*, *Mobile*, *Paris* and *Tempest*. Each of these sequences have been split into GOPs of eight frames, whom sixteen first (*i.e.* 128 frames) have been motion estimated and compensated by *motion tubes*.

In each GOP, a single family of 396 *motion tubes*, of 16×16 reference section, is used to describe its spatio-temporal content. Relatively crude parameters are used to the motion estimation, including a pixellic motion precision and a limited search range of 8×8 pixels. The LAOTMC mode has been enabled by default, such that 16×16 blocks can be split twice down to 4×4 blocks. In addition, the motion estimation is performed progressively across the temporal dimension, from one frame to the next (this can be interpreted as a progressive IPP GOP structure).

b Objective measurements

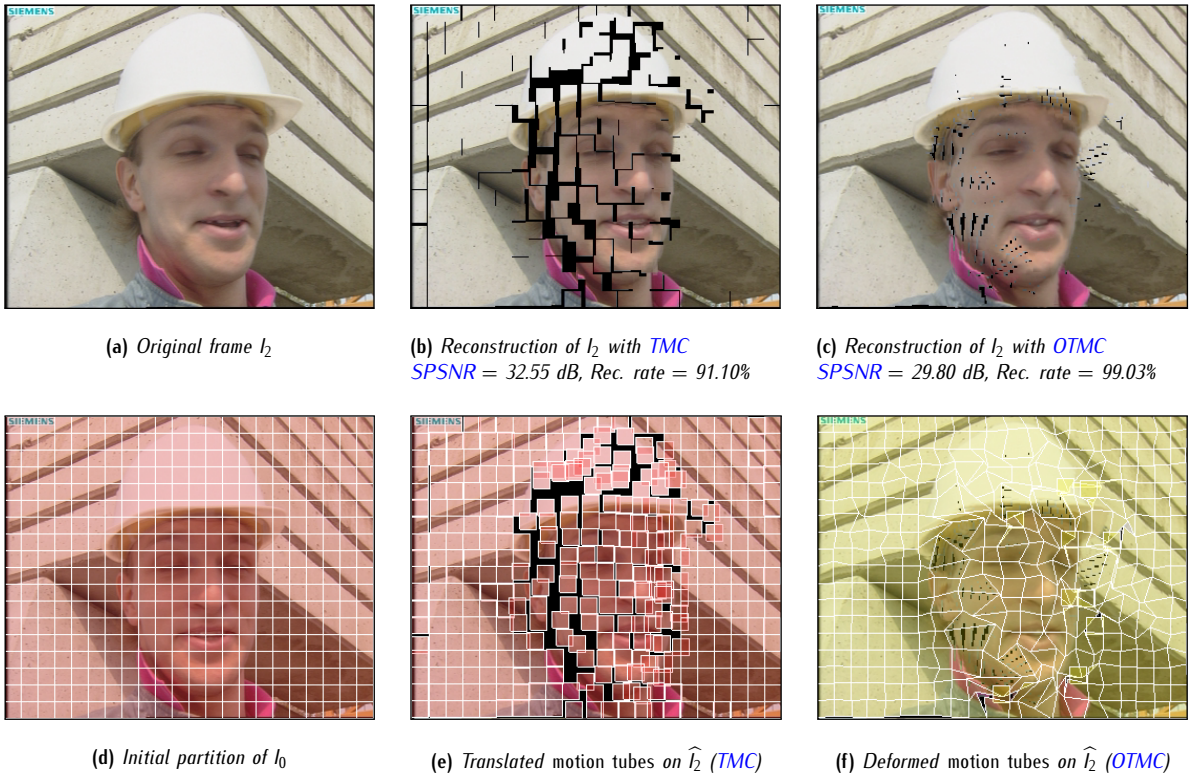
Table 5.3.1 shows the PSNRs and the reconstruction rates obtained for several motion hybridization scenarios, from the crudest (TMC mode only) to the most complex (all TMC and OTMC modes). Both PSNR and reconstruction rates are averaged over the six sequences and their 128 first frames. Since *motion tubes* do not provide, at this point, a complete rendition of the current image, its reconstruction measured from its Synthesis PSNR (SPSNR) (the PSNR of the reconstructed areas) and its reconstruction rate (the percentage of reconstructed areas). Gains in PSNR and reconstruction rates, with respect to the first scenario (TMC only), are also indicated.

Motion model				PSNR I_{ref}		SPSNR I_{cur}		Rec. rate	
TMC	OTMC	Left OTMC	Top OTMC						
✓	×	×	×	26.08 dB	- dB	26.72 dB	- dB	87.32 %	- %
×	✓	×	×	23.39 dB	-2.69 dB	22.94 dB	-2.78 dB	93.49 %	+6.17 %
×	✓	✓	✓	26.90 dB	+0.82 dB	26.88 dB	+0.16 dB	90.94 %	+3.63 %
✓	✓	×	×	27.51 dB	+1.42 dB	27.99 dB	+1.27 dB	88.85 %	+1.54 %
✓	✓	✓	✓	27.90 dB	+1.82 dB	28.32 dB	+1.60 dB	89.31 %	+2.00 %

Table 5.3.1: *Hybridization of the four motion modes: resulting PSNRs and reconstruction rates*

c Disconnected TMC versus connected OTMC

Let us focus on the first two scenarios: TMC only, and OTMC only. As can be seen from the results, the TMC motion mode provides a significantly better PSNR than the OTMC does. On the other hand, the OTMC is able to reconstruct a larger proportion of the images. This is easily explained by the fact that the OTMC motion mode is not able to represent the discontinuities of the motion field, and cannot catch up with areas corresponding to moving objects boundaries. However, forcing the *motion tubes* to be connected to each other limits the amount of unpredicted areas.



(a) *Original frame* I_2

(b) *Reconstruction of* I_2 *with TMC SPSNR = 32.55 dB, Rec. rate = 91.10%*

(c) *Reconstruction of* I_2 *with OTMC SPSNR = 29.80 dB, Rec. rate = 99.03%*

(d) *Initial partition of* I_0

(e) *Translated* motion tubes *on* $\widehat{I}_2$ *(TMC)*

(f) *Deformed* motion tubes *on* $\widehat{I}_2$ *(OTMC)*

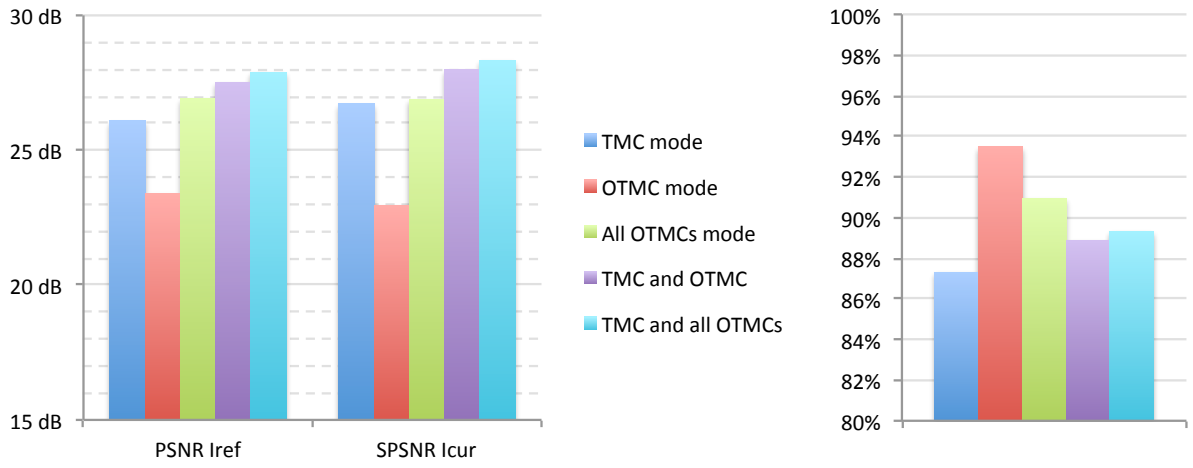
Figure 5.3.11: *TMC versus OTMC motion modes: influence on the synthesized images*

Figure 5.3.11 shows the motion compensation prediction of the third frame I_2 of sequence *Foreman* whether TMC or OTMC motion modes are used. This confirms the interpretations of the objective measurements: compared to the TMC,

the OTMC does increase the amount of reconstructed areas, while lowering the overall quality of the compensation (in particular, distorted edges can be observed–fig. 5.3.11c–). Figures 5.3.11e and 5.3.11f show the displacement and the deformation of the (idealistic) shapes of the *motion tubes*, with respect to the initial partition of the first image I_0 shown in figure 5.3.11d. Note that, in figure 5.3.11f, a few *motion tubes* are still disconnected from their neighbours (thus simply translated blocks) despite the use of the connected OTMC motion mode: this was purposely done to avoid degenerated deformations which critically impact the efficiency of the model.

d How hybridizing the different motion model does improve the motion compensation

While the efficiency of the OTMC in regards to the TMC is arguable, their hybridization undoubtedly ends up in an improved motion compensation. Figure 5.3.12 shows how progressively hybridizing the different motion modes increases both the PSNRs and the reconstruction rates. They all provide PSNRs higher than the reference score of the TMC. Hybrid scenarios, however, do not provide a reconstruction rate as high as the sole use of the OTMC does. In the end, hybridizing all four motion modes respectively increases both PSNR and reconstruction rates by 1.82 dB and 2% in average. More detailed results can be found in section C.1 of appendix C (table C.1.1 and figure C.1.1).



(a) *Averaged reconstruction PSNRs* **(b)** *Reconstruction rates*

Figure 5.3.12: *Hybridization of the four motion modes: an increase in both PSNRs and reconstruction rate*

Figure 5.3.13 shows the motion compensation prediction of the third frame I_2 of sequence *Foreman* obtained with three hybrid scenarios. In figures 5.3.13d, 5.3.13e and 5.3.13f, each motion mode is distinguished by its colour: red for TMC, yellow for full OTMC, blue for left OTMC and green for top OTMC. As can be seen, the full OTMC motion mode is advantageously used in areas which contain few edges. On the contrary, edgy areas are most often handled by partially disconnected (top and left OTMC) or disconnected (TMC) modes. If all four motion modes are enabled, the selected motion models roughly divide up into: 40% of TMC, 30% of full OTMC, 15% of left OTMC and 15% of top OTMC as well.

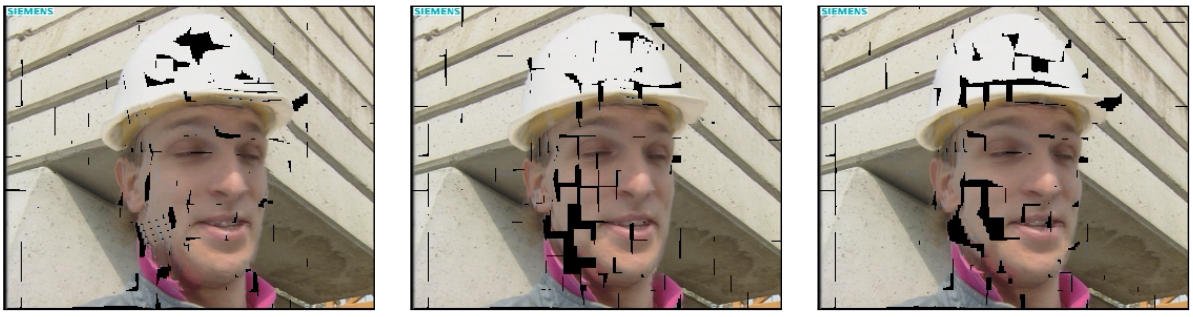
5.3.5.2 A focus on the Local Adaptive OTMC

a Setting up the experiments

Previous experiments used LAOTMC whenever a connected motion mode was available. Should the case arise, 16×16 blocks were split twice down to 4×4 blocks. It might be interesting, as well, to investigate the effects of the LAOTMC on the synthesized images. To this end, three scenarios have been evaluated, for the same six sequences as before:

1. in the first one, no LAOTMC was performed;
2. in the second one, LAOTMC was used to split original 16×16 blocks down to 4×4 blocks;
3. in the third one, finally, LAOTMC was used to split the original blocks down to 2×2 blocks.

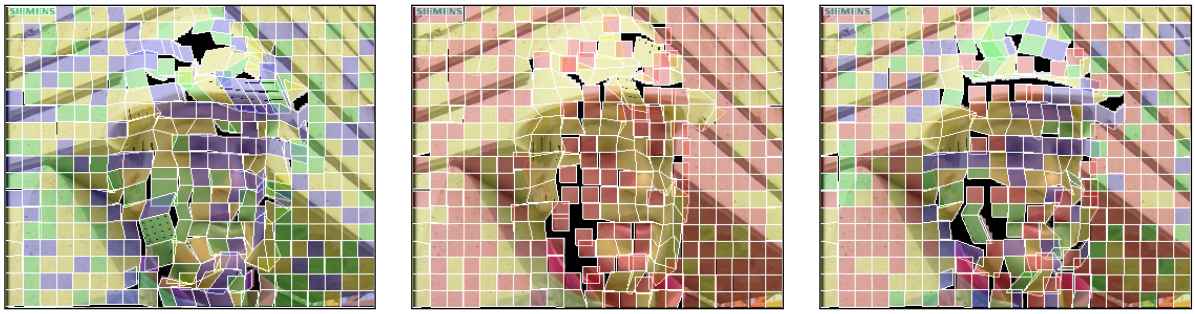
In each case, all three connected motion modes were enabled (*i.e.* full, top and left OTMC), while disconnected TMC mode was disabled, as it is not affected by the LAOTMC mechanism. Figure 5.3.14 shows the averaged PSNRs and



(a) *Reconstruction of I_2 using full OTMC, left OTMC and top OTMC modes*
SPSNR = 32.93 dB, Rec. rate = 96.91%

(b) *Reconstruction of I_2 using TMC and full OTMC modes*
SPSNR = 33.47 dB, Rec. rate = 95.78%

(c) *Reconstruction of I_2 using TMC, full OTMC, left OTMC and top OTMC modes*
SPSNR = 33.96 dB, Rec. rate = 95.44%



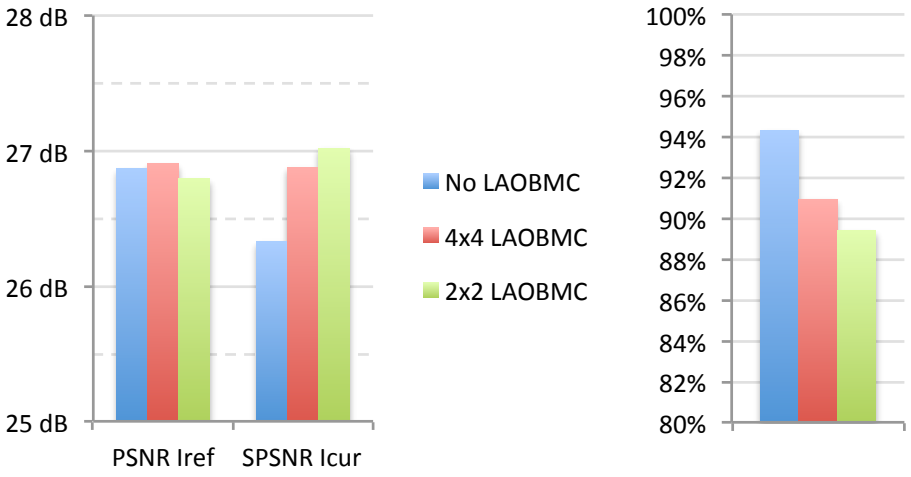
(d) *Deformed* motion tubes *on $\widehat{I}_2$ (full OTMC, left OTMC and top OTMC)*

(e) *Deformed* motion tubes *on $\widehat{I}_2$ (TMC and full OTMC)*

(f) *Deformed* motion tubes *on $\widehat{I}_2$ (TMC, full OTMC, left OTMC and top OTMC)*

Figure 5.3.13: *Hybridization of the four motion modes: influence on the synthesized images*

reconstruction rates obtained in each case. As can be seen, it has little influence on the PSNR of the reconstructed reference image I_{ref}. It does have, however, a significant influence on the PSNR of the reconstructed areas of the current frame I_{cur}. Indeed, an increase of 0.7 dB is observed in the reconstructed areas when the finest LAOTMC mechanism is activated; this, however, comes to the cost of a significant reduction of nearly 5% in reconstruction rate. In this connection, the second scenario (4×4 LAOTMC) appears as a reasonable trade-off between quality, reconstruction rate, and complexity. Detailed results can be found in section C.2 of appendix C.



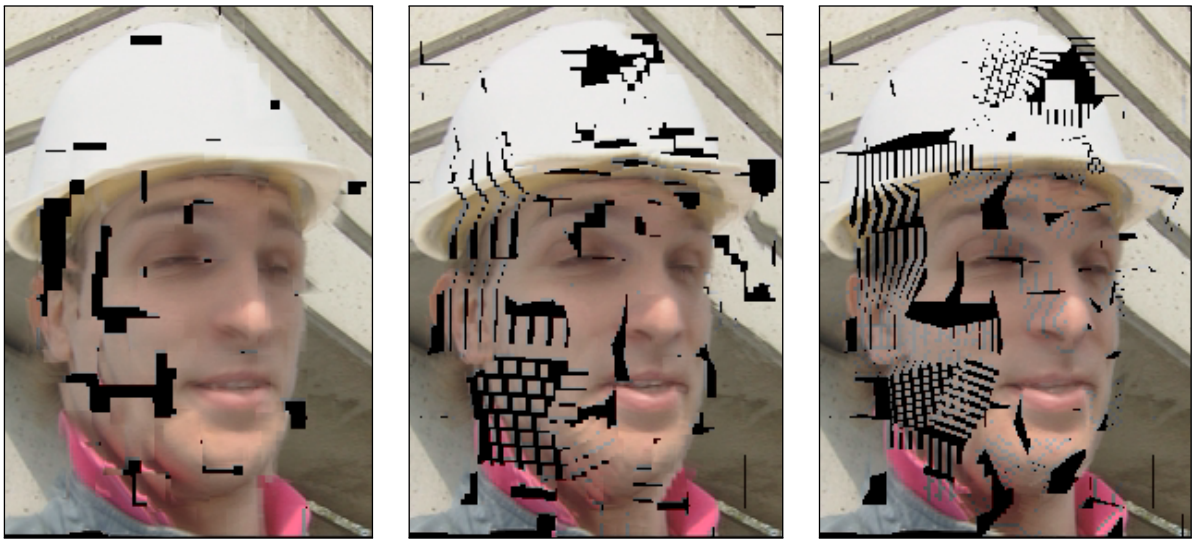
(a) *Averaged reconstruction PSNRs*

(b) *Reconstruction rates*

Figure 5.3.14: *LAOTMC: a trade-off between quality and percentage of reconstruction*

b Reducing the blurring effect due to the overlapped block motion model

Figure 5.3.15 shows the effect of LAOTMC on the fourth image I_3 of sequence *Foreman*. On the left half of Foreman's face, the large amplitude of the deformation shows how LAOTMC actually works. In figure 5.3.15a, no LAOTMC has been performed, and original 16×16 blocks are obvious, and crudely describe the deformation. In figure 5.3.15b, original blocks are split down to 4×4 sub-blocks. As can be seen, they are nicely spread over Foreman's whole cheek and jaw. In figure 5.3.15c, finally, the splitting operation spreads 2×2 sub-blocks in a similar way.



(a) *Reconstruction of I_3 without LAOTMC SPSNR = 29.55 dB, Rec. rate = 96.88%*

(b) *Reconstruction of I_3 with 4×4 LAOTMC SPSNR = 30.98 dB, Rec. rate = 93.15%*

(c) *Reconstruction of I_3 with 2×2 LAOTMC SPSNR = 31.81 dB, Rec. rate = 92.30%*

Figure 5.3.15: *LAOTMC mechanism: influence on the synthesized images*

While OBMC is well-known for reducing the blocking effects, it is to the cost of an overall blurring effect. Similarly, the OTMC compensation mechanism suffers from the same problem. However, by refining the displacement of the blocks through LAOTMC, it is expected for this blurring effect to be significantly reduced: refining the motion field prevents from undesirable motion approximations. Focusing on the nose, the mouth and the right half of Foreman's face in figure 5.3.15, one can see how LAOTMC effectively reduces the overall blurring effect, from which figure 5.3.15a suffers.

5.4 A simple and regularized motion estimation process

Whichever the motion mode, a single motion vector $\overrightarrow{d^{\mathbf{X}}_{BR}}$ needs to be estimated in order to represent the displacement and the deformation of the current *motion tube* **X**. The matching criterion, as mentioned earlier, optimizes the backward compensation of the *motion tube* onto its reference support in $I_{\mathbf{ref}}$. As can be seen from previous section, the PSNR of $I_{\mathbf{ref}}$ reconstruction and the PSNR of the synthesized areas of $I_{\mathbf{cur}}$ are very close to each other: the approach, so far, is consistent. Prior any investigation on specific features of the proposed motion estimation mechanism, section 5.4.1 will briefly review the search strategy used to localize the optimal motion vector $\overrightarrow{d^{\mathbf{X}}_{BR}}^{*}$.

5.4.1 Searching for appropriate motion vectors

5.4.1.1 An EPZS-like motion estimation search strategy

While the proposed tube-based representation was designed from coding tools whose intrinsic complexity is fairly reasonable, the actual complexity of the implementation was not a major concern. For this reason, the motion search strategy has not been thoroughly optimized and still relies on a full-search mechanism within a limited search-window. Still, it appears that EPZS [HM99, OAL01, Tou02] and its predecessors are not only fast but also tend to regularize the estimated motion field. Indeed, prior to the local Diamond Search, the search area is initialized around the best position

taken from a set of spatio-temporal predictors. Not only does this allow for the search area dimensions to be drastically reduced, but it also avoids incoherent positions from being evaluated. Consequently, the proposed motion estimation process is performed in four steps:

1. at first, evaluate the positions obtained from a set of spatio-temporal predictors (see table 5.4.1);
2. then, initialize the search area around the best predicted position;
3. evaluate all pixellic positions within the search area (this step, currently performed through an exhaustive search, could be greatly optimized from the use of a sub-optimal search strategy);
4. lastly, evalute sub-pixellic positions around the best position found at step 3; keep the best one $\overrightarrow{d^{\mathbf{X}}_{BR}}^{*}$.

In total, seven spatio-temporal predictors are considered at the first step. Let **X** be the current *motion tube* and respectively **A**, **B** and **C** its top-left, top and left neighbours. Table 5.4.1 lists the different motion predictors available at time instant t. In all our experiments on 352×288 CIF sequences, a search area of 9×9 pixels has been used.

Null predictor		Spatial predictors		Temporal predictors	
Zero:	$[0,0]^T$	Top:	$\overrightarrow{d^{\mathbf{B}}_{BR}}(t)$	Previous:	$\overrightarrow{d^{\mathbf{X}}_{BR}}(t-1)$
		Left:	$\overrightarrow{d^{\mathbf{C}}_{BR}}(t)$	Extrapolation:	$2 \cdot \overrightarrow{d^{\mathbf{X}}_{BR}}(t-1) - \overrightarrow{d^{\mathbf{X}}_{BR}}(t-2)$
		Median:	$med\left(\overrightarrow{d^{\mathbf{A}}_{BR}}(t), \overrightarrow{d^{\mathbf{B}}_{BR}}(t), \overrightarrow{d^{\mathbf{C}}_{BR}}(t)\right)$		
		Peach:	$\overrightarrow{d^{\mathbf{C}}_{BR}}(t) + \overrightarrow{d^{\mathbf{B}}_{BR}}(t) - \overrightarrow{d^{\mathbf{A}}_{BR}}(t)$		

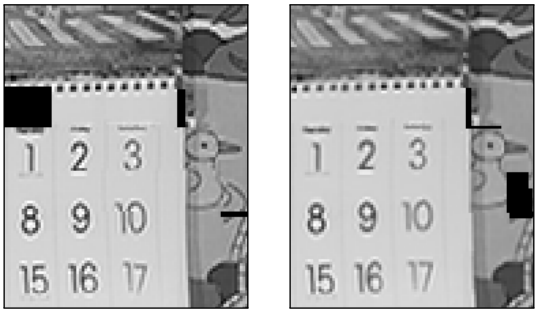
Table 5.4.1: *Set of spatio-temporal motion predictors used to initialize the search area*

5.4.1.2 Search strategy and sub-pixellic motion estimation

Sub-pixellic motion estimation usually provides a better reconstruction quality [BDH99]. Consequently, half-pixellic and quarter-pixellic positions lying around the best pixellic candidate are also evaluated. State of the art improvements are observed: the PSNR of synthesized areas, is increased by 1 dB to 2 dB. On sequences showing slow moving textures (e.g. *Mobile*), it may even be increased by 4 dB for some images. In addition, the overall sequence looks much better when playing it, as texture displacements are much more smooth (figures 5.4.1b and 5.4.1c). Table 5.4.1a gives the average PSNR of synthesized areas (namely, the SPSNR) for the first three GOPs of *Mobile* and *Foreman* CIF sequences.

Sequence	*ME accuracy*	SPSNR	Gain
Foreman	Full-pixel	29.90 dB	-
	Half-pixel	30.42 dB	+0.52 dB
	Quarter-pixel	30.59 dB	+0.70 dB
Mobile	Full-pixel	23.37 dB	-
	Half-pixel	24.98 dB	+1.62 dB
	Quarter-pixel	25.48 dB	+2.11 dB

(a) *Influence of the motion field accuracy on the SPSNR*



(b) Mobile*: full-pel motion estimation/compensation*

(c) Mobile*: quarter-pixel motion estimation/compensation*

Figure 5.4.1: *Sequences* Mobile *and* Foreman*: influence of the motion field accuracy*

5.4.2 Causal and anti-causal motion estimation steps

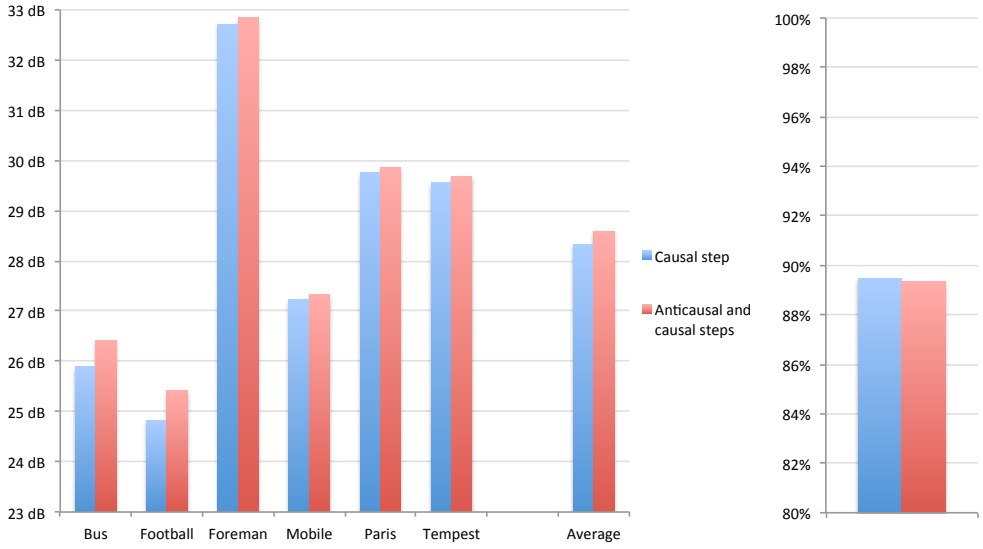
Section 5.2.5.2 introduced the concept of causal and anti-causal scans for the motion estimation of spatially-successive *motion tubes*. This section will show how considering both directions of connection does increase the quality of the reconstructed images. To this end, a series of experiments has been performed on the usual set of six sequences, with all

four motion modes activated. In the first set of experiments, the motion estimation is performed through a single causal motion estimation step. Then, the second set of experiments successively performs an anti-causal and a causal motion estimation steps.

(a) *Causal: SPSNR = 25.71 dB, Rec. rate = 84.16%* **(b)** *Anticausal & causal: SPSNR = 25.85 dB, Rec. rate = 83.60%*

Figure 5.4.2: *Reconstruction of image I_4 from sequence* Bus*: influence of the direction of the motion estimation*

From figure 5.4.2, it appears that performing a motion estimation in both directions contribute to the regularization of the motion field. Indeed, it uniformly captures the displacement of the background in sequence *Bus*, while the single-step motion estimation introduces discrepancies in the very same area of the background. This phenomenon is easily explained: due to the EPZS-like search strategy, the set of motion predictors, as any predictive filtering system, introduces a delay. Any discontinuity in the motion field is then spread over several prediction steps, thus across several *motion tubes*. By considering the correlation of the motion field into both causal and anti-causal directions, such phenomena are avoided as the deformation and the displacement of any *motion tube* is very likely to be highly correlated with either its causal and/or its anti-causal neighbours.



(a) *Reconstruction PSNRs* **(b)** *Average rec. rates*

Figure 5.4.3: *Causal and anti-causal motion estimation steps: influence on the PSNRs and the reconstruction rate*

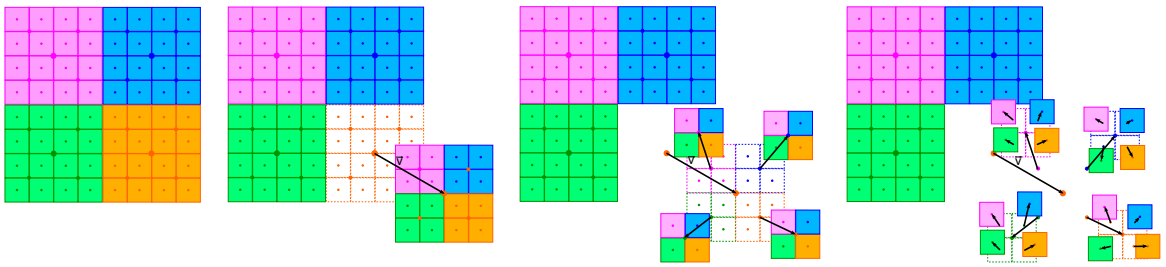
From measurements shown in figure 5.4.3, however, it can be seen that the additional motion estimation step does not greatly influence the objective quality of the reconstructed images. While the reconstruction rate is barely affected, the PSNR of the reconstructed areas is still increased, in average, by 0.26 dB. More importantly, it appears that sequences whose motion is complex, *i.e. Bus* and *Football*, are respectively improved by 0.52 dB and 0.59 dB: performing both anti-causal and causal motion estimation steps is thus advised for such sequences.

5.4.3 Spatio-temporal regularization of the *motion tubes*

While the accuracy of the displacements and the deformations of the *motion tubes* across time and space plays a key role into the proposed representation, the overall spatio-temporal regularity is even more essential to obtain a coherent and compact representation. Besides the search strategy which relies on both a predictive and a bi-directional approach, some additional regularization mechanisms were introduced as well. In this connection, section 5.4.3.1 will briefly describe a multigrid version of the motion estimation, and section 5.4.3.2 will dwell about the introduction of a rate-distortion criterion. Both these mechanisms will greatly contribute to remove the discrepancies of the motion field and ease its compaction. Their actual performances will be compared in section 5.4.3.3.

5.4.3.1 A multigrid approach to the regularization of motion tubes

In order to perform a multigrid motion estimation, *motion tubes* are organized into a hierarchical structure. Individual *motion tubes* are localized at the bottom of this structure. With each additional level, individual *motion tubes* or available groups of *motion tubes* are merged into even larger groups of *motion tubes*. A *coarse-to-fine* optimization process is then applied to the structure. Figure 5.4.4 illustrates how this mechanism is applied to an initial square set of 16×16 *motion tubes*. All of them are supposed to undergo simple translations. Eventually, additional relaxation steps can be required to locally unleash the motion field smoothing constraint.



(a) *Initial groups of* motion tubes **(b)** *First level displacement* **(c)** *Second level displacement* **(d)** *Final* motion tubes *displacement*

Figure 5.4.4: *Multigrid estimation of the translation of a group of* motion tubes

5.4.3.2 A rate-distortion approach to the regularization of motion tubes

Regularization is not only required to provide a coherent spatio-temporal motion information, but also help the estimation process to output easy-to-encode motion parameters, hence guaranteeing the representation to be compact. The multigrid approach, as it will be seen in next section, does improves the overall coherence of the representation. However, it does not guarantees that estimated motion parameters are effectively easily compacted.

An alternative approach to the regularization would then be to take into account the motion bitrate during the estimation process. In other words, this corresponds to a rate-distortion optimization of the motion parameters. A Lagrangian matching criterion -see equation (3.7)- is used to evaluate each motion candidate:

$$\xi_{\text{RDO}}\left(\mathcal{T}(t_0)\ ,\ \overline{\mathcal{T}}(t_0)\right) = \xi_{\text{Dist}}\left(\mathcal{T}(t_0)\ ,\ \overline{\mathcal{T}}(t_0)\right) + \lambda_{\text{motion}}\left[\cdot\boldsymbol{R}\left(\overrightarrow{d^{\mathbb{X}}_{BR}}^{*}\right) + \boldsymbol{R}(C)\right] \tag{5.25}$$

where $\boldsymbol{R}\left(\overrightarrow{d^{\mathbb{X}}_{BR}}^{*}\right)$ is the bitrate associated to motion vector $\overrightarrow{d^{\mathbb{X}}_{BR}}^{*}$, and $\boldsymbol{R}(C)$ is the bitrate associated to the motion mode (top and left connections). As for the Lagrange multiplier λ_{motion}, H.264/AVC's typical values are being used. In particular, they depend on the distortion criterion used to evaluate the matching candidates in the motion estimation step, and

$$\lambda_{\text{motion}} = \begin{cases} \sqrt{\lambda_{\text{mode}}} & \text{if using SAE} \\ \lambda_{\text{mode}} & \text{if using SSE} \end{cases} \tag{5.26}$$

where λ_{mode} is obtained from the input Quantization Parameter (QP):

$$\lambda_{\text{mode}} = 0.85 \times 2^{\text{QP}/3-4} \tag{5.27}$$

The regularization is achieved through the computation of $\boldsymbol{R}\left(\overrightarrow{d^{X}_{BR}}^{*}\right)$. $\overrightarrow{d^{X}_{BR}}^{*}$ is first predicted from a one of the predictors of table 5.4.1. Then, an entropy coding step is performed to estimate the bitrate $\boldsymbol{R}\left(\overrightarrow{d^{X}_{BR}}^{*}\right)$ used to signal the predictor index and transmit the motion residue $\overrightarrow{d^{X}_{BR}}^{*} = \overrightarrow{d^{X}_{BR}}^{\text{pred}} + \overrightarrow{d^{X}_{BR}}^{\text{res}}$. From the spatio-temporal nature of the predictors, the motion parameters are consequently spatio-temporally regularized.

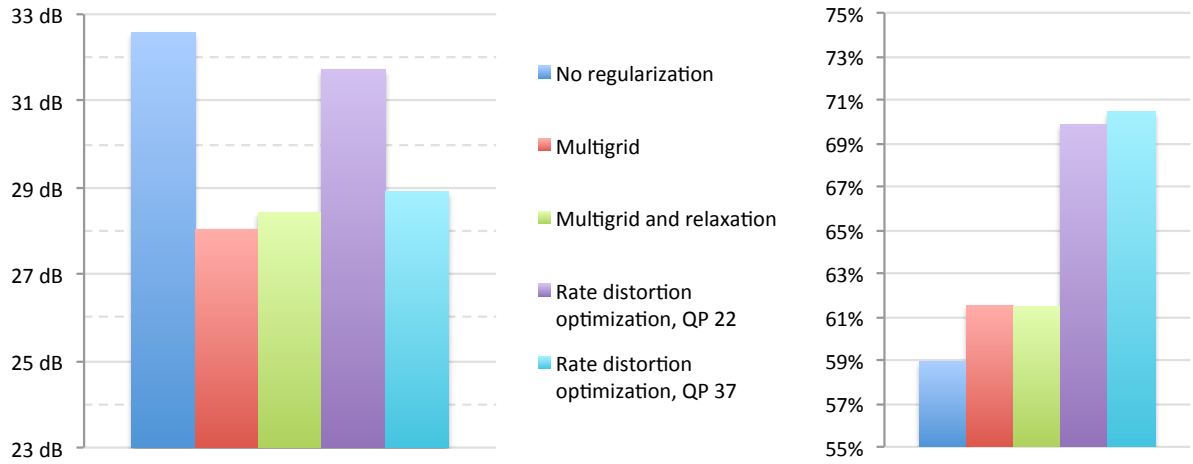
5.4.3.3 Performances of the regularization

a Setting up the experiments

In order to exhibit the regularization abilities of both multigrid and rate-distortion approaches, *motion tubes* have been parametrized in such a way that spatio-temporal discrepancies are very likely to happen. In particular, their size have been reduced from 16×16 down to 4×4 pixels. This will make the block matching motion estimator much more sensitive to noise, hence much more likely to end up in local minima. In addition, the GOP length has been increased from 8 to 32 frames; the farther from the reference instant (*i.e.* the first image of the GOP), the less likely good matches are to be found. To compensate for these settings, the search area has been increased from 9×9 to 17×17 pixels.

b Objective measurements

From figure 5.4.5, it clearly appears that both multigrid and rate-distortion regularization mechanisms both significantly increase the reconstruction percentage. Among other clues, an increase in reconstruction percentage corresponds, in most cases, to an increase in coherence and regularity. As could be expected, both multigrid and rate-distortion approaches provide an increase in reconstruction rate to the cost of a reduction in PSNR. In this direction, the rate-distortion regularization seems to perform a much better job: especially when a low QP (22) is used, the reduction in PSNR is limited to -0.85 dB, for a increase of nearly 11% in reconstruction rate. The multigrid regularization, on the other hand, decreases the PSNR by more than 4 dB (-4.16 dB at best), and only increases the reconstruction rate by 2.5%. Both mechanisms, however, are not contradictory, and may be combined to further improve the regularization; this scenario, however, has not been evaluated in practice. Further results can be found in section C.3.1 of appendix C.

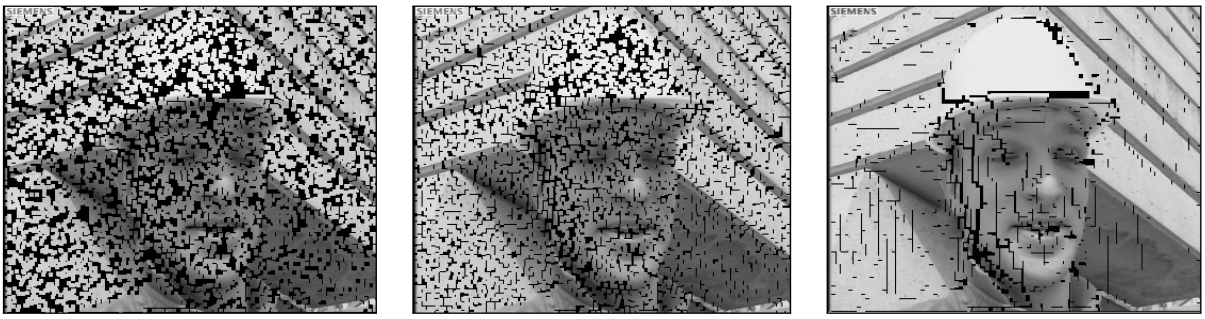


(a) *PSNRs of reconstructed areas* **(b)** *Reconstruction rates*

Figure 5.4.5: *Regularization algorithms: average influence on the PSNR and the reconstruction rate*

c Impact of the regularization on the reconstructed images: visual results

Besides its performances in terms of reconstruction rates and PSNR, figure 5.4.6 shows how much visually better images are when the rate-distortion regularization is activated: nearly all the image is reconstructed, for a fairly nice result. In particular, one can see how coherent the edges are; this is not the case for the multigrid regularization. Still, multigrid regularization significantly improves the overall coherence, and outputs a motion field which is, visually, a lot less noisy than the one produced by an non-regularized estimation.



(a) *Without regularization*
SPSNR = 41.94 dB, Rec. rate = 68.12%

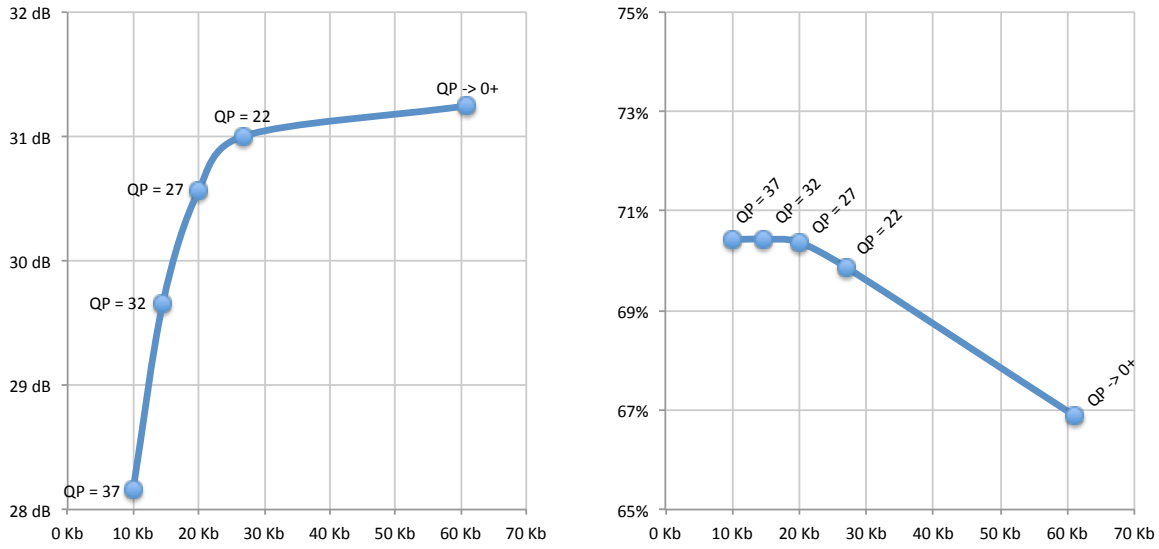
(b) *Multigrid regularization*
SPSNR = 40.63 dB, Rec. rate = 82.78%

(c) *Rate-distortion regularization*
SPSNR = 39.81 dB, Rec. rate = 93.45%

Figure 5.4.6: *Visual impact of the regularization on the second image of sequence* Foreman

d A focus on the rate-distortion mechanism and output motion bitrates

Controlling the Quantization Parameter (QP) of the rate-distortion regularization mechanism does not only influence the overall coherence of the motion information, but also controls the final bitrate which will be allocated to transmit this information. Figure 5.4.7 shows the evolution of both the PSNR of the reconstructed areas and the reconstruction rate with several values of QP (0+, 22, 27, 32 and 37). When $QP \rightarrow 0+$, no regularization is performed ($\lambda_{\text{motion}} = 0$).



(a) *SPSNR vs bitrate*

(b) *Reconstruction rate vs bitrate*

Figure 5.4.7: *Rate-distortion regularization: relationships between motion bitrate, PSNR and reconstruction rate*

Here, the motion bitrates indicate the average amount of information to transmit the motion of the *motion tubes* per frame; as could be expected, it is fairly high due to the small 4×4 dimensions of the *motion tubes*, hence to their large number. In figure 5.4.7a, a standard rate-distortion curve profile is observed: the higher the QP, the lower the motion bitrate. As for figure 5.4.7b, it shows that the reconstruction rate is not generally increasing with the QP: beyond QP 27, the reconstruction rate is stable. Further results can be found in section C.3.2 of appendix C.

For some sequences (*e.g.*: *Foreman*), the reconstruction rate is slightly lowered when increasing the QP beyond 27: this is explained by the fact that high QPs force some *motion tubes* to remain outside of the image whenever they disappear (to this point, no mechanism is in charge of detecting such events). In terms of reconstruction, this is semantically better. Above all, one should remember that, in a very similar way the QP is used to control the output bitrate in H.264/AVC, one can trust the motion QP to control the overall compacity of the tube-based representation.

5.5 Exhibiting motion tubes trajectories through appropriate motion predictors

Until now, the successive deformations of a *motion tube* were progressively estimated on a frame-by-frame basis. In terms of GOPs, this corresponds to an predictive IPP structure: the current warping $w_{k\to 0}$ of a *motion tube* $\mathbf{X}$ was initialized from its predecessor $w_{k-1\to 0}$, thus preventing $\mathbf{X}$ from behaving too much incoherently across time. Even so, such an approach does not guarantee any temporal coherence on the long run; and *motion tubes* should be even more encouraged to behave coherently across time. The motion estimation process has been subsequently improved, and:

- a **temporally-hierarchical** motion estimation process is used to further limit the temporal discrepancies of the *motion tubes* (inspired from hierarchical B-frames GOP structures);
- a motion prediction mechanism is used to clearly exhibit the **trajectory** of the *motion tubes*.

5.5.1 Hierarchical motion estimation across a GOP

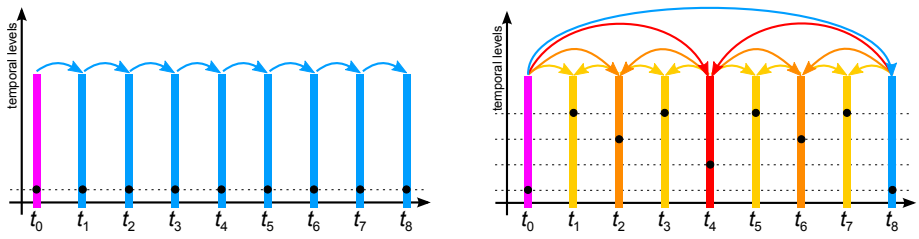
Instead of estimating and describing the successive deformations of a *motion tube* in a progressive fashion (figure 5.5.1a), *i.e.* from one instant to the next, it is proposed to regularize its deformation over a GOP by processing the temporal axis in a hierarchical fashion (figure 5.5.1b). The current displacement and deformation can now be predicted from two reference instants t_{ref0} and t_{ref1} surrounding the current instant, such that $t_{\text{ref0}} < t_{\text{ref}} < t_{\text{ref1}}$.

5.5.1.1 Hierarchical bi-prediction of the motion vectors

In order to enable the current motion vector $\overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{cur}})$ to be predicted from $\overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{ref0}})$ and $\overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{ref1}})$, the temporal motion predictors from table 5.4.1 are modified:

- predictor `previous` is generalized to a **time-backwards** prediction, and $\overrightarrow{d^{\mathbf{X}}_{BR}}^{\mathbf{pred}}(t_{\text{cur}}) = \overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{ref0}})$;
- predictor `next` is added and provides a **time-forwards** prediction, such that $\overrightarrow{d^{\mathbf{X}}_{BR}}^{\mathbf{pred}}(t_{\text{cur}}) = \overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{ref1}})$;
- predictor `extrapolation` is replaced by $\overrightarrow{d^{\mathbf{X}}_{BR}}^{\mathbf{pred}}(t_{\text{cur}}) = 1/2 \cdot \overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{ref0}}) + 1/2 \cdot \overrightarrow{d^{\mathbf{X}}_{BR}}(t_{\text{ref1}})$: predictor `interpolation`.

Figure 5.5.1b illustrates the hierarchical prediction on a GOP of nine images. At first, vector $\overrightarrow{d^{\mathbf{X}}_{BR}}^{\mathbf{pred}}(t_8)$ is predicted from `previous` predictor. In between t_0 and t_8, motion vectors are all predicted from `interpolation` predictor.



(a) *Progressive structure* **(b)** *Hierarchical structure*

Figure 5.5.1: *Progressive versus hierarchical motion prediction structures*

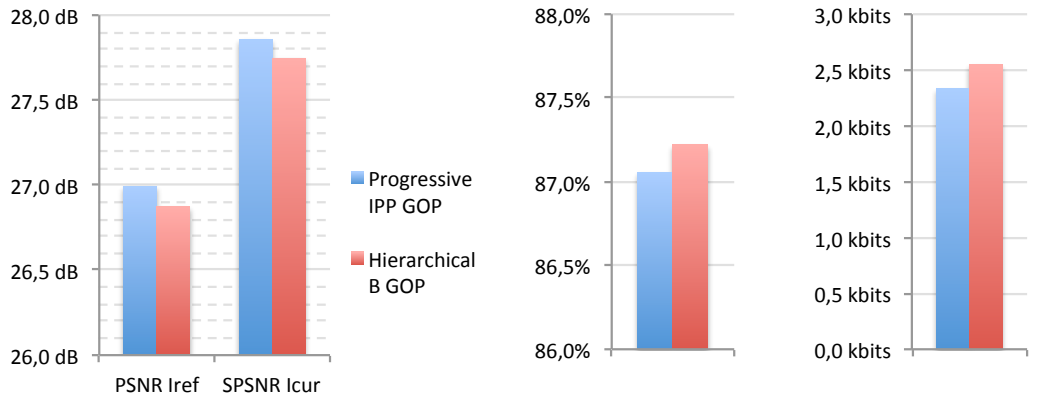
In practice, the motion $w_{0\to 8}$ between the extremities of the GOP is difficult to estimate directly from images I_0 and I_8. Both progressive and hierarchical processes can then be combined to overcome this particular issue: the motion is first crudely estimated through an IPP GOP, then refined through a hierarchical motion estimation step.

5.5.1.2 Progressive versus hierarchical motion estimation

In practice, it turns out that, even preceded by a progressive motion estimation step, the hierarchical motion estimation handles with difficulty the displacements and the deformations of the *motion tubes* between non-consecutive time instants. Compared to an estimation process void of any regularization mechanism, the hierarchical motion estimation still provides some kind of regularization, but not as efficiently as previous regularization mechanisms do.

In order to compare temporally-progressive and temporally-hierarchical approaches to motion estimation, the usual set of six sequences has been processed through GOPs of nine images. The search area has been widened to 17×17 pixels to allow for the hierarchical motion estimation to catch larger motion, as it needs to estimate the motion between temporally-distant images. In addition, both causal and anticausal motion estimation steps have been performed. The motion QP was set to 22; very similar results were obtained for higher QPs. Finally, only the translational TMC mode has been enabled for these experiments.

Figure 5.5.2 shows the influence of the GOP structure on the PSNR, the reconstruction rate and the motion cost. Eventually, the hierarchical estimation worsens the estimation process: in its attempt to regularize the motion between remote images, it settles for worse matches and slightly lowers the PSNR by 0.1 dB. On the other hand, the reconstruction rate is slightly improved by 0.15%, which is not significant. Finally, the allocated motion bitrate is even higher when compared to the progressive estimation and is increased by nearly 10%. In other words, the temporally hierarchical motion estimation should not be used under such conditions. In an attempt to improve the hierarchical motion estimation, it was proposed to increase the motion QP with increasing temporal levels, in a very similar way this is performed within H.264/AVC. In practice, neither did this improve nor did this worsen the quality of the hierarchical motion estimation. With variable-size *motion tubes*, however, hierarchical GOP structures proved to be more effective than progressive ones.



(a) *Reconstruction PSNRs* **(b)** *Reconstruction rates* **(c)** *Reconstruction PSNRs*

Figure 5.5.2: *Progressive versus hierarchical motion estimation along the temporal axis: compared performances*

Visually, the hierarchical motion estimation handles the motion discontinuities with more difficulty than the progressive estimation does. Yet, the global motion is more regularly captured. Unfortunately, both phenomena can only be perceived when actually playing the reconstructed sequences; for this reason, there will not be any visual result in this section.

5.5.2 Temporal prediction of the *motion tube* trajectories

Previous regularization mechanisms, whether they were carried out by the estimation process or intrinsically provided by the motion model, can be divided into two categories: spatial and temporal ones. So far, both directions of regularization could be combined in a very limited way. However, it is only natural to expect from spatially-neighbouring *motion tubes* to undergo similar displacements and deformations, but also for individual *motion tubes* to undergo a trajectory as consistent as possible. This section will introduce a temporal prediction mechanism which will exhort *motion tubes* to undergo trajectories as coherent as possible, while leaving room for some additional spatial regularity constraints.

5.5.2.1 Towards a motion prediction scheme exhibiting the trajectory of the *motion tubes*

It is now proposed an alternative prediction structure which takes into account the overall trajectory of the *motion tubes*. The motion prediction is now built in two steps:

1. according to the GOP structure, (resp.) progressive or hierarchical, the trajectory of the current *motion tube* X first temporally predicted from (resp.) `extrapolation` or `interpolation` predictors;
2. the temporal prediction is then refined using any of the available predictors from table 5.4.1.

Figure 5.5.3 shows this prediction scheme on a hierarchical GOP structure. The overall motion vector is now given by

$$\overrightarrow{d^{\mathrm{X}}_{BR}}(t) = \underbrace{\widetilde{T_{\mathrm{X}}}(t)}_{\text{predicted trajectory}} + \underbrace{\overrightarrow{\Delta d^{\mathrm{X}}_{BR}}^{\mathrm{pred}}(t)}_{\text{spatial prediction}} + \overrightarrow{d^{\mathrm{X}}_{BR}}^{\mathrm{res}}(t) \tag{5.28}$$

where $\widetilde{T_{\mathrm{X}}}(t)$ is the trajectory prediction, $\overrightarrow{\Delta d^{\mathrm{X}}_{BR}}^{\mathrm{pred}}(t)$ is the prediction refinement, and $\overrightarrow{d^{\mathrm{X}}_{BR}}^{\mathrm{res}}(t)$ is the motion residue.

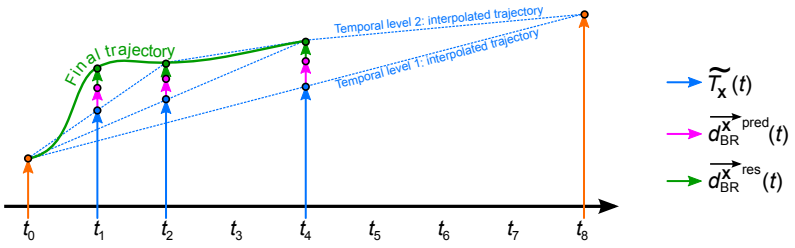
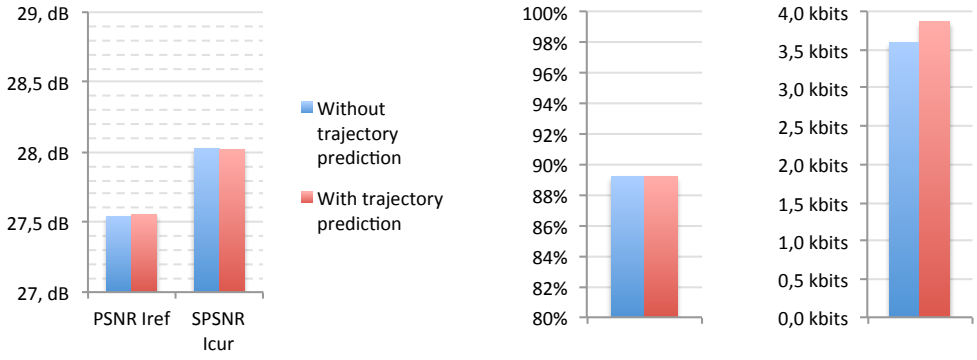


Figure 5.5.3: *A motion vector prediction mechanism exhibiting the trajectories of the* motion tubes

5.5.2.2 Temporal rate-distortion regularization of the trajectory

In practice, it turns out that exhibiting the trajectory within the prediction process does not improve the representation. Yet, it does not worsen the estimation quality: in average, neither the PSNRs nor the reconstruction rate are affected. The motion cost, however, is increased by nearly 8%. The GOP structure does not affect these results. For the record, the search area was set to 9×9 pixels progressive GOPs, and 17×17 pixels for hierarchical GOPs, and all motion modes were enabled. Finally, the motion QP was set to 22; though very similar results were obtained for higher QPs. Figure 5.5.4 shows how the trajectory prediction impacts the PSNR, the reconstruction rate and the motion cost.



(a) *Reconstruction PSNRs* **(b)** *Reconstruction rates* **(c)** *Motion costs*

Figure 5.5.4: *Progressive versus hierarchical motion estimation along the temporal axis: compared performances*

Visual results, similarly, do not show any significant differences, and are not subject to any particular interpretation. In conclusion, the trajectory prediction, in its current state, is not especially advised. However, applied on variable size *motion tubes* (they are introduced in next section), the temporal prediction proved to be quite efficient when it comes to predict the hierarchical structure, provided that a hierarchical GOP structure is used.

5.6 Content-adaptive variable size *motion tubes*

While block-based representations proved to be a simple and efficient way to partition image sequences from a compression perspective, they often suffer from their inability to adapt to the geometry of the spatial contents. Consequently, several still image and video compression schemes using variable block size have also been proposed, including the LAR still image coder [DBBR07] and the ITU-T H.264/AVC video compression standard.

Until now, *motion tubes* were initialized from a regular partition of the reference image into fixed-size blocks. However, areas where the motion field is uniform may be favourably represented by *motion tubes* of large dimensions. On the other hand, areas where the motion field is rapidly varying may be much better handle through numerous *motion tubes* of small spatial dimensions. As a consequence, it is important to adapt their size and shape (square, rectangular) to the images contents, in order to reach an optimal tradeoff between partitioning accuracy and amount of motion information.

To this end, it is now proposed to allow the *motion tubes* to be split into sub-tubes. Each of these will then be able to undergoing individual displacements and deformations. A hierarchical structure, optimized though a rate-distortion mechanism, will then be provided; thus ensuring that the size of the *motion tubes* are fit to the images contents, while limiting the increase in motion information bitrate.

5.6.1 A hierarchical structure for the *motion tubes*

5.6.1.1 A set of partitioning operators

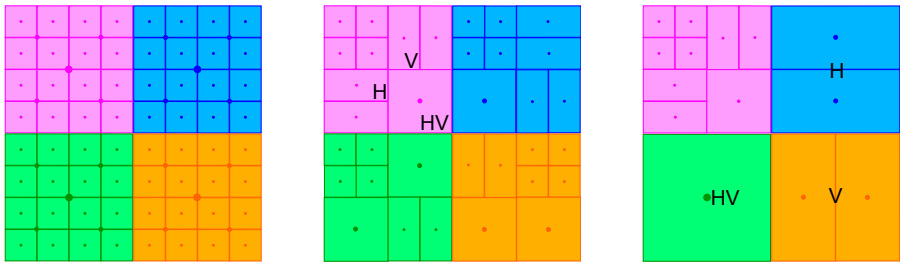
In a very similar way *motion tubes* were processed by the multigrid regularization, a pyramid of *motion tubes* is built. Let $\mathbf{X}^0$ be a top-level *motion tube* of dimensions $32 \times 32 pixels$. May this be favourable, $\mathbf{X}^0$ can be split into several sub-tubes, according to three partitioning patterns:

- with `N` **(no) split:** $\mathbf{X}^0$ is kept still;
- with `H` **(horizontal) split:** $\mathbf{X}^0$ is split into two 32×16 rectangular sub-tubes $\mathbf{X}^{*,0}$ (top) and $\mathbf{X}^{*,1}$ (bottom);
- with `V` **(vertical) split:** $\mathbf{X}^0$ is split into two 16×32 rectangular sub-tubes $\mathbf{X}^{0,*}$ (left) and $\mathbf{X}^{1,*}$ (right);
- with `HV` **(horizontal & vertical) split:** $\mathbf{X}^0$ is split into four 16×16 square sub-tubes $\mathbf{X}^{0,0}$ (top-left), $\mathbf{X}^{0,1}$ (top-right), $\mathbf{X}^{1,0}$ (bottom-left) and $\mathbf{X}^{1,1}$ (bottom-right).

The process can then be recursively iterated for `HV` partitions. For instance, $\mathbf{X}^{0,0}$, $\mathbf{X}^{0,1}$, $\mathbf{X}^{1,0}$ and $\mathbf{X}^{1,1}$ can be further split into 16×8, 8×16 and 8×8 sub-sub-tubes, and so on... The partition is then simply described by the corresponding set of successive splitting operations.

5.6.1.2 A bottom-up hierarchical motion estimation

Numerous approaches can be used to optimize a hierarchical structure. Here, the bottom-up approach is a straightforward candidate, as it allows for the structure to be parsed in a simple way, with a limited use of dynamic programming techniques. Instead of splitting the *motion tubes* into sub-tubes, it starts by evaluating minimum-sized sub-tubes (*e.g.* 4×4 sub-tubes), and then recursively evaluates higher levels by merging current sub-tubes into larger sub-tubes, following the available partitioning patterns. The partitioning patterns, and the estimation process are illustrated in figure 5.6.1.



(a) *Bottom level sub-tubes* **(b)** *Second level partition* **(c)** *Third level partition*

Figure 5.6.1: *Successive steps of the bottom-up hierarchical motion estimation*

5.6.2 Rate-distortion optimization of the hierarchical structure of *motion tubes*

While increasing the number of *motion tubes* may significantly improve the reconstruction, it also increases the amount of motion information to be transmitted, as there are as many motion vectors to transmit as there are *motion tubes* or sub-tubes. The partitioning information also needs to be transmitted, hence to be taken into account. In an eye to

compression, it is critical for the hierarchical structure to provide an optimal tradeoff between representation cost and reconstruction quality. Once again, it is proposed to optimize the structure through a rate-distortion approach.

To this end, the Lagrangian is modified and now incorporates the cost of the partitioning information. Let us now recall the *motion tube* $\mathbf{X}^0$ and its possible children $\mathbf{X}^{*,0}$ (top), $\mathbf{X}^{*,1}$ (bottom), $\mathbf{X}^{0,*}$ (left), $\mathbf{X}^{1,*}$ (right), $\mathbf{X}^{0,0}$ (top-left), $\mathbf{X}^{0,1}$ (top-right), $\mathbf{X}^{1,0}$ (bottom-left) and $\mathbf{X}^{1,1}$ (bottom-right). The optimal partitioning pattern $\mathcal{P}^*$ for $\mathbf{X}^0$, from a rate-distortion perspective, is given by

$$\mathcal{P}_0^* = \arg\min_{\mathcal{P}_0} J(\mathcal{P}_0, \mathbf{X}^0) \tag{5.29}$$

where $J\left(\mathcal{P}_0, \mathbf{X}^0\right)$ is recursively defined as:

$$J\left(\mathcal{P}, \mathbf{X}^0\right) = \begin{cases} \xi_{RDO}\left(\mathbf{X}^0\right) + \lambda_{\text{motion}} \cdot \boldsymbol{R}\left(\mathtt{N}\right) & \text{if } \mathcal{P} = \mathtt{N} \\ \sum_{i=0}^{1} J\left(\mathtt{N}, \mathbf{X}^{i,*}\right) + \lambda_{\text{motion}} \cdot \boldsymbol{R}\left(\mathtt{H}\right) & \text{if } \mathcal{P} = \mathtt{H} \\ \sum_{i=0}^{1} J\left(\mathtt{N}, \mathbf{X}^{*,i}\right) + \lambda_{\text{motion}} \cdot \boldsymbol{R}\left(\mathtt{V}\right) & \text{if } \mathcal{P} = \mathtt{V} \\ \sum_{i=0}^{1}\sum_{j=0}^{1} J\left(\arg\min_{\mathcal{P}_{i,j}} J(\mathcal{P}_{i,j}, \mathbf{X}^{i,j}), \mathbf{X}^{i,j}\right) + \lambda_{\text{motion}} \cdot \boldsymbol{R}\left(\mathtt{HV}\right) & \text{if } \mathcal{P} = \mathtt{HV} \end{cases} \tag{5.30}$$

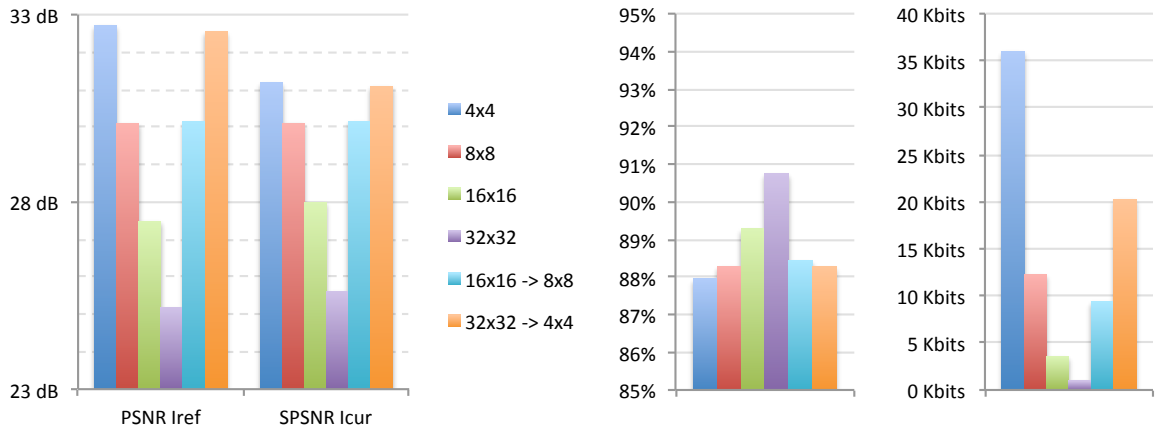
where $\xi_{RDO}\left(\mathbf{X}\right)$ is short notation for the classic Lagrangian defined in equation (5.25).

5.6.3 Performances of the variable size motion tubes

5.6.3.1 Setting up the experiments

Once again, the usual set of six sequences have been processed through *motion tubes* of various dimensions. A first series of four experiments uses fixed-sized *motion tubes*; successively 4×4, 8×8, 16×16 and 32×32. Then, a second series of experiments employed a hierarchical structure of *motion tubes* to provide variable-size *motion tubes*. In each case, all motion modes were enabled, along with the LAOTMC. The search range was set to 9 by 9 pixels; a progressive GOP structure was used, and the temporal prediction was disabled.

5.6.3.2 Influence of the size of the motion tubes



(a) *Reconstruction PSNRs* **(b)** *Reconstruction rates* **(c)** *Reconstruction PSNRs*

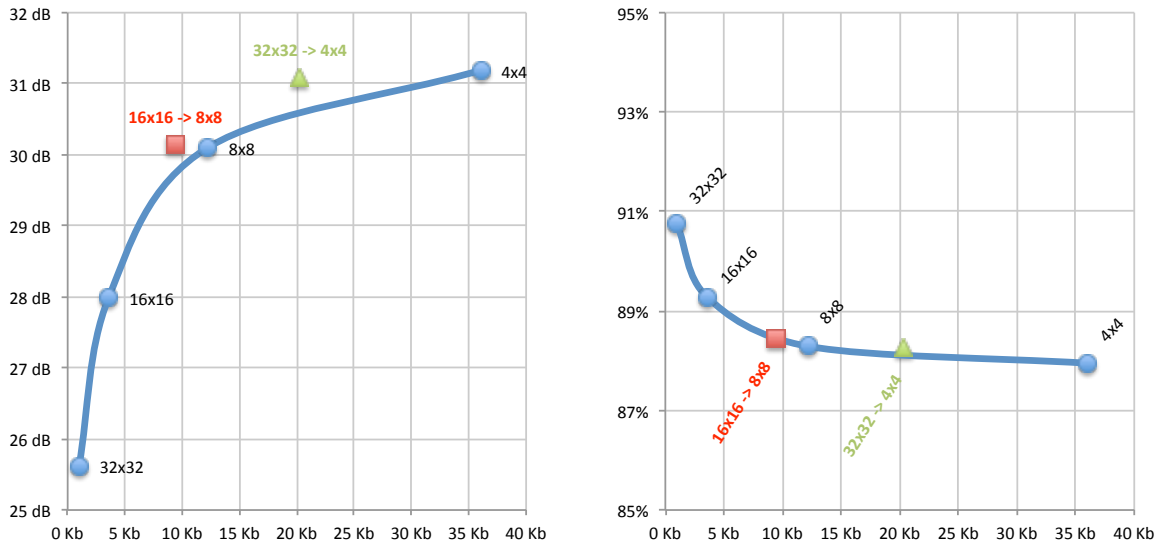
Figure 5.6.2: *Influence of the spatial dimensions of the* motion tubes*: a tradeoff between size and bitrate*

How much does the size effectively matters? Preliminary evaluations showed that the spatial dimensions of the *motion tubes* has a large impact on the provided quality and reconstruction rate, and especially on the motion bitrate. Indeed,

the spatial resolution of the *motion tubes* is highly correlated to the resolution of the motion field. The higher the resolution, the more accurate the motion information. As can be seen from figure 5.6.2, increasing the *motion tubes* size from 4×4 dramatically decreases the PSNR by 7.5 dB. In the meantime, however, the motion bitrate drops by more than 97%. Obviously, both configurations are a quite extreme and should not be considered as practical settings. Still, they illustrate how much important it is to appropriately select the spatial dimensions of the *motion tubes*. Note that the reconstruction rate is far less affected by the dimensions of the *motion tubes*.

5.6.3.3 How the hierarchical partition provides a tradeoff between spatial resolution and bitrate

Figure 5.6.3 plots both the PSNR of the reconstructed areas and the reconstruction rate against the motion bitrate. The curve represents the best performances brought by fixed size *motion tubes*. The square and the triangle marks, on the other hand, correspond to the performances brought by the hierarchical structure. The square corresponds to a hierarchical structure which allows for *motion tubes* whose dimensions range from 8×8 to 16×16 pixels. As for the triangle, it corresponds to a hierarchical structure which allows for *motion tubes* range from 4×4 to 32×32 pixels.



(a) *SPSNR vs bitrate*

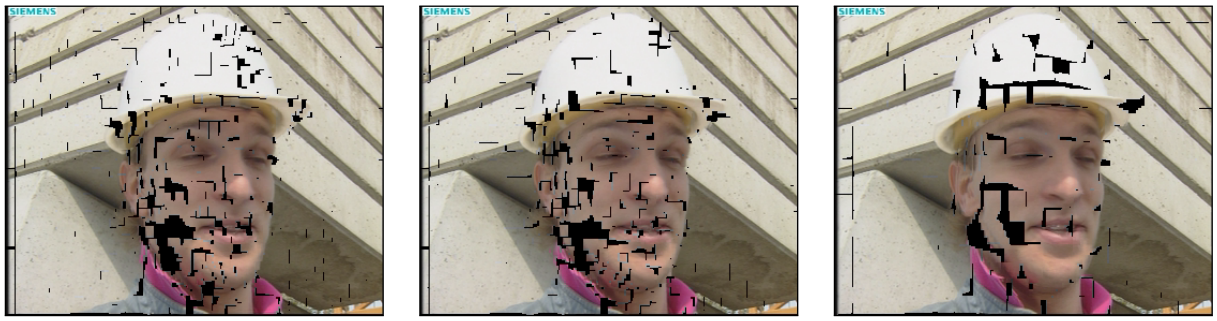
(b) *Reconstruction rate vs bitrate*

Figure 5.6.3: *Variable-size* motion tubes*: relationships between motion bitrate, PSNR and reconstruction rate*

As can be seen, they both provide an increase in terms of PSNR, for a fixed bitrate. Alternatively, they provide a much more compact motion information for a fixed PSNR. In terms of reconstruction rate, both structures fall onto the curve of the fixed-sized *motion tubes*: the hierarchical structure does not improve nor does worsen the reconstruction rate. Corresponding PSNR, reconstruction rate and motion bitrate values can be found in figure 5.6.2. Further results can be found in section C.4 of appendix C.

5.6.3.4 Visual impact on the reconstructed images

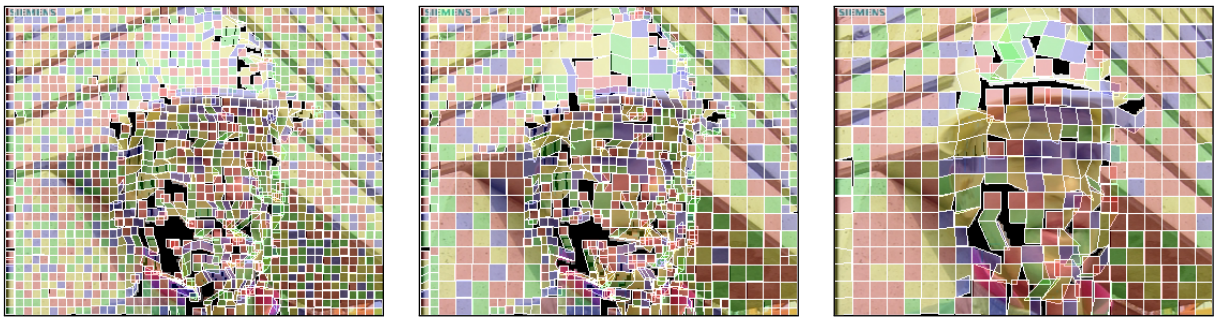
As can be seen in figure 5.6.4, the hierarchical structure is nicely optimized such that large *motion tubes* are used to represent uniformly moving areas. In sequence *Foreman*, this mainly corresponds to the background. By doing this, it saves a large part of the motion bitrate to represent more complex motions in a more accurate way. Areas where the motion is either complex or discontinuous are split into smaller *motion tubes*; here, these are mainly found on the face of Foreman, but also on the boundaries of the images wherein the motion field is also discontinuous as it ends. Also, the ability to identify uniformly moving areas helps with the regularization of their motion.



(a) *Reconstruction of* I_2 *using* 8×8 motion tubes *SPSNR = 36.55 dB, Rec. rate = 94.31%*

(b) *Reconstruction of* I_2 *using a hierarchial structure of* motion tubes *(*8×8 *to* 16×16*) SPSNR = 36.24 dB, Rec. rate = 94.81%*

(c) *Reconstruction of* I_2 *using* 16×16 motion tubes *SPSNR = 33.96 dB, Rec. rate = 95.44%*



(d) *Deformed* 8×8 motion tubes *on* $\widehat{I_2}$

(e) *Deformed variable-size* motion tubes *on* $\widehat{I_2}$

(f) *Deformed* 16×16 motion tubes *on* $\widehat{I_2}$

Figure 5.6.4: *Variable size motion tubes: influence on the synthesized images for sequence* Foreman

5.7 Transmitting the motion parameters: a focus on coding mechanisms

The interest of a rate-distortion approach to the motion estimation of the *motion tubes* is twofold: it is a simple and very effective way to regularize the motion field (both in time and space); in addition, the provided QP allows the user to control the bitrate of the output motion information. Along with an efficient motion coding scheme, it provides a low-cost representation of the deformations of the *motion tubes*. Most of the results showed in previous sections included the bitrate of the motion information; yet it was not detailed how exactly was the motion information entropy coded.

This section will detail the different natures of the motion information, along with corresponding coding schemes. As the deformations are described by several types of information, the coding process needs to be adapted to each of its elements. The whole coding process is strongly inspired from H.264/AVC's entropy coding scheme which was reviewed in section 3.3.2: it consists of a CABAC entropy coder (see section 3.3.2.2) several coding contexts, each of which being associated to a specific motion information.

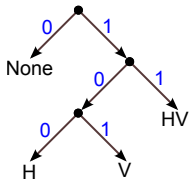
5.7.1 Overall structure of the motion information and corresponding bitrates

Whichever the motion mode used to describe the deformation of a *motion tube*, a single motion vector needs to be transmitted for each time instant. As seen in section 5.5.2.1, this motion vector is signalled through several piece of information. In the end, the nature of the motion information is fourfold:

1. the **partitioning** information describing, for each time instant, how *motion tubes* may be split into sub-tubes;
2. the **connection** flags indicating which *motion tubes* are connected to each other; in other words, the motion modes;
3. the **prediction** mode used to generate the predicted motion vector;
4. and finally, the motion **residual** information, *i.e.* the difference between the predicted and the actual motion vectors.

5.7.1.1 The partitioning information

To each partitioning mode (None, H, V and HV) is associated a binary symbol, following a Variable Length Code (VLC). Corresponding symbols are expressed by one, two or three bins. The latter are then further entropy coded using CABAC: to each bin is associated a particular CABAC context (A, B or C), as each of the three bins may follow a specific distribution law. Figure 5.7.1a shows the binary tree used to binarize the symbols, while table 5.7.1b shows corresponding symbols. Note that the partition is also temporally predicted: from the partition of the previously processed time instant, it is only transmitted additional partitioning flags which indicate whether or not to further partition the *motion tube*.



(a) *VLC tree*

Partitioning mode $\mathcal{P}$	None	H	V	HV	CABAC context
1^{st} bin	0	1	1	1	A
2^{nd} bin	-	0	0	1	B
3^{rd} bin	-	0	1	-	C
Symbol	0	100	101	11	-

(b) *Partitioning symbols input to the CABAC encoder*

Figure 5.7.1: *Partitioning information: VLC binarization and associated CABAC contexts*

5.7.1.2 Motion modes and connection flags

Motion modes are signalled through *top* and *left* connection flags (C_T and C_L). As connection flags already consist in a series of binary symbols, they are directly entropy coded using appropriate context models. The latter depends on the value of the connection flags of the *motion tube* at previous processing instant.

5.7.1.3 Spatial and temporal motion predictors

In the end, only two different motion predictors $\widetilde{S}_0$ and $\widetilde{S}_1$ are used to perform the entropy coding operation. For the TMC motion modes, both `top` and `left` predictors are advised. For OTMC motion modes, the temporal predictor is advised, the other one being respectively the `median`, the `top` and the `left` predictors for full, top and left OTMC modes. A binary flag is used to signal which of the two predictors needs to be used. Whenever both predictors are equal, however, no flag needs to be transmitted. In addition, the *temporal* prediction mode may also weight both contributions, which, again, avoids the additional prediction binary flag.

5.7.1.4 Residual motion information: an H.264/AVC-like entropy coder

Previous information allowed the motion model to retrieve the motion field from various information, including partitioning symbols, connection flags and prediction modes. Now remain the motion residues $\overrightarrow{d_{\text{res}}}$, which also need to be sent so that the exact motion vectors can be retrieved at the decoder. It is proposed to encode the residual motion information exactly as it is done by H.264/AVC.

At first, a single binary flag is sent to signal whether the residual motion vector is or is not null. The flag is entropy coded with CABAC; corresponding coding context depends on the sum of the bottom-right motion vectors of top and left neighbouring *motion tubes* **B** and **C** (respectively $\overrightarrow{d^{\mathbf{B}}_{\text{BR}}}$ and $\overrightarrow{d^{\mathbf{C}}_{\text{BR}}}$ If null, no more information is transmitted; else, residual amplitudes along directions x and y are transmitted: the sign of $\overrightarrow{d_{\text{res}}}$ is transmitted as a single bit. Then, its quarter-pixellic amplitude $\overrightarrow{d_{\text{res}}}$ is first binarized using exponential Golomb-Rice VLCs; output bits are then further entropy coded using CABAC.

5.7.2 Building the motion binary bitstream

In the end, $\boldsymbol{R}(\mathcal{C}_{\mathcal{W}}(\mathcal{W}))$, the overall bitrate used by $\mathcal{M}_T$'s motion information, is then obtained by summing individual bitrates corresponding to each type of information: partitioning, connection, prediction, and residual bitrates. Over a

whole GOP of G_S images, $\mathcal{M}_\mathcal{T}$'s overall motion coding cost is then given by

$$\begin{aligned} \boldsymbol{R}\left(\mathcal{C}_\mathcal{W}\left(\mathcal{W}\right)\right) &= \sum_{i=0}^{G_S-1} \boldsymbol{R}\left(\mathcal{C}_\mathcal{W}\left(w_{i\rightarrow i+1}\right)\right) \\ &= \sum_{i=0}^{G_S-1} \left[\boldsymbol{R}\left(\mathcal{P}\right) + \boldsymbol{R}\left(C_{\mathrm{T},i}\right) + \boldsymbol{R}\left(C_{\mathrm{L},i}\right) + \left(1-\delta_{\widetilde{S}_0,\widetilde{S}_1}\right)\cdot \boldsymbol{R}\left(\widetilde{S}_0,\widetilde{S}_1\right) + \boldsymbol{R}\left(\overrightarrow{d_{\mathrm{res},i}}\right)\right] \end{aligned} \tag{5.31}$$

where $\delta_{a,b}$ is the Kronecker delta ($\delta_{a,b}=1$ if $a=b$; otherwise, $\delta_{a,b}=0$). The motion binary bitstream is simply obtained by concatenating all these information. Figure 5.7.2 illustrates the repartition of the motion bitrate between all these information for the first three GOPs of sequence *Foreman* when QPs 22 and 37 are used. All motion modes have been enabled, and a hierarchical structure is used to handle *motion tubes* whose size range from 8×8 to 16×16 pixels. As it may be expected, the motion residues represent the largest part of the information. As for the partitioning information, it does not seem to be much affected by the motion QP; at low bitrates, it is more efficient to preserve the partition and reduce the amount of information used by the residues, instead of the other way round.

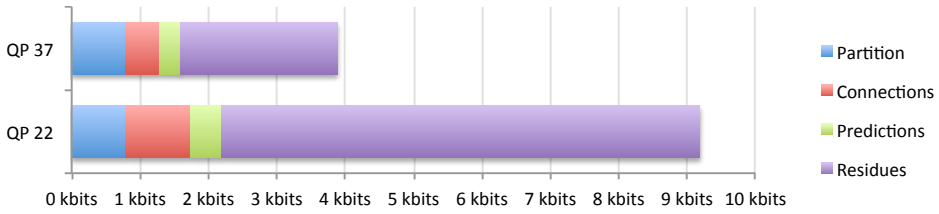


Figure 5.7.2: *Repartition of the motion bitrate into the different types of information*

As the motion information of neighbouring *motion tubes* are interdependent, all the motion parameters needs to be sent to the decoder, even if a single *motion tube* needs to be decoded. This may prove to be a problem when, as will be seen in chapter 7, some of the *motion tubes* will need to be removed as they do not correspond to a spatio-temporal area of the sequence which cannot be handled through the use of *motion tubes*. In such cases, the prediction mechanism will be altered, such that it does not involve the disabled *motion tubes* anymore.

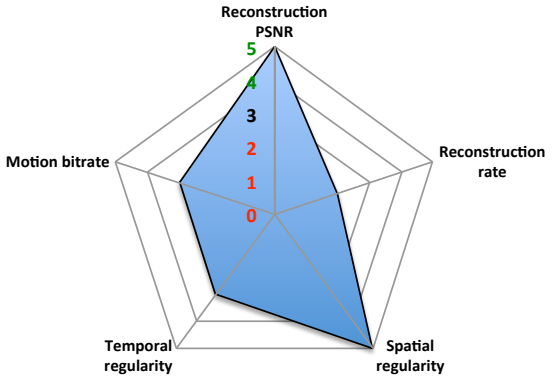
5.8 Conclusion

This chapter provided an hybrid motion model to the *motion tubes*, in order to describe their displacements and deformation across time. From chapters 2 and 3, geometrical models were identified as a simple and effective approach to the representation of the motion field. From a compression perspective, it was also required for the motion model to be relatively low-computational, and to involve as few parameters as possible. To match all these requirements, a modified version of the SOBMC was introduced, and describes the deformation of the *motion tubes* through following modes:

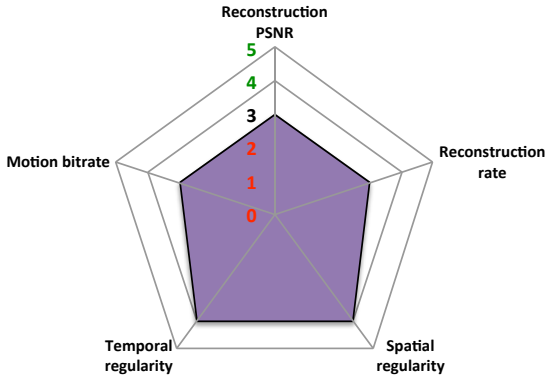
- a **disconnected** model (Tube Motion Compensation), describing **translational** motions;
- a **connected** model (Overlapped Tube Motion Compensation), whose abilities lie in between OBMC and CGI;
- intermediate left and right OTMC modes, **connecting** the *motion tubes* into a single direction.

Hybridizing these four models proved to be quite effective and provided great improvements in comparison to a simple translational motion model. But, most importantly, a set of regularization mechanisms were provided to guide the estimation so that it outputs a coherent spatio-temporal motion information and prevents from motion discrepancies to appear. Most often, though, these mechanisms would improve a given feature, but eventually decrease some other feature. Until now, five main criteria have been used to assess the objective and subjective quality of the reconstructed images:

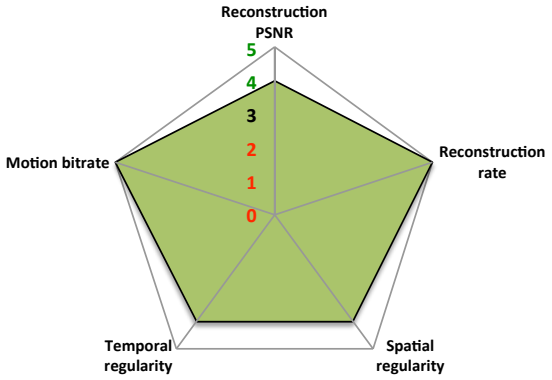
1. the PSNR of the reconstructed areas;
2. the reconstruction rate: percentage of the images which have been synthesized;
3. the spatial regularity of the motion information;
4. the temporal regularity of the motion information;
5. and finally, the compactness of the representation (the motion bitrate).



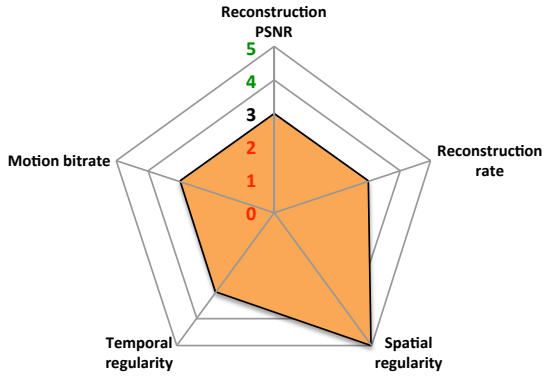
(a) *LAOTMC mechanism*



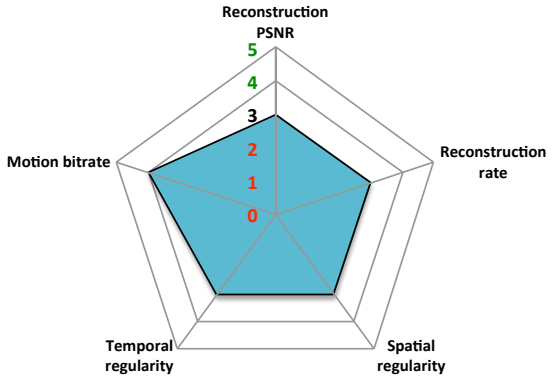
(b) *EPZS-like search strategy*



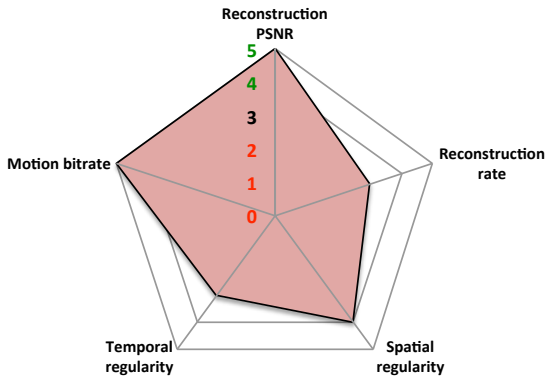
(c) *Rate-distortion optimization*



(d) *Causal & anticausal estimation directions*



(e) *Temporal prediction mechanisms*



(f) *Variable-size* motion tubes

Figure 5.8.1: *A shallow representation of the abilities of the different mechanisms introduced within this chapter*

Figure 5.8.1 summarizes the impact that provided regularization mechanisms have on each of these for criteria. They are graded on a scale from 0 to 5, where grade 3 corresponds to no impact at all. Grades shorter than 3 hence correspond to a worsening impact, while grades higher than 3 correspond to an improving impact. According to the targeted use and to the sequences, following charts may be used to select which mechanism should or should not be used.

Still, it can be seen from synthesized images that *motion tubes* are not able to represent the whole images: their reconstruction is scattered with numerous *holes* (*i.e.* unpredicted areas). Also, there is no way to control whether a *motion tube* should be used or not, whichever its abilities to represent the corresponding spatio-temporal area. Chapter 6 will provide additional techniques dedicated to the completion of the reconstructed images, while chapter 7 will discuss the quality assessment of the *motion tubes*, such that those unable to provide good enough synthesized textures are removed from their family.

Chapter 6

Improving synthesized textures using motion tubes

Building on the motion model provided in chapter 5, image sequences can now be partially represented from a set of *motion tubes* evolving across time and space. Synthesized images, however, suffer from an obvious phenomenon: unpredicted areas introduce a large number of *holes* in the reconstructed images. Two scenarios can be easily distinguished: small and large unpredicted areas. As capable is the motion model, some complex deformations may not be appropriately represented, thus introducing a drift in the motion field. As a consequence, it scatters the reconstruction with relatively small holes, lying in between several reconstructed areas which should have been kept connected. On the other hand, larger holes are often synonymous with a lack of *motion tubes*: whenever a piece of texture is unavailable at the reference instant, it cannot be tracked across the GOP as it is not hold by any *motion tube*.

Another problem is the crudeness of the current textural model. In chapter 5, the textural information $\mathcal{T}$ was only sourced from the reference image I_0, and kept still across the whole GOP. Such a static textural model cannot reflect the changes undergone by the textural information across time, which include resolution and illumination changes. Not only may this introduce resolution losses and illumination mismatches, but it may also bias the motion estimation.

Whichever way, all these limitations dramatically reduce the quality of the synthesized images. This chapter will be dedicated to these issues, and will propose various solutions. Section 6.1 will first precisely identify the artefacts induced by the representation. Then, section 6.2 will introduce an intra-tube mechanism which enables the textural information to evolve across time: *B-tubes*. Next section 6.3 will investigate how the representation can be improved by using several families of *motion tubes*, and will define the notion of *B-families* of *motion tubes*. Later, section 6.4 will provide a set of low-computational inpainting algorithms which will be used to fill remaining holes in the reconstructed images. Finally, section 6.5 will conclude the chapter.

6.1 Limits of the tube-based representation

Despite all the efforts put into the construction of a simple but capable and effective motion model for the *motion tubes*, previous chapter highlighted how much synthesized images are scattered with numerous unpredicted areas. Even the smallest ones critically affect the overall perception of the images: the eye is systematically attracted towards these regions. Not only does this make difficult to visually assess the quality of the reconstruction, but it also prevents the *motion tubes* from being used in practice. Section 6.1.1 will further investigate these issues. The inability to handle texture changes significantly reduces the fidelity of the representation, and may even prevent *motion tubes* from being tracked properly. Section 6.1.2 will further investigate how it actually impacts the images.

6.1.1 Incomplete reconstruction of the image sequences

Let us start with a simple observation: figure 6.1.1 shows how much synthesized images are scattered with numerous unpredicted areas, or *holes*. Not only reconstructed images are unusable, but these holes also complicate both objective and subjective quality assessment. From an objective perspective, this forced us, in previous chapter, to consider both the PSNR of the reconstructed areas and the reconstruction rate to numerically assess the images quality. From a subjective

point of view, unpredicted areas have a strong influence on the viewer: the eyes are inevitably attracted towards the boundaries between synthesized and non-synthesized areas, as the HVS pays great attention to edges. The visual masking effect, in particular, is mostly responsible for the large importance attached to these boundary areas.

Also, it appears from figure 6.1.1 that the *holes* dimensions can range from only a few pixels to large image areas. Following this observation, it is proposed to distinguish two scenarios:

- any **patch of texture** which is **unavailable** at the **reference instant** cannot be registered by any *motion tube*. May this happen, it is most likely to be responsible for the largest holes. As a consequence, it is necessary to initialize additional *motion tubes* at appropriate time instants and locations to account for the missing spatio-temporal information;
- whenever the provided **motion model** describes the displacements or the deformations too **crudely**, it is very likely for **local mismatches** to occur. In particular, this can be responsible for most small holes, lying in between two patches of textures which have been dragged apart, but should not have. In such case, one can make the assumption that the missing textural information is highly correlated to the textural information it is surrounded by. Hence, texture and/or structure synthesis algorithms are very indicated as ways to fill these small holes.

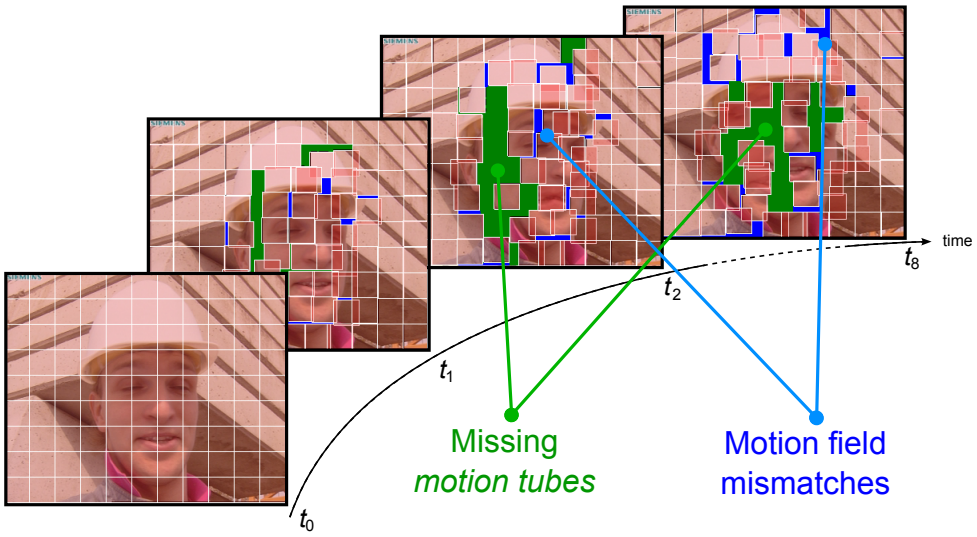


Figure 6.1.1: *Holes in the synthesized images: missing tubes and deformation mismatches*

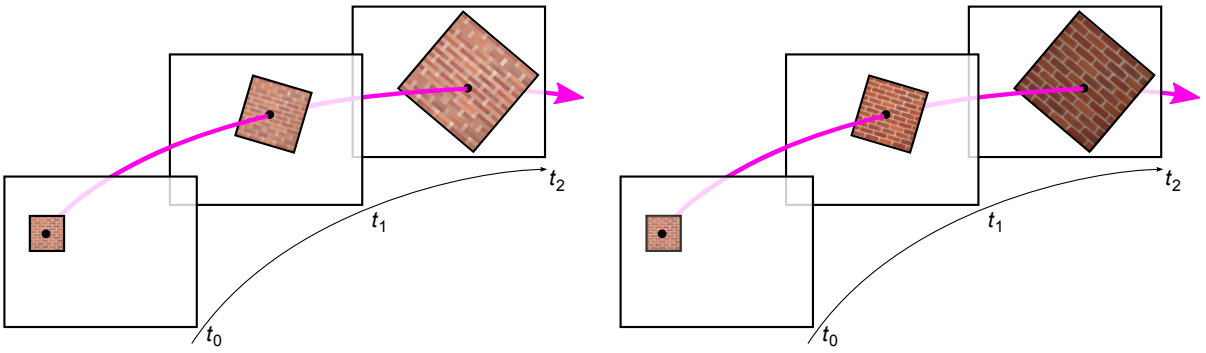
In order to provide a complete spatio-temporal representation of the image sequences, it is proposed to initialize several families of *motion tubes* at different time instants to describe a single GOP. Such a mechanism is introduced in section 6.3. This way, textures which are occulted at a given reference instant will be registered by another family of *motion tubes* initialized at another reference instant. Remaining holes will be filled by an inpainting mechanism introduced in section 6.4.

6.1.2 Time-evolving textural information

Until now, it has been assumed that the textural information is persistent throughout time. Consequently, chapter 5 made the assumption that *motion tubes* simply undergo a geometrical deformation in time and space. As a consequence, their textural information was expected to be static along their whole lifespan $\mathcal{L}$.

$$\forall t \in \mathcal{L}, \quad \mathcal{T}(t) = \mathcal{T} \tag{6.1}$$

In practice, this assumption proves to be inaccurate: due to illumination changes, the overall aspect of a piece of texture can greatly change. In addition, any deformation requiring the reference patch of texture to be stretched will result in resolution losses: synthesized textures will be blurred. Figure 6.1.2 illustrates both resolution losses and illumination changes. In figure 6.1.2a, a static texture is projected in all three instants: provided synthesis quality is rather poor. On the other hand, figure 6.1.2b shows that whenever the textural information is appropriately evolving throughout time, the synthesized images are much more accurate.



(a) *Static textural information* **(b)** *Variable textural information*

Figure 6.1.2: *Time-static versus time-evolving textural information*

Whereas a large number of *motion tubes* can be synthesized from static textures, some others will require the textural information to be changed across time. Which mechanisms should be proposed to provide a time-evolving textural information? In this connection, it is essential to remember some of our initial requirements: a low amount of information to transmit, and a low computational complexity. As a first step towards a time-evolving representation of the textural information of the *motion tubes*, section 6.2 will introduce a multi-prediction mechanism in charge of computing the texture $\mathcal{T}$ from a weighted sum of several reference textures.

6.2 Towards a time-evolving representation of the textures for the *motion tubes*

The evolution of a patch of texture across time is not only defined by its displacement and deformation, but also by its intrinsic textural changes. Numerous ways to represent the textures have been provided throughout the years; in particular, model-based approaches (see section 1.1.2.1) proved to be quite efficient when it comes to parametrize their changes, and could easily improve the *motion tubes*. However, this thesis does not focus on intra-coding mechanisms and did not thoroughly investigate this potential improvement axis. Still, it is important for the textural changes to be accounted for, even crudely.

As an intermediate solution towards an accurate time-evolving representation of the textures, it is proposed to source the textural information from several reference images. In such case, the final texture results from a weighted average of several temporally-aligned patches of textures. *Motion tubes* benefiting from several texture reference instants are called *bi-predictive tubes*, or shortly, *B-tubes*. In its simplest version, the mechanism only considers two reference instants, hence its name.

6.2.1 Bi-predictive motion tubes

Let $\{I_0, \ldots, I_8\}$ be a GOP of $G_S = 9$ consecutive images. Until now, a single reference image $I_{\mathbf{ref}}$ (I_0 in chapter 5) was expected to be available at the decoder side. From now on, it is assumed that two reference images $I_{\mathbf{ref0}}$ and $I_{\mathbf{ref1}}$ are available. Typically, these can be located at the extremities of the GOP, such that $I_{\mathbf{ref0}} = I_0$ and $I_{\mathbf{ref1}} = I_8$.

6.2.1.1 Sourcing the textural information from two reference images

It is first assumed that the displacement and the deformation $w_{\text{ref0}\to\text{ref1}}$ of a *motion tube* $\mathbf{X}$ between both reference instants is known. As a consequence, $\mathbf{X}$'s texture $\mathcal{T}_{\mathbf{X}}(t)$ can be synthesized at any time instant from the GOP by a weighting average of its textures at both reference instants. Thus, $\forall t \in [t_0, t_8]$,

$$\overline{\mathcal{T}_{\mathbf{X}}}(t) = \overline{I_t}\left(\mathbf{\Omega}_{\mathbf{X}}(t)\right) = \begin{cases} \mathcal{T}_{\mathbf{X}}(t_{\text{ref0}}) = I_{\text{ref0}}\left(\mathbf{\Omega}_{\mathbf{X}}(t_{\text{ref0}})\right) & \text{if } t = t_{\text{ref0}} \\ \mathcal{T}_{\mathbf{X}}(t_{\text{ref1}}) = I_{\text{ref1}}\left(\mathbf{\Omega}_{\mathbf{X}}(t_{\text{ref1}})\right) & \text{if } t = t_{\text{ref1}} \\ \dfrac{\alpha_{\text{ref0}}(t)\cdot\mathcal{T}_{\mathbf{X}}(t_{\text{ref0}}) + \alpha_{\text{ref1}}(t)\cdot\mathcal{T}_{\mathbf{X}}(t_{\text{ref1}})}{\alpha_{\text{ref0}}(t) + \alpha_{\text{ref1}}(t)} & \text{elsewhere} \end{cases} \tag{6.2}$$

where $\alpha_{\text{ref0}}(t) = 8 - |t - t_{\text{ref0}}|$ and $\alpha_{\text{ref1}}(t) = 8 - |t - t_{\text{ref1}}|$ weights both textural contributions depending on the distance between current and reference instants. The farther the reference instant t_{ref} is from the current instant t (inside the GOP), the shorter $\alpha_{\text{ref}}(t)$ is. Using this linearly interpolated textural information, *motion tubes* may then be able to sustain smooth texture changes, thus improving the overall reconstruction quality. This mechanism is similar to the weighted prediction provided by H.264/AVC video compression standard.

6.2.1.2 A multi-referenced textural information

As explained previously, *B-tubes* source their textural information from more than one reference instants. In this way, existing AS video compression schemes similarly use several reference instants to synthesize the images [TZ94, WXCM99, Cam04, LG08]; they have been briefly reviewed in section 3.4.2.1. In particular, works from Cammas [Cam04] and Le Guen [LG08] strongly inspired the bi-predictive mechanism provided by the *B-tubes*.

All these video compression schemes, however, suffer from resolution losses, as they use a single reference instant in the end: textures from the many different reference instants are first projected on a *main* reference instant; the different contributions are then merged and projected into the current instant to obtain the final synthesized image. As textures undergo two successive motion compensations, projected textures often suffer from resolution losses. Halfway through a GOP $\{I_0, \dots, I_8\}$, for instance, [Cam04] predicts the texture $\mathcal{T}(t_4)$ of image I_4 from both images I_0 and I_8. I_8's texture is first projected in t_0 onto $w_{8\to0}(\mathcal{T}(t_8))$; both I_0 and $w_{8\to0}(\mathcal{T}(t_8))$'s textures are then projected in t_4, and

$$\overline{\mathcal{T}}(t_4) = w_{0\to4}\Big(1/2 \cdot \mathcal{T}(t_0) \quad + \quad 1/2 \cdot w_{8\to0}(\mathcal{T}(t_8))\Big) \tag{6.3}$$

Motion tubes, however, do not suffer from this problem, as there is no *main* reference instant. Textures are motion compensated a single time only, which reduces the risks of resolution losses. Indeed, it was seen in chapter 5 that *motion tubes* warping operators could be easily composed and inverted, such that any image from any instant can be directly motion compensated at any other time instant. From this perspective, our approach is similar to motion compensated lifting approaches (*i.e.* MCTF). Using *motion tubes*, equation (6.3) is now simplified into

$$\begin{aligned}\overline{\mathcal{T}}(t_4) &= 1/2 \cdot w_{0\to4}\left(\mathcal{T}(t_0)\right) &+ \quad 1/2 \cdot w_{8\to4}\left(\mathcal{T}(t_8)\right)\\ &= 1/2 \cdot w_{0\to4}\Big(I_0\left(\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_0)\right)\Big) &+ \quad 1/2 \cdot w_{8\to4}\Big(I_8\left(\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_8)\right)\Big)\end{aligned} \tag{6.4}$$

6.2.2 Hierarchical bi-predictive motion tubes

Until now, it has been assumed that one or two reference images were available on the decoder side. In other words, an intra-coding mechanism needs to be set up aside from the tube-based motion compensation mechanism. It is now assumed that this intra-coding mechanism also sends residual information which enables the tube-based representation to be completed and corrected.

In section 5.5.1 of previous chapter, a hierarchical motion bi-prediction mechanism was provided and relied on a hierarchical GOP structure. It is proposed to apply the exact same mechanism to the texture synthesis. For each time instant and corresponding temporal layer, reference textures are sourced from neighbouring instants in the previous temporal layers. $\mathcal{T}(t_4)$ is now predicted from $\mathcal{T}(t_0)$ and $\mathcal{T}(t_8)$, such that

$$\overline{\mathcal{T}}(t_4) = 1/2 \cdot w_{0\to4}\left(\mathcal{T}(t_0)\right) \quad + \quad 1/2 \cdot w_{8\to4}\left(\mathcal{T}(t_8)\right) \tag{6.5}$$

Then, (resp.) $\mathcal{T}(t_2)$ and $\mathcal{T}(t_6)$ are similarly predicted from (resp.) $(\mathcal{T}(t_0), \mathcal{T}(t_4))$ and $(\mathcal{T}(t_4), \mathcal{T}(t_8))$:

$$\begin{cases}\overline{\mathcal{T}}(t_2) = 1/2 \cdot w_{0\to2}\left(\mathcal{T}(t_0)\right) &+ \quad 1/2 \cdot w_{4\to2}\left(\mathcal{T}(t_4)\right)\\ \overline{\mathcal{T}}(t_6) = 1/2 \cdot w_{4\to6}\left(\mathcal{T}(t_4)\right) &+ \quad 1/2 \cdot w_{8\to6}\left(\mathcal{T}(t_8)\right)\end{cases} \tag{6.6}$$

Doing this, the textural information changes are even more precisely accounted by the *motion tubes*, and one can expect a significant increase in reconstruction quality. Next section will confirm whether or not both simple and/or hierarchical *B-tubes* actually improve synthesized images.

6.2.3 Tubes versus B-tubes: compared performances

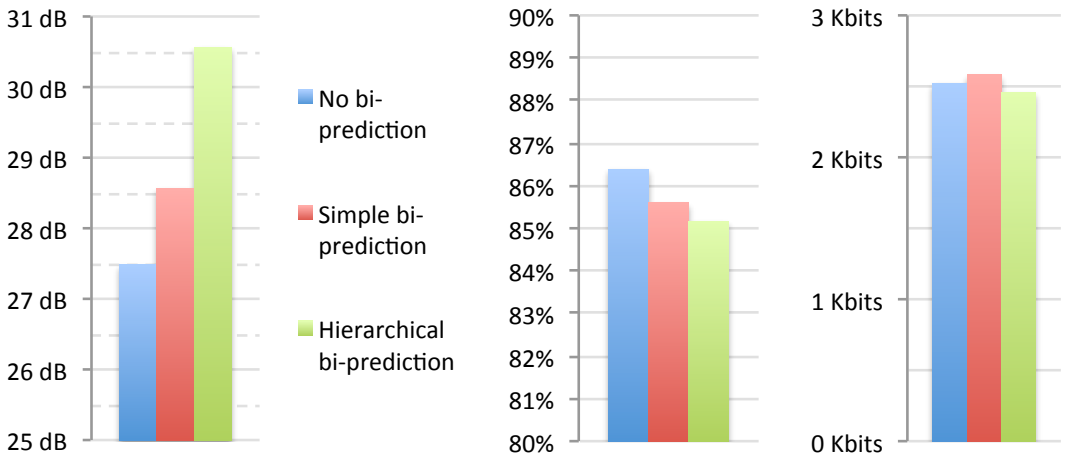
6.2.3.1 Setting up the experiments

In order to evaluate hypothetical improvements *B-tubes* can bring, three series of experiments have been performed. For each of them, only the translational TMC motion mode was enabled to describe the displacements of 16×16 *motion tubes*. A hierarchical GOP structure was used in order to easily put in place *B-tubes*. In order to guarantee for the motion information to be regular enough, the search area has been enlarged to 17 by 17 pixels, and both causal and anticausal motion estimation steps have been performed.

In the first series of experiments, no bi-prediction was used;. In the second series of experiments, then, a simple bi-prediction mechanism has been activated, such that both reference images are located at the extremities of the GOP: $I_{\text{ref0}} = I_0$ and $I_{\text{ref1}} = I_8$. Finally, a third series of experiments performed a hierarchical bi-prediction as was earlier explained in section 6.2.2.

6.2.3.2 Impact on the PSNR, the reconstruction rate and the motion bitrate

As can be seen from figure 6.2.1, *B-tubes* actually provide a significant increase in PSNR. While the simple bi-prediction mechanism is responsible for an increase of 1.08 dB, the hierarchical bi-prediction even more increases the PSNR by 3,08 dB. The reconstruction rate and the motion cost are barely affected by the bi-prediction. The reconstruction rate is decreased, at worst, by 1.22%, while the motion cost is increased by 2.32% with the simple bi-prediction, and decreased by 2.61% with its hierarchical variant. Provided that the intra-coding mechanism efficiently compacts the extra textural information, *B-tubes* seem to be source of great improvements in terms of synthesized quality.



(a) *Reconstruction PSNRs* **(b)** *Reconstruction rates* **(c)** *Motion bitrate*

Figure 6.2.1: B-tubes*: a conclusive improvement in reconstruction quality*

6.2.3.3 Visual results

Figure 6.2.2 shows how much the bi-prediction improves the quality of reconstruction. Foreman's eyes, in particular, are much more precisely rendered when hierarchical *B-tubes* are being used. More generally, the overall blurring effect which can be observed on Foreman's face is drastically reduced, and blocking artefacts as well. This confirms what objective measurements previously suggested, and further validates the concept of *B-tubes*. Further results can be found in section D.1 of appendix D.

(a) *No bi-prediction* **(b)** *Bi-prediction* **(c)** *Hierarchical bi-prediction*

Figure 6.2.2: *Tubes versus* B-tubes*: visual results*

6.3 B-families of motion tubes: how to combine several families of motion tubes

At this point, *motion tubes* mainly suffer from their inability to provide a complete reconstruction of the images. Small and large holes were previously distinguished; this section will focus on the first ones. Large reconstruction holes are especially problematic, as they generally correspond to textural areas which are unavailable at the reference instant. They can be the consequence of occlusions, camera motions, etc. As a solution, it is proposed to include additional *motion tubes* to describe these areas. Obviously, these *motion tubes* need to be instantiated from a frame which actually contains the textures missing to the spatio-temporal representation. This section will show how additional families of *motion tubes* may greatly improve the representation, and introduce the concept of *B-families* of *motion tubes*, as an inter-tube bi-prediction mechanism.

6.3.1 B-families of motion tubes: temporally overlapping the motion tubes

Often, the duration of a GOP (0.26 seconds for a GOP of 8 images at 30 frames per second) is negligible in comparison to the average lifespan of an object from the scene. Provided that the background itself is not undergoing massive changes during such a short period of time, it is very likely that the largest part of the textural information can be found either in the first image or in the last image of the GOP. From a *motion tube* perspective, this means that describing a GOP through two families of *motion tubes* may provide most of the spatio-temporal information needed to fully synthesize the sequence. A first family is instantiated on the first image of the GOP and tracked forwards towards its end, while a second family is instantiated on the last image of the GOP and tracked backwards towards its start. While any couple of frames taken from the GOP could be used to describe its spatio-temporal contents, it is quite intuitive to locate them as far from each other as possible: in other words, at the extremities of the GOP.

6.3.1.1 Instantiating *motion tubes* at the extremities of each GOP

As a first step towards an adaptive instantiation mechanism for the *motion tubes*, it is proposed to describe the spatio-temporal information with two families of *motion tubes*. ITU-T H.264/AVC and its predecessors defined the concept of B-frames: frames whose texture is predicted using two reference instants (in most cases, a previous frame and a future frame). Following this terminology, we introduce here the concept of *B-families* of *motion tubes*: a spatio-temporal structure that relies on two families of *motion tubes* to represent a GOP $\{I_0, \dots, I_8\}$. A first family $\mathcal{F}_{\mathcal{M}_T}(t_0)$ is referenced at time instant t_0 and tracked from t_0 to t_8. A second family $\mathcal{F}_{\mathcal{M}_T}(t_8)$ is referenced at time instant t_8 and tracked from t_8 to t_0. The couple ${}^0\mathcal{F}_{\mathcal{M}_T}(t_0, t_8) = \{\mathcal{F}_{\mathcal{M}_T}(t_0), \mathcal{F}_{\mathcal{M}_T}(t_8)\}$ is called a *B-family* of *motion tubes*.

a Synthesing images from two families of motion tubes

Let ${}^{\mathcal{F}}\bigcup$ be an operator used to compose the contributions of several families of *motion tubes* into a single image. It is defined as

$$ {}^{\mathcal{F}}\bigcup\Big[\mathcal{R}_a,\,\mathcal{R}_b\Big](x,y)=\begin{cases}\varnothing & \text{if } \mathcal{R}_a(x,y)=\varnothing \text{ and } \mathcal{R}_b(x,y)=\varnothing\\ \mathcal{R}_a(x,y) & \text{if } \mathcal{R}_a(x,y)\neq\varnothing \text{ and } \mathcal{R}_b(x,y)=\varnothing\\ \mathcal{R}_b(x,y) & \text{if } \mathcal{R}_a(x,y)=\varnothing \text{ and } \mathcal{R}_b(x,y)\neq\varnothing\\ \mathcal{R}_a(x,y)+\mathcal{R}_b(x,y) & \text{if } \mathcal{R}_a(x,y)\neq\varnothing \text{ and } \mathcal{R}_b(x,y)\neq\varnothing\end{cases} \tag{6.7}$$

Let $\left\{\mathcal{M}^{t_0}_{\mathcal{I}_i}\right\}_{i=1\ldots N_0}$ and $\left\{\mathcal{M}^{t_8}_{\mathcal{I}_i}\right\}_{i=1\ldots N_8}$ be respectively the members of families $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ and $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$. At time instant t, $t_0<t<t_8$, the synthesized image $\overline{I_t}$ is now given by

$$\begin{aligned}\overline{I_n} &= {}^{\mathcal{F}}\bigcup\Big[\quad \mathcal{R}\Big({}^1\!/_8\cdot\alpha_0(t)\cdot\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0),\,t\Big)\quad ,\,\mathcal{R}\Big({}^1\!/_8\cdot\alpha_8(t)\cdot\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8),\,t\Big)\quad\Big]\\ &= {}^{\mathcal{F}}\bigcup\left[\quad {}^1\!/_8\cdot\alpha_0(t)\cdot\bigcup_{i=1}^{N}\mathcal{R}\left(\mathcal{M}^{t_0}_{\mathcal{I}_i},w_{t_0\to t_n}\right),\,{}^1\!/_8\cdot\alpha_8(t)\cdot\bigcup_{i=1}^{N}\mathcal{R}\left(\mathcal{M}^{t_8}_{\mathcal{I}_i},w_{t_{G_S}\to t_n}\right)\quad\right]\end{aligned} \tag{6.8}$$

where $\mathcal{R}$ is the *motion tube* rendering operator and $\bigcup$ the *motion tube* composition operator. Coefficients $\alpha_0(t)=8-t$ and $\alpha_8(t)=t$ linearly weight the contributions of $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ and $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ according to the current time index inside the GOP. The closer from its reference instant, the more importantly a *motion tube* contributes to the reconstruction of the current image.

Again, the concept of *B-family* is not limited to the use of two families located at the extremities of the GOP, and any couple of reference instants can be used, along with appropriate weighting coefficients. Also, it is important to note that *B-tubes* and *B-families* of *motion tubes* are not in contradiction and can be combined. However, if two *motion tubes* $\mathcal{M}^{t_0}_{\mathcal{I}_i}\in\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ and $\mathcal{M}^{t_8}_{\mathcal{I}_i}\in\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ are undergoing similar trajectories and deformations, combining their contributions through *B-family* ${}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0,t_8)$ will have the same effect as using a single *B-tube*.

b Bi-directional motion tubes

As image $I_{8(p+1)}$ is both the last image from the p^{th} GOP $\left\{I_{8p},\ldots,I_{8(p+1)}\right\}$, and also the first image from the $p+1^{\text{th}}$ GOP $\left\{I_{8(p+1)},\ldots,I_{8(p+2)}\right\}$, its textural information is shared by two families of *motion tube*:

1. the family instantiated at the end of the p^{th} GOP, tracked backwards from $t_{8(p+1)}$ to t_{8p};
2. the family instantiated at the beginning of the $p+1^{\text{th}}$ GOP, tracked forwards from $t_{8(p+1)}$ to $t_{8(p+2)}$.

Both families can then be merged into a single family of *motion tubes* which is initialized at $t_{8(n+1)}$, and tracked both backwards and forwards in time. By successively concatenating such a temporal construction, one ends up in bi-directional families of *motion tubes* as it is showed in figure 6.3.1.

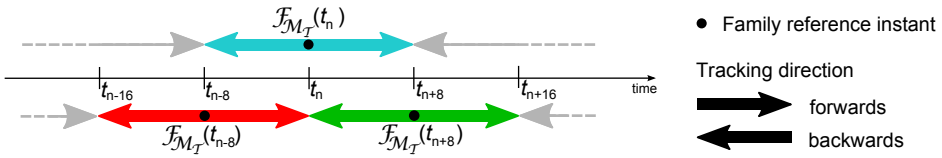


Figure 6.3.1: B-families *of* motion tubes, *a structure based on the temporal overlap of two families of* motion tubes

6.3.1.2 Hierarchically overlapping the families of *motion tubes*

B-families of *motion tubes* rely on the assumption that the entire textural information can actually be sourced from only two reference instants, which needs for the GOP spatio-temporal content to be quite consistent. At some point, however, *motion tubes* will not be able to handle highly non-stationary sequences; in sequence *Tempest*, for instance, the fast moving flying leaves. Still, additional families of *motion tubes* may further complete the reconstruction. This section will now extend the concept of *B-families* of *motion tubes* into a hierarchical inter-tube bi-predictive structure.

a Hierarchical B-families of motion tubes

Once again, let consider a GOP $\{I_0, \ldots, I_8\}$ of nine consecutive images, and let it be organized into a set of hierarchical temporal levels. For each temporal level, it is proposed to build a *B-family* consisting of the *B-family* of the previous temporal level and an additional family instantiated at a new reference instant taken from the current temporal level. This process is illustrated in figure 6.3.2. Four *B-families* are successively obtained:

1. ${}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_8) = \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0) \cup \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ when $t_0 < t < t_8$ (first temporal level);
2. ${}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8) = {}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_8) \cup \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$ when $t_0 < t < t_8$ (second temporal level);
3. ${}^2\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_2, t_4, t_8) = {}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8) \cup \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_2)$ when $t_0 < t < t_4$ (third temporal level);
4. ${}^2\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_6, t_8) = {}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8) \cup \mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_6)$ when $t_4 < t < t_8$ (third temporal level).

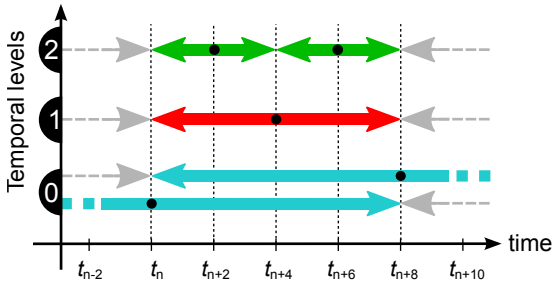


Figure 6.3.2: *Hierarchical* B-tubes*: overlapping lifespans*

Family	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8
$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$	×	✓	✓	✓	✓	✓	✓	✓	✓
$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$	×	✓	✓	✓	✓	✓	✓	✓	×
$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$	×	✓	✓	✓	×	✓	✓	✓	×
$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_2)$	×	✓	×	✓	×	–	–	–	–
$\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_6)$	–	–	–	–	×	✓	×	✓	×

Table 6.3.1: *Families used to synthesize the images*

b How to combine several B-families of motion tubes

While hierarchically increasing the number of families of *motion tubes* undoubtedly improves the overall representation, it also drastically increases the amount of multi-registered areas, hence the amount of spatio-temporal redundancies. In order to control the increase in motion information, essential *motion tubes* should only be kept in the final representation. However, building an optimal set of *motion tubes* is a complex problem which will be tackled in chapter 7.

Still, it is critical for the hierarchical structure of families of *motion tubes* to require as few *motion tubes* as possible. As a first solution towards an optimal hierarchical structure, it is proposed to synthesize the images in a hierarchical fashion as well. Let us now focus on the reconstruction of an image I_t, $t_0 < t < t_4$, from the usual GOP $\{I_0, \ldots, I_8\}$. The hierarchical set of families contributing to its reconstruction is given by *B-family* ${}^2\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_2, t_4, t_8)$. I_t's reconstruction is progressively built, one temporal level after the other, as follows:

1. I_t is first reconstructed from *B-family* ${}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_8)$, into ${}^0\overline{I}_t$. The contributions of families $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ and $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ are combined with respect to equation (6.8);
2. in areas not yet reconstructed, the reconstruction of family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$ is used:

$$ {}^1\overline{I}_t(x,y) = \begin{cases} {}^0\overline{I}_t(x,y) & \text{if } {}^0\overline{I}_t \neq \varnothing \\ \mathcal{R}\left(\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4), t\right)(x,y) & \text{elsewhere} \end{cases}; \tag{6.9} $$

3. for areas not reconstructed by ${}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8)$, the reconstruction of family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_2)$ is used:

$$ {}^2\overline{I}_t(x,y) = \begin{cases} {}^1\overline{I}_t(x,y) & \text{if } {}^1\overline{I}_t \neq \varnothing \\ \mathcal{R}\left(\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_2), t\right)(x,y) & \text{elsewhere} \end{cases}. \tag{6.10} $$

This progressive reconstruction spares us from an ambiguous operation: how to play the reconstruction from each temporal level against each other? Provided that useless, inefficient and redundant *motion tubes* have been removed from each family, this reconstruction mechanism should be reassessed in order to use all the available textural information, instead of discarding contributions from higher temporal levels whenever the current level already contributes to the reconstruction. As no selection mechanism has been designed yet, such an improvement will remain as a perspective.

6.3.2 Single versus multiple families of *motion tubes*: compared performances

6.3.2.1 Setting up the experiments

a Reconstructing a GOP from one, two or three families of motion tubes

In order to evaluate the improvements *B-families* of *motion tubes* bring in practice, three series of experiments have been performed on the usual set of six sequences. At first, a single family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ of *motion tubes* have been used to describe the spatio-temporal contents of each GOP $\{I_0, \ldots, I_8\}$. Then, a *B-family* ${}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_8)$ consisting of two families of *motion tubes* has been employed. Finally, a *B-family* ${}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8)$ consisting of three families has been used. In each case, a crude set of parameters have been used to set up *motion tubes*: only TMC motion mode has been enabled and fixed-size 16×16 *motion tubes* were used. A hierarchical GOP structure, however, was used.

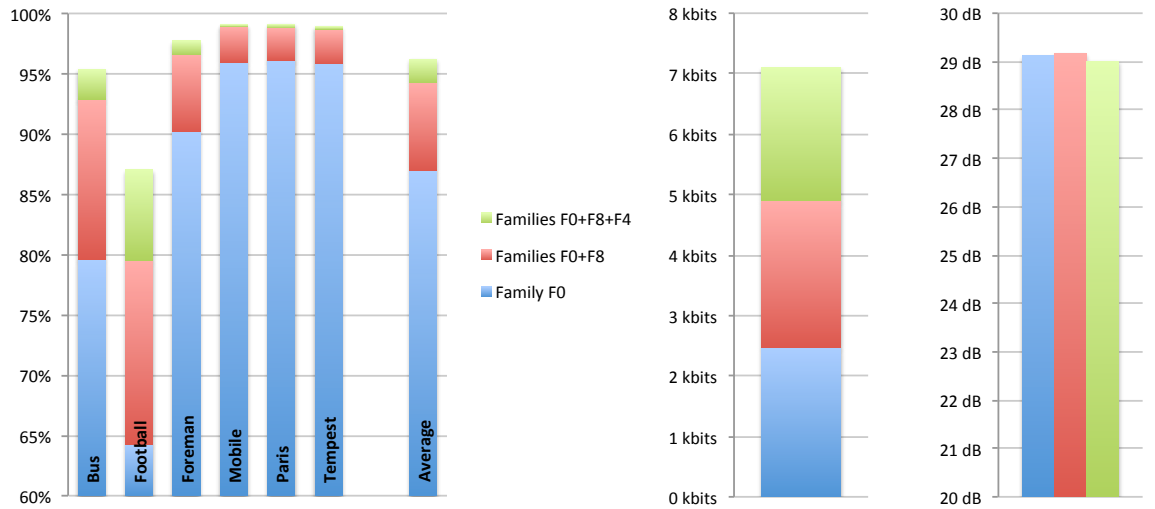
b A focus on reference images: biased results

Whenever a family of *motion tubes* was instantiated, it has been assumed that the current image was entirely available, and properly synthesized for it to be used as a reference frame. However, *motion tubes* do not include an intra-coding mechanism yet. Corresponding coding costs will not be assessed, and output bitrate will only reflect the evolution of the motion bitrate. Chapter 7 will provide a fully working compression scheme which, among other topics, handles this additional information.

Experiments, however, sourced the textural information from the original frames, such that any reference frame could be perfectly reconstructed from the corresponding family of *motion tubes*. As experiments results might have a significant bias by taking these perfectly reconstructed areas into account, it has been decided to not use a family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ at its reference instant t_{ref}. Corresponding frame I_{ref} will then be synthesized from other available families, such that provided results purely assess the reconstruction abilities of the *motion tubes*. The hypothetical intra-coding mechanism is later assumed to transmit textural residues to complete I_{ref}'s reconstruction, which is required to instantiate $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$. Table 6.3.1 lists the different families of a *B-family* ${}^2\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_2, t_4, t_6, t_8)$ and they contribute or do not contribute to the reconstruction of the current image.

6.3.2.2 Objective performances: influence on the PSNR, the reconstruction rate and the motion bitrate

Figure 6.3.3 shows how the reconstruction rate is significantly improved by the use of a *B-family* ${}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8)$ instead of a single family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$. In average, using both families $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ and $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ increases the reconstruction rate by 7%. It is further improved by 2% from the third family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$. The PSNR of the reconstructed areas is barely affected by the number of families in use: they all provide a similar quality of reconstruction.



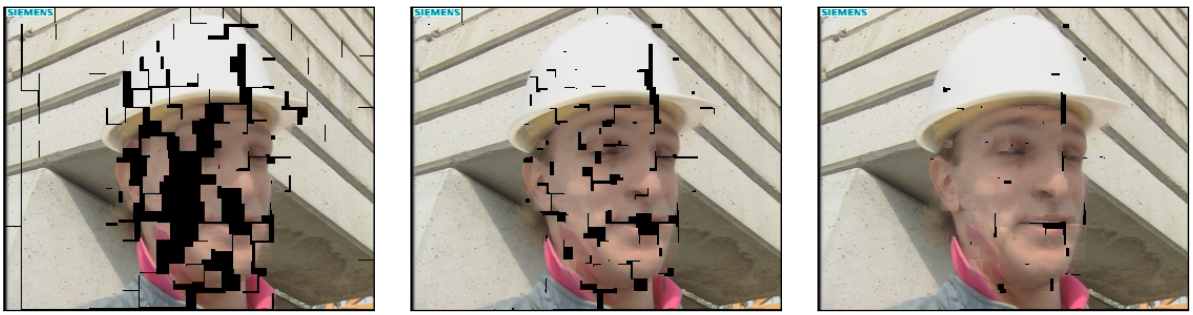
(a) *Contribution of each family to the reconstruction rate* **(b)** *Average motion bitrates* **(c)** *Average rec. SPSNRs*

Figure 6.3.3: B-families *of* motion tubes*: a conclusive improvement in reconstruction rate*

However, this comes to the cost of a significant increase in motion bitrate: as unused *motion tubes* have not been discarded yet, increasing the number of families linearly increases the motion bitrate (figure 6.3.3b shows that, unsurprisingly, they all allocate approximately the same bitrate per time instant). Still, the additional families $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$ and $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ only contribute to 11% of the reconstruction: the number of their *motion tubes* actually being used is very likely to be small, hence requiring a significantly lower motion bitrate to be transmitted in comparison to the bitrate allocated by the whole families.

6.3.2.3 B-families of motion tubes: visual results

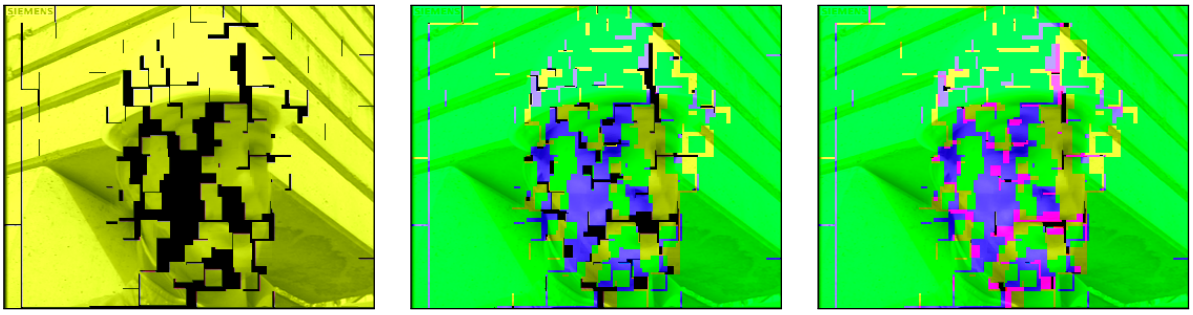
Figure 6.3.4 shows how *B-families* of *motion tubes* drastically improve the representation. Top figures show the output images, while bottom figures indicate which families of *motion tubes* have contributed to the reconstruction: yellow for $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$, blue for $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$, green for a weighted average of $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ and $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$, and finally magenta for $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$. Let us focus on Foreman's face: due to its rotation towards the right side of the camera plane, its left half is mostly visible within the second half of the GOP, while its right half is mostly visible on the first half of the GOP.



(a) $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$*'s reconstruction*
SPSNR = 35.79 dB, Rec. rate = 86.81%

(b) ${}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_8)$*'s reconstruction*
SPSNR = 36.52 dB, Rec. rate = 96,26%

(c) ${}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8)$*'s reconstruction*
SPSNR = 36.24 dB, Rec. rate = 99.29%



(d) *Areas reconstructed by* $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$

(e) *Areas reconstructed by* ${}^0\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_8)$

(f) *Areas reconstructed by* ${}^1\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0, t_4, t_8)$

Figure 6.3.4: B-families *of* motion tubes*: influence on the synthesis of the fourth image of sequence* Foreman

As could be expected, family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_0)$ mainly contributes to the reconstruction of the right half of the face, while family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_8)$ mainly contributes to the reconstruction of its left half. Both of these families have been able to catch the deformation of the background, which is mostly synthesized from a weighted average of both contributions. Finally, the third family $\mathcal{F}_{\mathcal{M}_\mathcal{T}}(t_4)$ nearly completes the reconstruction, by notably providing a large part of the mouth which was missing.

6.4 Towards a complete reconstruction of each image

Despite the significantly larger amount of reconstructed areas that can be obtained through *B-families* of *motion tubes*, a small but non-negligible part of the images remains unpredicted. Provided that most textures were available in any of the reference images, remaining holes are very likely to be quite small, and the consequence of local discrepancies or mismatches of the motion information. This can be observed in figure 6.3.4c.

Another problem raised by the use of *B-families* of *motion tubes* is the risk of allocating entire *motion tubes* to fill tiny holes, which could be properly restored without any additional *motion tubes*, hence without suffering from an increase in motion bitrate. This phenomenon can be observed in figures 6.3.4a and 6.3.4b: the background is scattered with unpredicted segments whose width is mostly limited to a single pixel. Naturally, a second family of *motion tubes* does remove these holes; however, this is very likely to require a significant number of additional *motion tubes* to be transmitted. In other words, the additional *motion tubes*, in these areas, can be interpreted as a waste in allocated bitrate.

For both these reasons, and as a last resort mechanism, it is now proposed to handle tiny holes and remaining unpredicted areas through several image restoration algorithms, capable of synthesing both structure and textures information from surrounding available reconstructed areas. Such algorithms (see section 3.4.2.2) are often very computationally demanding; for this reason, provided inpainting mechanisms will be relatively crude and rather simple to perform. Section 6.4.1 will briefly overview a simple spatial inpainting mechanism based on the popular median filter. Then, section 6.4.2 will introduce a spatio-temporal template matching algorithm based on the *motion tubes*.

6.4.1 A spatial inpainting mechanism based on the popular median filter

Many inpainting algorithms [BSCB00, CS01, CS02, Mas02] rely on the diffusion of available areas from the boundaries of the regions to inpaint towards their center. The diffusion may be isotropic or anisotropic; in the second case, available structural information (edges, gradients, dots) can be used to drive the diffusion. In [OBMC01], Oliveira showed that the variational diffusion process could be simplified into a series of simple iterative convolution operations, for a restoration quality nearly as high as standard diffusion algorithms provide.

The median filter has been widely used in many image processing applications; in particular, is it very appreciated when it comes to predict a value from its neighbours as it is both simple and very effective. Incomplete images can be interpreted as noisy signals, and inpainting algorithms as noise reduction mechanisms. To this end, it is proposed to propagate reconstructed pixels into unpredicted areas through a series of median-based convolution operations.

6.4.1.1 An isotropic median filtering process

The inpainting process is independently performed on each color component (luminance Y, chrominances U and V). Each unpredicted sample is restored by the median value of the available samples taken from a local neighbourhood. The neighbourhood is given by the shape and the dimension of the median filtering window: a $(2N+1) \times (2N+1)$ square window. N is adjusted according to the color component, and $N = N_Y$ for the lumiance, $N = N_U = N_V = 1/2 \cdot N_Y$ for chrominances.

The proposed inpainting mechanism is thus isotropic as it does not favour any particular direction. Any missing sample will be reconstructed provided that its closest available neighbour is not farther than N samples. The larger the filter size gets, the higher the number of inpainted pixels is (see figures 6.4.1c and 6.4.1d). By locally adapting the shape of the median windows according to the edges directions, this inpainting mechanism could be easily improved, and turned into an anisotropic mechanism. Also, a spatio-temporal filtering window could be used to further improve its restoration abilities. These will remain as minor perspectives, as this thesis does not focus on texture and structure restoration.

6.4.1.2 Iterative diffusion of the restoration

As it has been mentioned in previous sub-section, the number of restored samples depends on the dimension of the median window. However, it is crucial for the window to be small enough for it to consider a local neighbourhood only. In order to complete the reconstruction, it is now proposed to iterate the convolution, thus progressively propagating the restoration towards the centre of the unpredicted areas.

The number of iterations required to complete the reconstruction depends on the dimensions of the unpredicted areas and the size $(2N+1) \times (2N+1)$ of the filtering window used. Let $\mathbf{\Omega}^S$ be the set of synthesized samples (their coordinates) from the image plane $\mathbf{\Omega}$. Conversely, let Ω^{NS} be the set of non-synthesized samples from the image plane $\mathbf{\Omega}$. $\mathbf{\Omega}^S$ and $\mathbf{\Omega}^{NS}$ are complementary in $\mathbf{\Omega}$, and $\mathbf{\Omega}^S \oplus \mathbf{\Omega}^{NS} = \mathbf{\Omega}$. The number of iterations n_{iter} required to complete the reconstruction is given by

$$n_{\text{iter}} = \frac{1}{N} \cdot \max_{P_1 \in \mathbf{\Omega}^{NS}} \left[\text{dist} \left(P_1, \quad \arg\min_{P_2 \in \mathbf{\Omega}^S} \text{dist}\,(P_1, P_2) \right) \right] \tag{6.11}$$

where dist(P_1, P_2) is the distance between positions P_1 and P_2. This may require a large number of iterations to be performed in large holes, at the expense of the overall complexity. Figure 6.4.1b shows the final inpainting restoration on the eighth image of sequence *Foreman*. Due to the crudeness of the inpainting algorithm, large inpainted regions suffer from an obvious blurring effect. As for the reconstruction PSNR, it is drastically reduced by 7dB. Yet, a single iteration is enough to fill most of the little holes the background is scattered with, for a reduction in PSNR much more acceptable.

(a) *Initially synthesized image*
SPSNR = 33.55 dB, Rec. rate = 80.31%

(b) *Final inpainted image, N = 3*
SPSNR = 26.56 dB, Rec. rate = 100%

(c) *Inpainted image after one iteration, N = 3*
SPSNR = 32.47 dB, Rec. rate = 85.26%

(d) *Inpainted image after one iteration, N = 5*
SPSNR = 31.42 dB, Rec. rate = 88.45%

Figure 6.4.1: *Median-based inpainting mechanism: a crude way to fill unpredicted areas*

6.4.2 A spatio-temporal inpainting algorithm relying on the motion tubes

The provided median-based inpainting suffers, among other things, from its inability to use temporally neighbouring areas to restore the unpredicted samples. Consequently, only spatial correlations were used to drive the inpainting process. *Motion tubes*, on the other hand, provide an accurate spatio-temporal representation of the image sequences. This knowledge can be advantageously used to improve the inpainting mechanism.

6.4.2.1 Texture and structure synthesis through template matching

In this context, it is proposed to fill unpredicted areas following a template matching approach relying on the spatio-temporal information provided by the *motion tubes*. Instead of propagating the inpainted values using a filtering mechanism

as the median-based inpainting does, this approach reconstructs the unpredicted areas using appropriate patches of textures (templates) sourced from the spatio-temporal neighbourhood of the areas to be inpainted. In particular, both textures and structures may be reconstructed.

Template matching algorithm, similarly to other texture and structure synthesis algorithms, are quite computationally demanding. Both the search for matching templates and their fusion with already available areas (*e.g.* graph cuts) are two complex tasks. As *motion tubes* provide an accurate representation of the spatio-temporal content of the image sequences, their trajectories can be advantageously used to drive the inpainting mechanism. This spares the inpainting process from both the search for matching templates, and the fusion step: the whole template matching mechanism is reduced to a series of simple block-based motion compensations.

The search for matching templates is then avoided by simply using the spatio-temporal motion information provided by the *motion tubes*: missing areas located in the neighbourhood of a *motion tube* $\mathcal{M}_\mathcal{T}$ are very likely to be similar to collocated areas in the reference frame, given $\mathcal{M}_\mathcal{T}$'s trajectory and deformation. Also, the fusion step is intrinsically performed through the motion compensation of these collocated areas: there will not be any textural discrepancies at the boundaries between reconstructed and inpainted areas.

6.4.2.2 Using the *motion tubes* to drive the template matching

a How temporally collocated areas are likely to be a good match for unpredicted ones

A *motion tube* tracks a patch of texture across several images. Consequently, surrounding pixels are very likely to undergo the same displacement and deformation. Whenever an unpredicted area is close enough from a projected *motion tube* $\mathcal{M}_\mathcal{T}$, it may be reconstructed from its collocated area in $\mathcal{M}_\mathcal{T}$'s reference frame, following $\mathcal{M}_\mathcal{T}$'s trajectory and deformation. Each *motion tube* can be associated with a matching template consisting of the surrounding area.

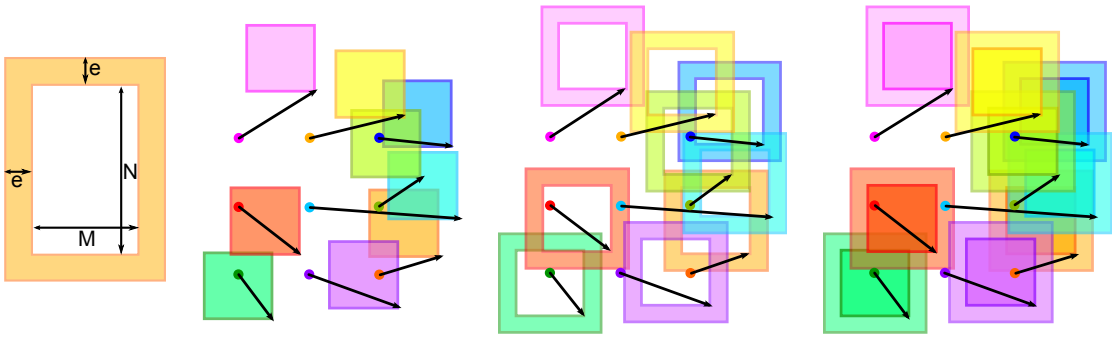
b Enlarging the *motion tubes* into overlapping patterns

Let $\mathcal{M}_\mathcal{T}$ be an $M \times N$ *motion tube* and t_{ref} its reference instant. $\mathcal{M}_\mathcal{T}$'s motion parameters are supposed to be available. It is proposed to enlarge $\mathcal{M}_\mathcal{T}$ by e pixels in each direction into a $(M+2e) \times (N+2e)$ *motion tube* (see figure 6.4.2a). The enlarged area, a border of thickness e pixels around $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$, plays the part of a matching template $\Upsilon_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$.

$$\Upsilon_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}}) = \sigma_e\left(\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})\right) \Delta \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}}) \tag{6.12}$$

where $\sigma_e(\mathbf{\Omega})$ is a stretching operator which enlarges the domain $\mathbf{\Omega}$ by e pixels, and Δ is the symmetric difference operator. $\Upsilon_{\mathcal{M}_\mathcal{T}}$ is used to inpaint the collocated areas at other time instants, following the trajectory and the deformation of the corresponding *motion tube*.

c Towards a spatio-temporal inpainting mechanism



(a) *Template* $\Upsilon_{\mathcal{M}_\mathcal{T}}$ **(b)** *Motion compensated tubes* **(c)** *Motion compensated templates* **(d)** *Final inpainted image*

Figure 6.4.2: *Tube-based template matching inpainting: a low computational inpainting algorithm*

Whenever and wherever unpredicted areas occur around projected *motion tubes*, corresponding templates are motion compensated as well to fill these areas. The motion compensated template $\widetilde{\Upsilon_{\mathcal{M}_\mathcal{T}}}(t_{\text{cur}})$ is given at time t_n by

$$\widetilde{\Upsilon_{\mathcal{M}_\mathcal{T}}}(t_{\text{cur}}) = w_{\text{ref}\to\text{cur}}\left(\Upsilon_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})\right) \tag{6.13}$$

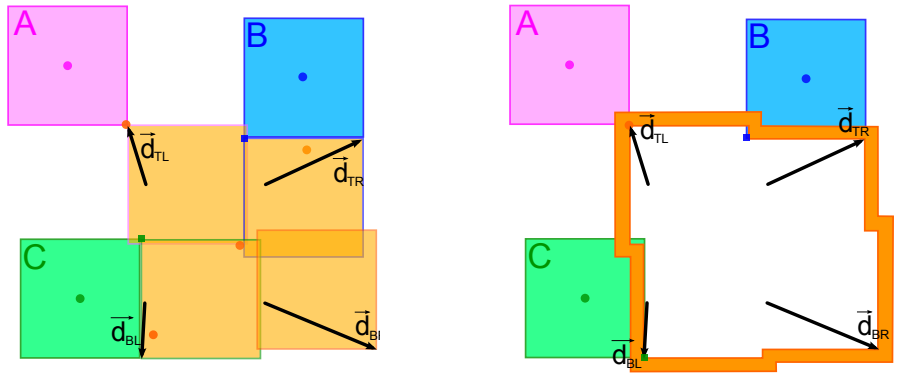
where $w_{\text{ref}\to\text{cur}}$ is the estimation of $\mathcal{M}_\mathcal{T}$'s deformation and displacement between t_{ref} and t_{cur}. This is illustrated in figure 6.4.2: the final image combines initially motion compensated *motion tubes* (figure 6.4.2b) and inpainted areas (figure 6.4.2c) to obtain the final image (figure 6.4.2d).

In the end, the provided template matching mechanism requires a lot less amount of computations, compared to classic template matching algorithms. However, in large unpredicted areas, some pixels may be located at more than e pixels from the closest *motion tube*, hence remain unpredicted. As a last resort, the median-based inpainting can be used to complete the reconstruction.

6.4.2.3 A focus on specific abilities the proposed inpainting mechanism inherits from the motion tubes

a A motion-adaptive template matching

The assumption that areas neighbouring the *motion tubes* should hold a textural information highly correlated to these *motion tubes* drove us to the proposed template-matching inpainting mechanism. As an extension to this idea, it may be considered that these neighbouring areas are also undergoing deformations similar to the ones undergone by the *motion tubes*. As matching templates are built from the available *motion tubes*, they also inherit most of their characteristics. As a consequence, they are also undergoing the deformations of the *motion tubes* they are build from -see equation (6.13)-.



(a) *OTMC: motion compensated* motion tube **(b)** *OTMC: motion compensated matching template*

Figure 6.4.3: *Shape-adaptativity of the template matching inpainting mechanism*

In figure 6.4.3: a *motion tube* $\mathcal{M}_\mathcal{T}$ is deformed through the OTMC motion mode (figure 6.4.3a); the corresponding matching template is accordingly deformed (figure 6.4.3b). In the end, this operation ensures the matching templates to be warped according to their corresponding *motion tubes*, such that the proposed inpainting mechanism takes into account both spatial and temporal correlations.

b How the textural information of the matching templates may be time-evolving

When performed on *B-tubes*, the template-based inpainting also benefits from the bi-prediction intra-tube mechanism. Indeed, as they undergo a similar motion compensation process, the textural information of the matching templates can be predicted from several time instants, following the GOP structure. In such case, equation (6.13) is modified into the following one:

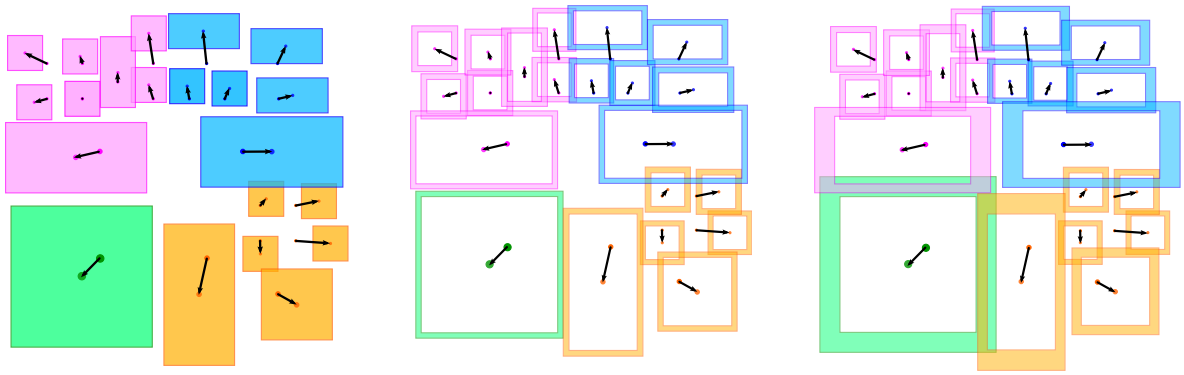
$$\widetilde{\Upsilon_{\mathcal{M}_\mathcal{T}}}(t_{\text{cur}}) = \frac{\alpha_{\text{ref0}}(t_{\text{cur}}) \cdot w_{\text{ref0}\to\text{cur}}\left(\Upsilon_{\mathcal{M}_\mathcal{T}}(t_{\text{ref0}})\right) + \alpha_{\text{ref1}}(t_{\text{cur}}) \cdot w_{\text{ref1}\to\text{cur}}\left(\Upsilon_{\mathcal{M}_\mathcal{T}}(t_{\text{ref1}})\right)}{\alpha_{\text{ref0}}(t_{\text{cur}}) + \alpha_{\text{ref1}}(t_{\text{cur}})} \tag{6.14}$$

where $\alpha_{\text{ref0}}(t)$ and $\alpha_{\text{ref1}}(t)$ are the weighting coefficients from equation (6.2).

c Towards a shape-adaptive template matching

When variable-size *motion tubes* are used, corresponding matching templates also have variable dimensions. Variable-size inpainting has already been proposed in the literature [DCOY03]. The thickness of our templates, yet, is fixed, as showed in figure 6.4.4b. As a perspective, though, one may adapt the thickness of the templates according to the dimensions of the *motion tubes* they are built from. Large *motion tubes* correspond to regions of uniform motion: surrounding regions

are also likely to undergo similar displacements and deformations. In these areas, one may use relatively thick templates. Conversely, small *motion tubes* correspond to regions where the motion field is much less coherent: surrounding regions are not very likely to undergo similar displacements or deformations. In these areas, one may use relatively thin templates. This improvement perspective is illustrated in figure 6.4.4c.



(a) *Variable-size* motion tubes **(b)** *Variable-size templates* **(c)** *An interesting perspective*

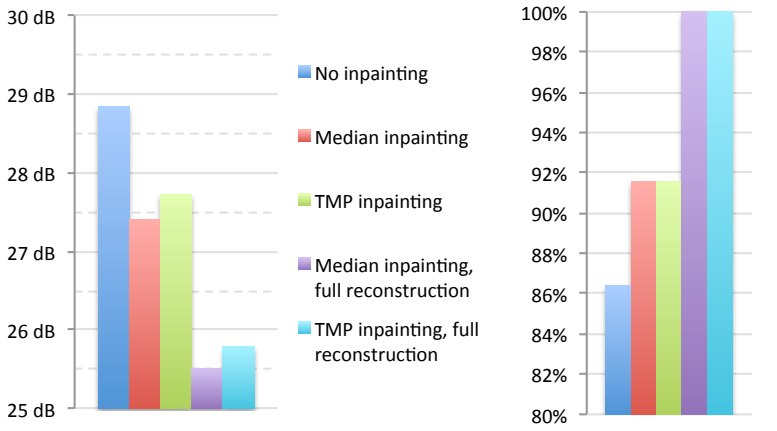
Figure 6.4.4: *Variable-size template matching: shape-adaptivity of the templates*

6.4.2.4 Compared performances of median-based and template-based inpainting mechanisms

a Setting up the experiments

Five series of experiments were performed on the usual set of six sequences. In each case, a crude set of parameters have been used to set up *motion tubes*: only TMC motion mode has been enabled and fixed-size 16×16 *B-tubes* were tracked across a hierarchical GOP. In the first series of experiments, no inpainting was performed. In the second and third series of experiments, median ($N = 2$) and template matching ($e = 2$) were respectively used to inpaint the images. The inpainting parameters (N and e) were set such that both inpainting mechanisms restore the same pixels. Finally, the fourth and the fifth series of experiments respectively used the same inpainting parameters as the second and the third series did, and performed additional iterations of median-based inpainting to complete the reconstruction.

b How the different inpainting mechanisms impact the PSNR and the reconstruction rate



(a) *Average SPSNR* **(b)** *Average reconstruction rates*

Figure 6.4.5: *Inpainting: a last resort mechanism to complete the reconstruction*

From figure 6.4.5, it appears that, for a constant reconstruction rate, template-based inpainting provides an increase in PSNR of 0.3 to 0.4 dB in comparison to the median-based inpainting. Still, these inpainting mechanisms are relatively crude and significantly decrease the reconstruction PSNR: the restoration of remaining unpredicted areas (here, 14% in average) diminishes the overall reconstruction PSNR by more than 3 dB.

However, these results need to be considered with caution: the second and the third series of experiments are much more representative of the true abilities of the inpainting. Indeed, inpainting is mostly dedicated to the restoration of small unpredicted areas, which have been mostly taken care of within these two experiments. In this case, the reduction in PSNR is only about 1.62 dB. Further results can be found in section D.2 of appendix D.

6.4.2.5 Inpainting: visual results

Figure 6.4.6 shows how much better the template-based inpainting is better. Restored images are much less blurred in comparison to the ones obtained by median-based restoration. Also, template-matching is capable, in a limited way, of synthesizing both the texture and the structure. This is clearly visible in figures 6.4.6c and 6.4.6d : Foreman's mouth is much more nicely rendered. Also, the straight edges from the background are better restored by the template-based inpainting.

(a) *Original image*

(b) *Initially synthesized image*
SPSNR = 33.55 dB, Rec. rate = 80.31%

(c) *Median inpainted image, N = 3*
SPSNR = 26.56 dB, Rec. rate = 100%

(d) *Hybrid template/median restoration, N = 3*
SPSNR = 28.46 dB, Rec. rate = 100%

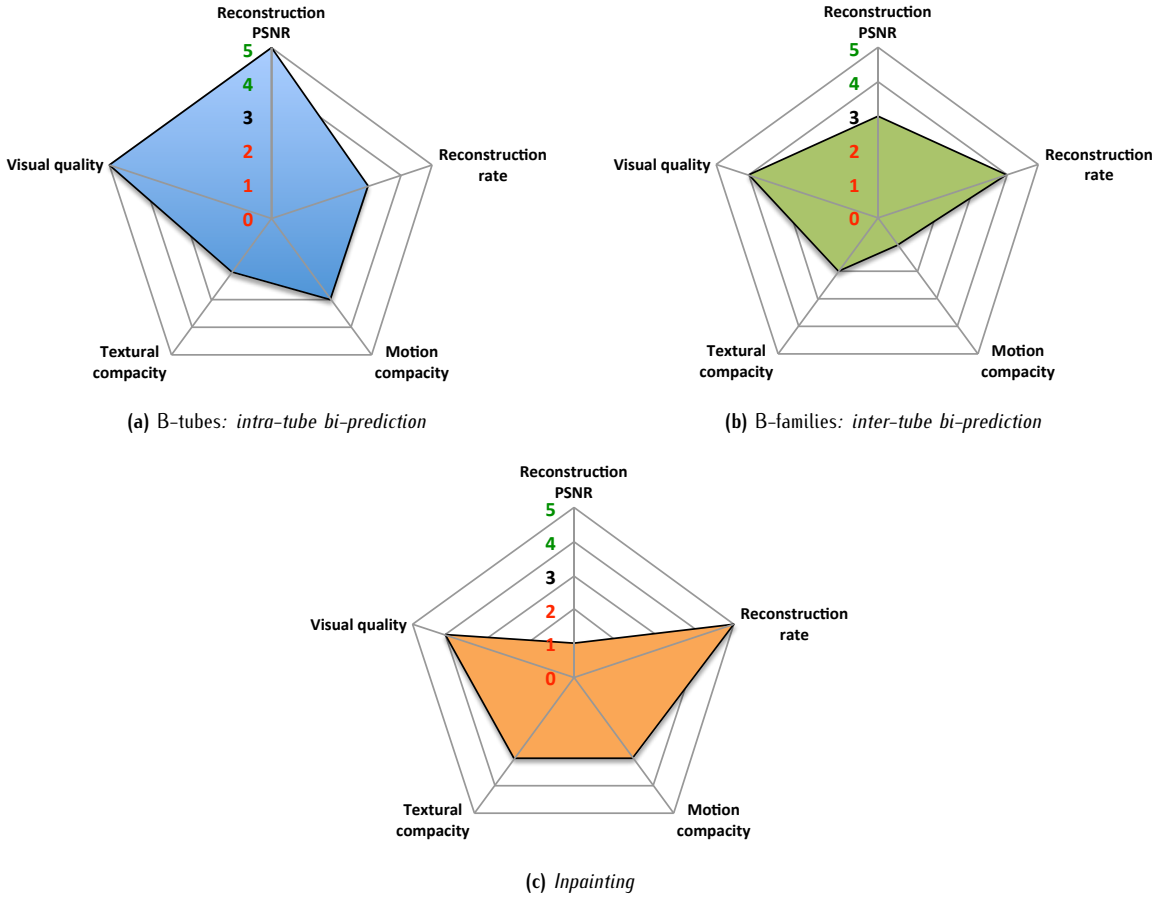
Figure 6.4.6: *Median-based inpainting mechanism: a crude way to fill unpredicted areas*

6.5 Conclusion

This chapter focused on the textural information hold by the *motion tubes* and provided a set of mechanism which enhance the quality of the synthesized textures, both in terms of objective/subjective quality and reconstruction rate. Targeted problems were the inability of the *motion tubes* to handle texture changes, and the lack of completeness of the reconstructed images. Three main mechanisms were provided:

1. **B-tubes** can source their textural information from several reference frames, thus crudely handling textural changes;
2. **B-families** of *motion tubes* were introduced to register a larger amount of textural information to be tracked across each GOP, hence drastically increasing the reconstruction rate;
3. spatial and spatio-temporal **inpainting** mechanisms were provided to fill remaining holes.

All of these mechanisms abilities are summarized in figure 6.5.1, in regards to five criteria: objective and subjective synthesis quality, motion and texture bitrate, and reconstruction rate. Score 3 means that the corresponding criterion is not affected; scores greater than 3 correspond to improving abilities, and scores lesser than 3 correspond to worsening effects. For the time being, both B-tubes and inpainting mechanisms are clearly advised. B-families, however, should not be used until the transmission of unused *motion tubes* can be avoided.



(a) B-tubes*: intra-tube bi-prediction*

(b) B-families*: inter-tube bi-prediction*

(c) *Inpainting*

Figure 6.5.1: *Chapter 6: a summary of the abilities of the provided texture improvement mechanisms*

Two problems still need to be raised. With *B-tubes* and *B-families* of *motion tubes*, one made the assumption that additional textural information would be available at the decoder side. This information overhead was neither dealt with nor taken into account in provided results; next chapter will propose a fully functional coder which actually handles this extra textural information. In addition, *B-families* of *motion tubes* critically increase the number of *motion tubes* to be transmitted, hence the allocated motion bitrate. However, a large part of these tubes are redundant in regards to the representation and may not even be used at all: a tube selection mechanism needs to be designed in order to remove redundant or inefficient ones. Next chapter will further investigate this problem.

Chapter 7

A compression-driven mechanism for the life and death of the motion tubes

Conceptualizing the notion of *motion tube* led to the construction of a threefold spatio-temporal information. While previous chapters investigated the deformation model of the *motion tubes* and the construction of their textural information, they did not provide any guidance regarding the way *motion tubes* should be started and terminated. As a consequence, it was assumed that *motion tubes* were lasting throughout the whole GOP (except B-families of high temporal levels which were only defined over a part of the GOP). In other words, this assumption implies that any patch of texture sourced from the reference instant can actually be matched in any other image from the GOP. Obviously, this hypothesis generally does not stand, as occlusions and disocclusions frequently happen in most image sequences.

As a first step towards a content-adaptive instantiation mechanism for the *motion tubes*, it was proposed in section 6.3 to build each GOP from several temporally overlapping families of *motion tubes*. From the additional *motion tubes*, the amount of unregistered textural information was drastically reduced, to the cost of a large increase in motion bit-rate. Be that as it may, the provided *motion tubes* instantiation mechanisms did not take into account two critical phenomenons:

- **occlusions** have a critical impact on the temporal consistency of the spatio-temporal information: *motion tubes* should be aware of the appearance and the disappearance of the patch of texture being tracked;
- from a **compression** perspective, the number of *motion tubes* should be kept as low as possible in order to transmit as few motion information as possible, provided that synthesized textures are accurate enough.

For both these reasons, it is critical for each of the *motion tubes* to be assessed as for their respective abilities to represent a spatio-temporal area of a GOP. From then, a decision mechanism may control their life and death, only keeping necessary *motion tubes*. Ideally, such a mechanism ought to take into account both *intra-tube* and *inter-tube* characteristics:

- **intra-tube** characteristics reflect the intrinsic abilities of a *motion tube* to describe a spatio-temporal area;
- **inter-tube** characteristics reflect the relative efficiency of a *motion tube*, in regard to its neighbours which may also synthesize, even partially, the same spatio-temporal area, thus introducing redundancies.

As a consequence, processing the *motion tubes* temporal information raises two major problematics: how *motion tubes* should be evaluated, and how provided metrics should be used to control whether to start, pause or terminate a *motion tube*. This chapter focuses on both these questions. Section 7.1 will first summarize the different problems an idealistic life and death mechanism should handle. As a first step towards an assessment metric, section 7.2 will then explain how *motion tubes* have been integrated within H.264/AVC coding scheme, and how its decision mechanism provides invaluable information regarding the *motion tubes* coding efficiency. Section 7.3 will further investigate H.264/AVC's decisions regarding the *motion tubes*, and provide a coder-dependent approach to the problem of their life and death. Later, section 7.4 will introduce an hybrid quality-stability metric for the *motion tubes*, which relies on H.264/AVC's decisions and also takes into account the temporal dimension of the *motion tubes*. From this metric, a first selection mechanism will be provided. Finally, section 7.5 will conclude the chapter.

7.1 An idealistic life and death mechanism for the motion tubes

Whichever way, a selection mechanism mostly relies on an assessment metric which evaluates the ability of a *motion tube* or a group of *motion tubes* to properly represent a spatio-temporal area of an image sequence in a compact manner. Prior to the establishment of an appropriate assessment metric, it is essential to define the objectives of a life and death mechanism; in particular, which phenomenons it should be able to detect and handle. Consequently, this first section will review different scenarios which should be avoided within the representation, and then introduce a high-level description of our understanding to an ideal selection mechanism for the *motion tubes*.

The main objective of a selection mechanism is to rule the life and death of each *motion tube*. It is assumed that a sufficient number of them are instantiated and tracked across the considered GOP: the decision mechanism does not aim at deciding when and where a *motion tube* should be started. Indeed, this problem has already been investigated through the use of families and *B-families* of *motion tubes*. On the other hand, when and where *motion tubes* should be terminated is another problem of critical importance, and will thus be addressed by the provided life and death mechanism.

7.1.1 Dealing with occlusions and disocclusions: adaptation of the lifespan

As it was earlier mentioned in the introduction of the current chapter, the proposed spatio-temporal description provided by the *motion tubes* may suffer from several issues. Besides deformation mismatches and incompleteness of the reconstructions (both these issues were partially addressed in chapter 6), the provided scheme is, for now, unable to detect the disappearance or the occlusion of a patch of texture being tracked. In such case, the corresponding *motion tube* should be stopped or at least, temporary disabled until the patch of texture reappears. According to the availability of a patch of texture throughout a GOP, four main scenarios can be foreseen:

1. the patch of texture is **fully available** throughout the whole GOP and its **deformation** is **simple enough** so that the corresponding *motion tube* manages to keep track of its evolutions across time;
2. the patch of texture is **unavailable** (except at the reference instant) and/or **cannot be tracked** due to a deformation much too complex;
3. the patch of texture is **available** in several successive images, but **disappears** at some point (occlusion);
4. the patch of texture is **unavailable** for some time, then **re-appears** at some point (disocclusion).

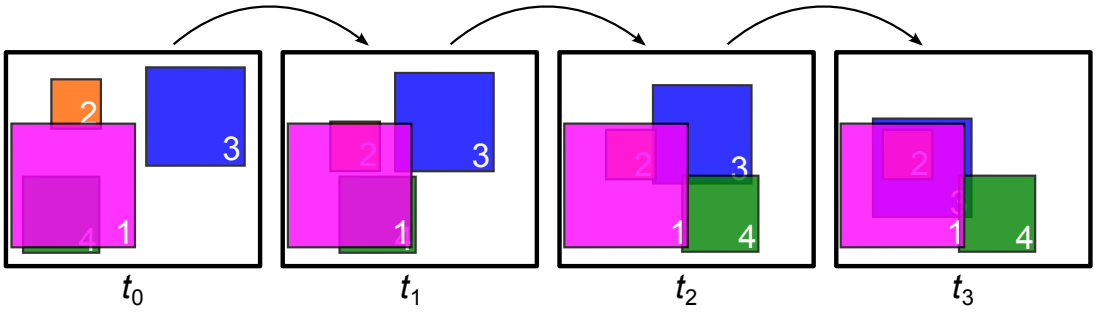


Figure 7.1.1: *Appearance and disappearance of several patches of textures*

Each of these scenarios is illustrated in figure 7.1.1 by a moving patch of texture captioned with the corresponding scenario index. The first two scenarios are the most trivial situations: in the first one, the patch of texture is easily registered by a *motion tube* across the whole GOP; in the second scenario, on the contrary, it should be avoided to instantiate a *motion tube* as it will not be able to provide any faithful spatio-temporal information. Third and fourth scenarios are much more problematic, as they require the *motion tubes* to detect occlusions and disocclusions.

Until now, *motion tubes* were processed regardless to such considerations; whenever a patch of texture was disappearing, the corresponding *motion tube* was continued and matched with an erroneous patch of texture. As a consequence, resulting spatio-temporal representation was biased and may exhibit some undesirable artifact: jerky displacements and deformations whose associated motion bit-rate may penalize the overall compression efficiency.

7.1.2 Limiting the amount of spatio-temporal redundancies

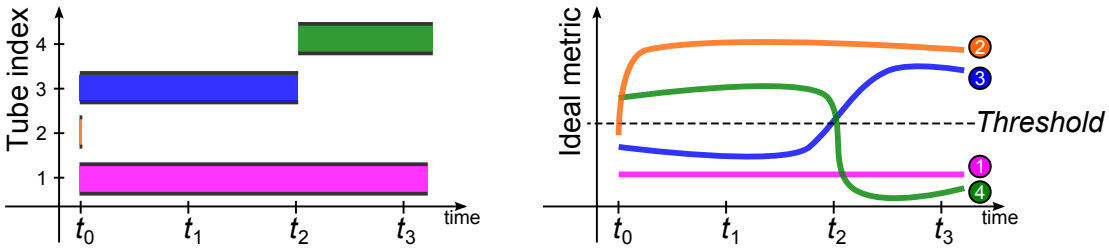
No matter how good or bad is the intrinsic quality or efficiency of a *motion tube*, it needs to be evaluated in regards to its neighbours. Indeed, it should be very likely for several neighbouring *motion tubes* to overlap, resulting in a consequent amount of multi-registered textures:

- from a **texture synthesis** perspective, while several contributions may increase the quality of the synthesized textures, too many of them may result in blurring artifact and lower objective and/or subjective synthesis quality;
- from a **compression** perspective, an increasing number of contributions will introduce a consequent amount of redundancy, which should be avoided at all costs.

For both these reasons, the amount of multi-registered textures needs to be controlled. Whenever two *motion tubes* (or more) are describing approximately the same spatio-temporal area, the final representation should keep as few of them as possible, providing the best tradeoff between amount of redundancies and quality of synthesis.

7.1.3 Towards an ideal life and death mechanism for the motion tubes

From previous considerations, it is obvious that the lifespan of the *motion tubes* should be limited to the time intervals at which the tracked patch of texture is available and can be tracked by the provided motion model. For instance, the *motion tube* tracking patch 3 (see figure 7.1.1) should be terminated at time instant t_2. As for the *motion tube* tracking patch 2, it should not be instantiated at all. Figure 7.1.2a shows the optimal lifespan of four *motion tubes* tracking the different patches of textures introduced in figure 7.1.1.



(a) *Optimal lifespans* **(b)** *An ideal assessment metric*

Figure 7.1.2: *An ideal decision mechanism: application to patches of textures from figure 7.1.1*

As a consequence, an ideal life and death mechanism should be able to detect when and where such scenarios occur. From then, it should settle the lifespan of affected *motion tubes* accordingly. As *motion tubes* provide an accurate spatio-temporal information, the occurrence of problematic scenarios is likely to be reflected on their behaviour: jerky displacements, incoherent deformations, sudden increase in matching error and/or motion coding cost, etc. All in all, any rupture in the temporal consistency of a *motion tube* should motivate its deactivation, reactivation or termination. Anyhow, regardless to the intrinsic quality of a *motion tube*, its existence should be conditioned by the set of overlapping *motion tubes* which contribute to the synthesis of the same spatio-temporal area. An ideal life and death mechanism may then be organized into two steps:

1. each *motion tube* is **independently assessed** by an appropriate metric in charge of detecting spatio-temporal inconsistencies, which may motivate its deactivation, reactivation or termination, from its parameters and synthesized texture. In figure 7.1.2b, for instance, *motion tubes* are kept only when the metric is inferior to a given threshold;
2. **redundant** *motion tubes* are identified and **removed**.

7.2 Assessing the motion tubes through state of the art H.264/AVC compression scheme

As can be seen from section 7.1, designing an evaluation metric for the *motion tube* is greatly dependent on the desired life and death mechanism. Rather than blindly assessing the *motion tubes* regardless to any encoding reference, it is

proposed as a first step to evaluate their quality and efficiency in regard to state of the art compression techniques provided by ITU-T H.264/AVC video compression standard. By making *motion tubes* competing with both *intra* and *inter* H.264/AVC coding modes, the coder rate-distortion decision mechanism will then be able to identify which areas of the images can effectively be synthesized by the *motion tubes*.

In H.264/AVC compression scheme, the motion field is re-initialized for each image, such that both textural and motion parameters can be optimized one macroblock at a time. With *motion tubes*, however, the provided representation exhibits a set of trajectories along successive GOPs. Consequently, they should not be optimized time instant by time instant, and an appropriate decision mechanism may consider the whole spatio-temporal area being synthesized.

Though H.264/AVC compression scheme does not provide an appropriate decision mechanism, it may still provide valuable information regarding the synthesis abilities of the *motion tubes*. Section 7.2.1 will present a crude way to integrate *motion tubes* into H.264/AVC. Then, section 7.2.2 will summarize various types of information indicating how *motion tubes* are eventually used by H.264/AVC.

7.2.1 Embedding the motion tubes into H.264/AVC coding scheme

In order to compare H.264/AVC coding modes and *motion tubes*, it is necessary to integrate the *motion tube* prediction mechanism within H.264/AVC coding loop. In order to keep the *motion tube* prediction mechanism as independent as possible from H.264/AVC structure, it is naturally proposed to provide an additional set of reference images which hold the *motion tube* prediction. As H.264/AVC builds images from a set of list of reference images and a set of coding tools, an existing coding mode is then modified to signal whether to use or not to use the additional reference list.

7.2.1.1 Modifying skip and direct coding modes to signal the use of the motion tubes

Naturally, integrating the *motion tube* mechanism into H.264/AVC coding scheme should require the addition of an extra coding mode dedicated to the *motion tubes*. However, this work focuses on the *motion tubes*, and not their full integration within H.264/AVC coding scheme. Instead of adding an extra coding mode, *motion tubes* are provided to the coder through already existing modes *skip* and *direct*. An additional binary flag is sent to signal whether to use or not to use the *motion tubes*. A more appropriate integration may later consist in an interesting perspective, provided that *motion tubes* effectively bring improved compression abilities.

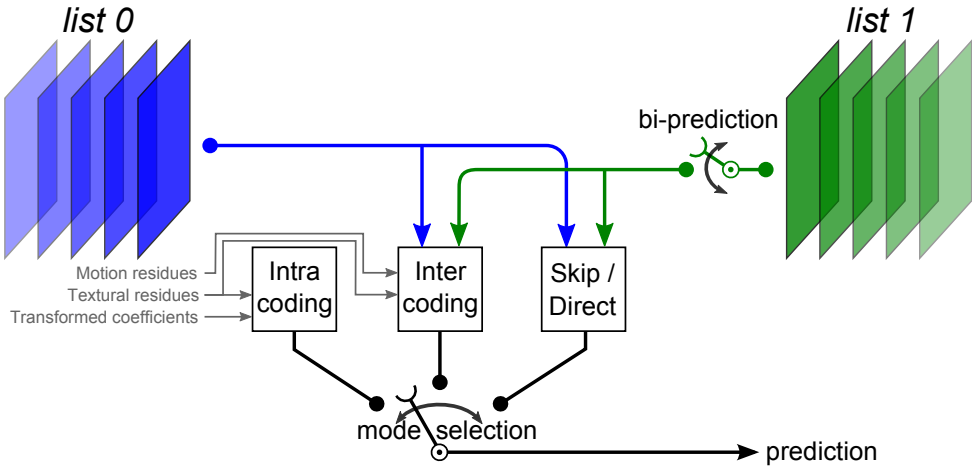


Figure 7.2.1: *Original H.264/AVC coding scheme and associated reference lists*

As can be seen from figure 7.2.1, both of these modes do not require any side-information to encode and decode the current *macroblock*. They simply interpolate the motion vectors from neighbouring *macroblocks*, and source the collocated *macroblocks* from two lists of reference images, *list 0* and *list 1*. Consequently, they can be easily modified to source the textural information from a third set of reference images synthesized by the *motion tubes*. In such case, interpolated motion vectors are set to zero as provided images are already motion compensated by the *motion tubes*.

Another consequence of the absence of side information needed to be transmitted in *skip* and *direct* modes is that required bit-rate simply consists in the cost of the binary flag signalling the use of these modes. Ideally, resulting Lagrangian should be modified to account for the cost of the *motion tubes*. However, the coding cost of a *motion tube* at a single time

instant has little if no meaning at all, and should be evaluated across its whole lifespan. In addition, the overlapping grid of the *motion tubes* and the *macroblock* grid do not match, such that any *macroblock* can be predicted from parts of a variable number of *motion tubes*: computing the associated coding cost is problematic. Consequently, no additional cost will be associated to the modified *skip* and *direct* coding modes. The overall bit-rate will then be given by the sum of the resulting H.264/AVC bit-rate and the motion bit-rate of the *motion tubes*.

7.2.1.2 A third reference list to provide the motion tube spatio-temporal prediction

According to the chosen coding mode, H.264/AVC may use up to two reference images to encode the *macroblocks*. In order to manage the different images from which textures can be sourced to encode each *macroblock*, H.264/AVC uses two lists of reference images, namely *list 0* and *list 1* (see figure 7.2.1). *Inter* coding modes predict the current *macroblock* from (at most) two collocated blocks; the first one is sourced from an image of *list 0*, while the second is sourced from an image of *list 1* (when bi-prediction is used). Reference lists *list 0* and *list 1* are efficiently managed such that they only contain necessary images: as soon as a reference image is not necessary anymore, it is removed from the list.

Motion tubes, however, require all the images to be available in order to estimate their deformation and to synthesize the reconstructed images. Modifying the state machine which rules the contents of the reference lists, again, may consist in a complex task. A simpler solution, from an implementation perspective, is to instantiate a third reference list, *list 2*, within which images reconstructed by the *motion tubes* are stored. As *list 2* is independent from the other lists, it does not obey to neither MMO commands nor implicit behavior imposed by the GOP structure.

In consequence, *skip* and *direct* modes (see section 3.3.1.2) are further modified to use *list 2* as an alternative reference list. An additional binary flag signals whether to use the original reference lists (*list 0* and *list 1*) or the third reference list (*list 2*). Classic *intra* and *inter* modes are now competing with the *motion tube* prediction. This is illustrated in figure 7.2.2.

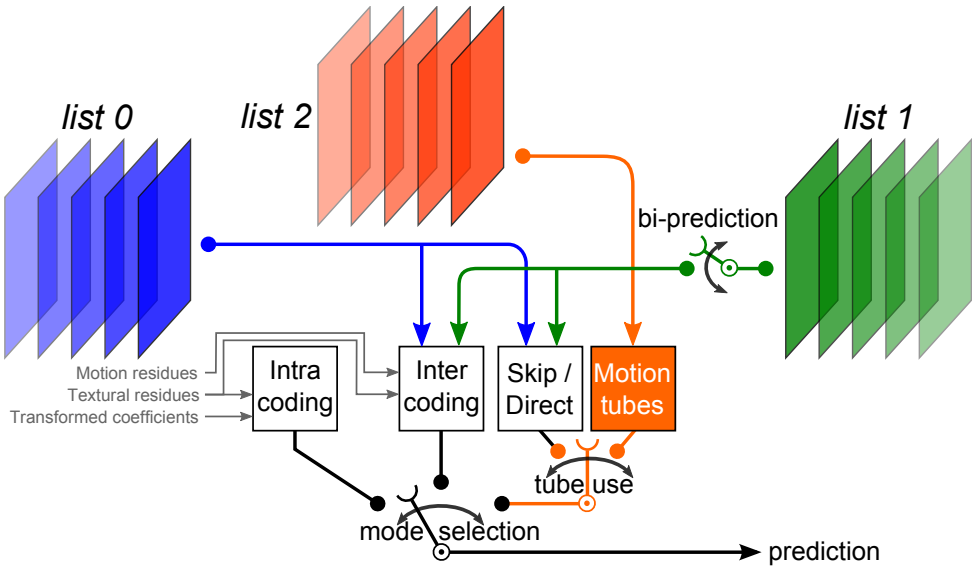


Figure 7.2.2: *Modified H.264/AVC coding scheme and additional reference list*

7.2.1.3 Motion estimation process

Estimating and compensating the deformation of the *motion tubes* requires original and encoded images. According to the GOP structure, the availability of these images may greatly vary, thus limiting the way *motion tubes* can be processed. Again, one may modify the way the Joint Scalable Video Model (JSVM) handles these images; yet, it is simpler to process the *motion tubes* according to the availability of the original and encoded images.

a Original or encoded images as input data to the motion estimation

Motion estimation can be performed from two different couples of images:

1. **between I_{ref} and I_{cur}:** the motion parameters are computed from two original images, then applied to a coded/decoded image $\overline{I_{ref}}$ to generate the motion compensated prediction;
2. **between $\overline{I_{ref}}$ and I_{cur}:** the motion parameters are computed from the original current image and the coded/decoded reference image $\overline{I_{ref}}$. Eventual impairments between original and decoded images may lower the estimation quality.

Whenever encoded images are too much distorted from the encoding steps, strong divergences can be observed in the motion field whether original or encoded images are used to perform the estimation. In H.264/AVC's closed-loop structure, motion estimation is typically performs following the second scenario. While H.264/AVC's closed-loop structure typically performs the estimation between $\overline{I_{ref}}$ and I_{cur}, one may wonder whether such a scheme still holds with *motion tubes*.

b IPP and hierarchical B GOP structures

In H.264/AVC's reference implementation, the JSVM, original images are only partially available during the coding process, depending on the GOP structure. In predictive structures $\mathtt{I}_0\mathtt{P}_1\mathtt{P}_2\mathtt{P}_3\mathtt{P}_4\ldots$, only the previous image is. In hierarchical B structures $\mathtt{I}_0\mathtt{b}_4\mathtt{B}_3\mathtt{b}_5\mathtt{B}_2\mathtt{b}_7\mathtt{B}_6\mathtt{b}_8\mathtt{P}_1\ldots$, not only reference images from the previous temporal level are available, but also previous images which have not been coded yet. Generally, hierarchical GOP structures provide better compression performances than progressive structures do.

At the last image of a hierarchical GOP, all its original images are available as they have been loaded into memory by the JSVM, and yet not coded. This is an ideal place for the *motion tubes* to be processed. *Motion tubes* can also be temporally processed either in a progressive or in a hierarchical fashion. When their motion estimation is exclusively performed on the original images, both temporal structures can be handled. However, when their motion estimation is computed from original and coded images, their temporal structure needs to be synchronized with H.264/AVC's GOP structure. Both scenarios were evaluated, clearly showing that using the original images is best. Indeed, the distortion of the encoded images, especially at low bitrates, prevents the *motion tubes* from accurately seizing the spatio-temporal coherence on the long run.

7.2.2 Measuring the motion tubes impact on H.264/AVC coding scheme: preliminary results

Motion tubes are now competing with H.264/AVC's coding modes. This may answer numerous questions on the synthesis and the compression abilities of the *motion tubes*. When and where are *motion tubes* selected? Is their information of any help to the coder? If so, does the gain in H.264/AVC bit-rate compensates the *motion tube* coding cost?

In order to validate the competition between *motion tubes* and H.264/AVC classic coding modes, a set of CIF sequences (*Bus*, *Football*, *Foreman*, *Mobile*, *Paris* and *Tempest*) have been encoded with and without the *motion tube* mode. These encoding operations led to a large amount of information of various nature (bit-rates, PSNR, coding mode statistics, etc). Their interpretation will help us assess the abilities of the *motion tubes*.

7.2.2.1 A first glance at rate-distortion curves: impact of the *motion tubes* on the overall bit-rate

a Setting up the experiments: a set of default parameters for the motion tubes

Parameter	Value	Effect on the representation
Motion mode	TMC	Translational motions only
Tube size	$32\times32 \longrightarrow 4\times4$	Variable-size *motion tubes*
Motion tube GOP structure	$\mathtt{I}_0\mathtt{b}_4\mathtt{B}_3\mathtt{b}_5\mathtt{B}_2\mathtt{b}_7\mathtt{B}_6\mathtt{b}_8\mathtt{P}_1\ldots$	Hierarchical GOP structure
Motion estimation	Causal and anticausal	Two steps to improve the estimation
Motion precision	1/4 pixel	Quarter-pixellic accurate motion estimation
Motion QP	$QP_{tube} = QP_{AVC}$	AVC's QP is used to set the tube motion QP
Matching criterion	SAD	Default H.264/AVC matching criterion
Template matching inpainting	✓	Best inpainting abilities
Median inpainting	✓(iterated)	Images completely reconstructed
B-tubes	×	No textural multi-prediction
B-families	×	A single family of *motion tubes* per GOP

Table 7.2.1: *A set of default parameters for the* motion tubes

As could be seen in previous chapters, *motion tubes* are driven by a large set of mechanisms, hence numerous parameters. As a consequence, synthesized images greatly vary according to the provided configuration. In order to perform forthcoming experiments, it has been decided to configure the *motion tubes* from a set of parameters which guarantee a proper synthesis quality. Inpainting, for instance, should absolutely be set such that synthesized images are complete. As for the GOP structure, both H.264/AVC and *motion tubes* are set up to use dyadic hierarchical structures of nine images. Table 7.2.1 summarizes the main parameters of the *motion tubes* along with their default values.

b Resulting rate-distortion curves: mitigated performances

Figure 7.2.3 shows the average gains in H.264 bitrate when *motion tubes* are enabled. It appears that *motion tubes* are effectively chosen by H.264/AVC, which further valids the proposed representation. However, it appears that the gain in H.264 bitrate (+4.2% in average) cannot compensate the motion bitrate of the *motion tubes*, by far, since the final lost in bitrate is about 28%. Still, as all *motion tubes* are transmitted, this gap may be significantly reduced if unused *motion tubes* were discarded prior to their transmission. Be that as it may, the gain in H.264 bitrate does prove that *motion tubes* provide, in some areas, a valid representation of the sequence. Gains are highly correlated to the figures from previous chapters; sequence *Football*, in particular, is badly handled by the *motion tubes*.

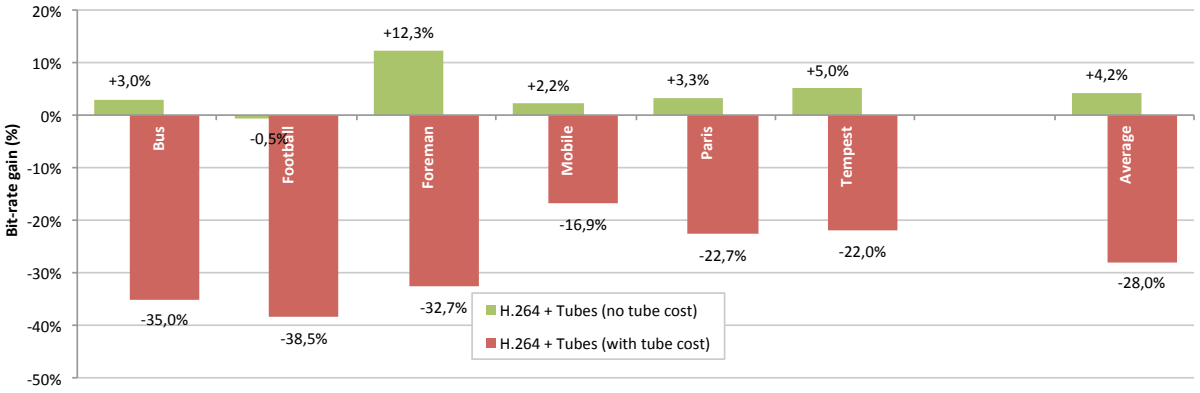
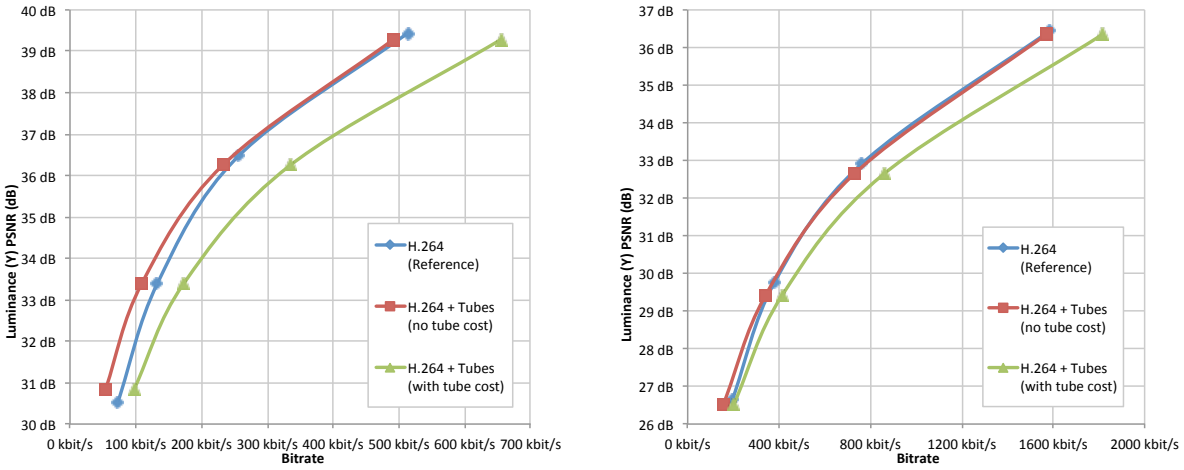


Figure 7.2.3: *Gains in H.264 bitrate obtained by embedding* motion tubes *to the coder*

Figure 7.2.4 shows the rate-distortion profiles of sequences *Foreman* and *Mobile*. It appears that *motion tubes* are much more efficient for low bitrates. This is easily explained by the fact that *motion tubes* are solely embedded within modes *skip* and *direct*: there is no way to correct their prediction from textural residues as remaining *inter-coding* modes do. At high bitrates, the motion prediction is generally not accurate enough, and H.264/AVC favours classical *inter* modes.



(a) *Sequence* Foreman

(b) *Sequence* Mobile

Figure 7.2.4: *Rate-distortion curves: a comparison between the original and the modified H.264 coder*

7.2.2.2 A focus on motion tubes textural improvement mechanisms and their effect on resulting bit-rates

In chapter 6, several multi-prediction mechanisms were provided to improve the synthesized images. *B-tubes* and *B-families* of *motion tubes* proved to be quite good improvements. However, they required some additional textural information to be transmitted. Now that *motion tubes* are integrated within H.264/AVC, this textural information is available: it can be sourced from the encoded images. This section compares the rate-distortion performances of the *motion tubes* within H.264/AVC coding scheme, when *B-tubes* or *B-families* of *motion tubes* are enabled.

a A significant bitrate saving if the cost of the motion tubes is not taken into account

Figure 7.2.5 shows how *B-tubes*, and especially *B-families* of *motion tubes* impact the coder's decisions: the average gain in bitrate is increased from 4.2% to 6.6% with *B-tubes*, and up to 15% with *B-families* of *motion tubes*.

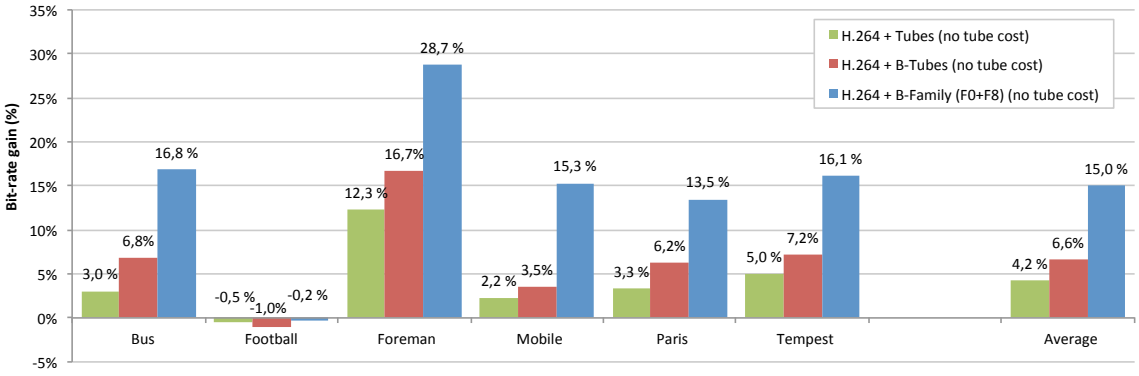
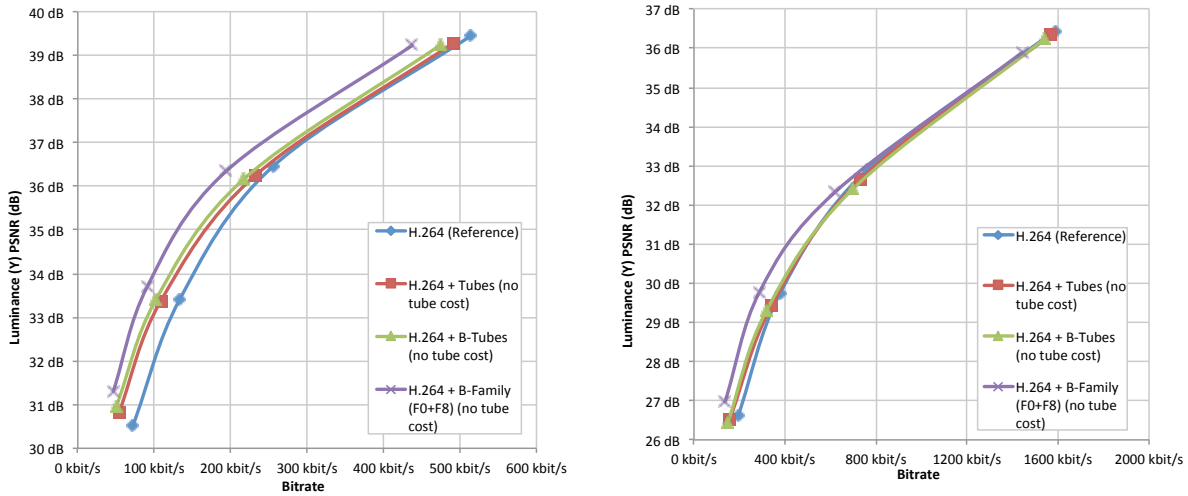


Figure 7.2.5: B-tubes *and* B-families*: gains in H.264 bitrate (without motion tubes cost)*

Figure 7.2.6 shows corresponding rate-distortion curves for sequences *Foreman* and *Mobile*. Again, low bitrates are much more in favour of *motion tubes*. As could be expected, using several families of *motion tubes* clearly improves the representation, as it nearly quadruple the bitrate saving percentage, and provides an improvement for all bitrates.



(a) *Sequence* Foreman **(b)** *Sequence* Mobile

Figure 7.2.6: *Rate-distortion curves: a comparison between the original and the modified H.264 coder*

b Incorporating the motion cost of the motion tubes into the figures

If we now take the motion cost of the *motion tubes* into account, there is no bitrate saving at all, but quite the contrary. Still, *B-tubes* slightly reduce the lost in bitrate by 2% in comparison to standard *motion tubes*. On the other hand, the

instantiation of two families of *motion tubes* through *B-families* roughly multiplies by two the tubes coding cost. Figure 7.2.7 shows how the overal bitrate is affected by *B-tubes* and *B-families* of *motion tubes*.

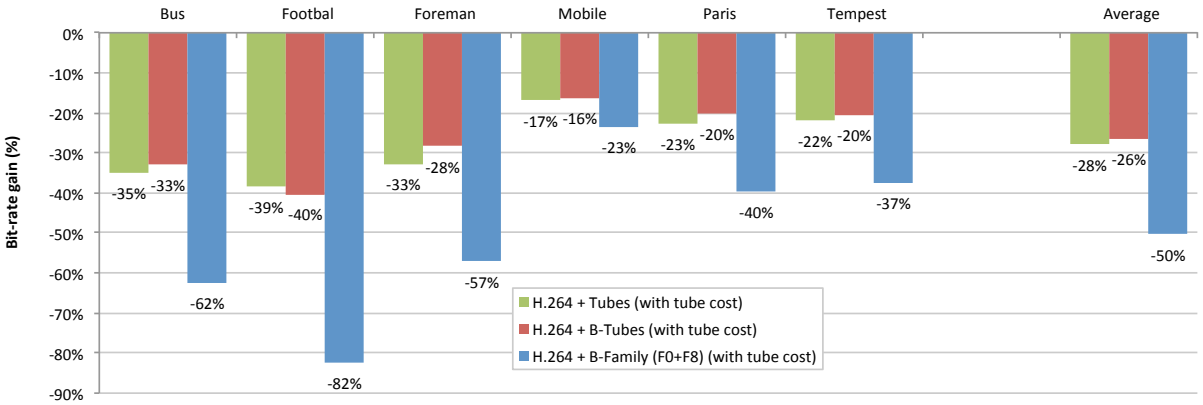
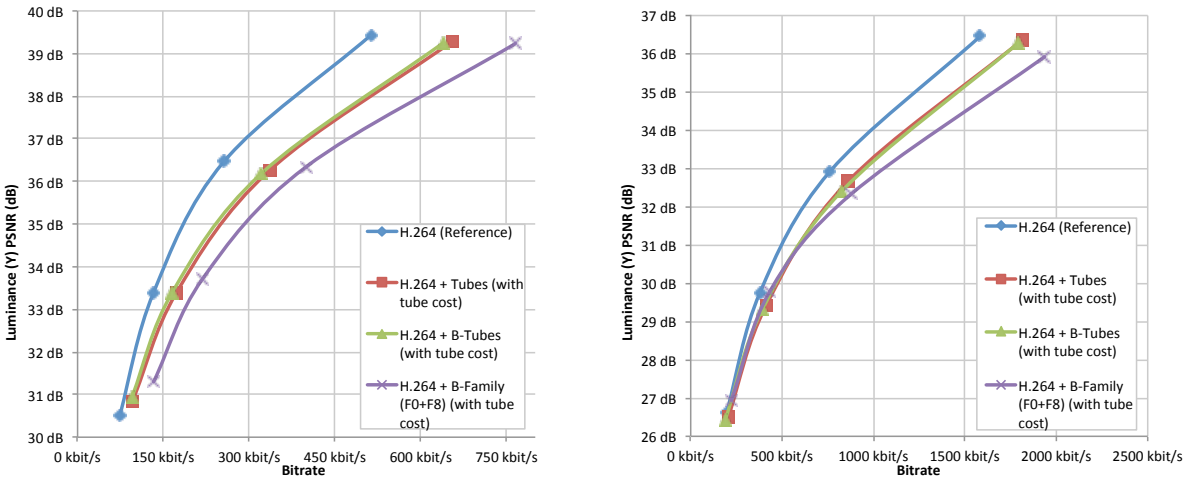


Figure 7.2.7: B-tubes *and* B-families*: gains in H.264 bitrate (with motion tubes cost)*

As a consequence, the gap between H.264 bitrate savings and the motion cost of the tubes is significantly increased, which ends up in very poor compression performances. However, by removing the cost of unused *motion tubes*, these dramatic figures may be significantly improved, even though whether to use or not to use a *motion tube* will need to be signalled as well. Figure 7.2.8 shows corresponding rate-distortion curves for sequences *Foreman* and *Mobile*.



(a) *Sequence* Foreman **(b)** *Sequence* Mobile

Figure 7.2.8: *Rate-distortion curves: a comparison between the original and the modified H.264 coder*

7.3 An insightful investigation on the impact of the motion tubes on H.264/AVC's decisions

Previous section only provided a little insight on the impact of the *motion tubes* on the initial H.264/AVC coding scheme, and showed encouraging results as for their interest in regard to compression. However, these preliminary results did not provide any accurate information regarding the way *motion tubes* were used by the coder. From the experiments, a lot of questions emerge.

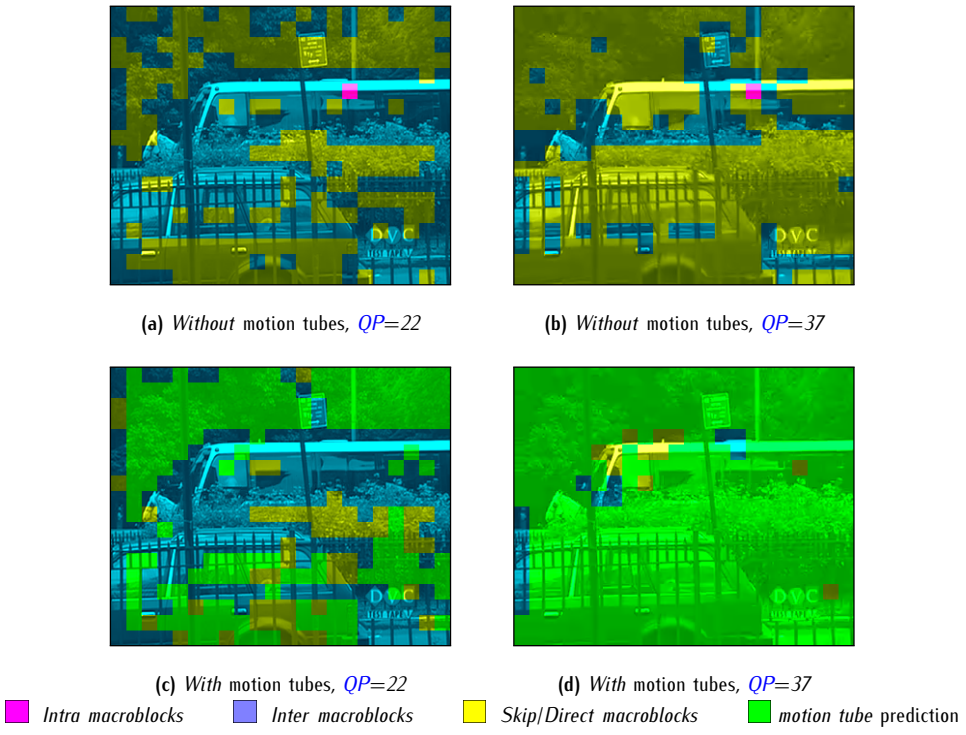
- When and where exactly *motion tubes* are selected by the coder?
- Which initial coding modes are advantageously replaced by the *motion tubes*? Which are not? To which extent?

- From a subjective perspective, from which artifact do encoded sequences suffer?
- Which influence has the QP on the decisions?

This section will first investigate the decisions made by the modified H.264/AVC compression scheme in a top-down fashion, first focusing on overall figures, then laying on more specific information describing in details how the *motion tubes* are used by H.264/AVC compression scheme. These invaluable information will shed light on the construction spatio-temporal decision mechanism in charge of the life and death of *motion tubes*.

7.3.1 A glimpse into the motion tube selection patterns

Before providing any statistics regarding the use of the *motion tubes* and their impact on other coding modes, let us look at typical decision examples. Figure 7.3.1 shows several maps describing the use of *intra*, *inter*, *skip* and *tube* coding modes on the second image I_1 of sequence *Bus*. As *motion tubes* are initialized at previous instant from image I_0, it is expected for the *motion tubes* to provide a prediction of high quality. As a consequence, maps should indicate that a large amount of *macroblocks* is predicted from the *motion tubes*.



(a) *Without* motion tubes, *QP=22* **(b)** *Without* motion tubes, *QP=37*

(c) *With* motion tubes, *QP=22* **(d)** *With* motion tubes, *QP=37*

Figure 7.3.1: *Influence of the* motion tubes *on H.264/AVC's decisions, sequence* bus, I_1

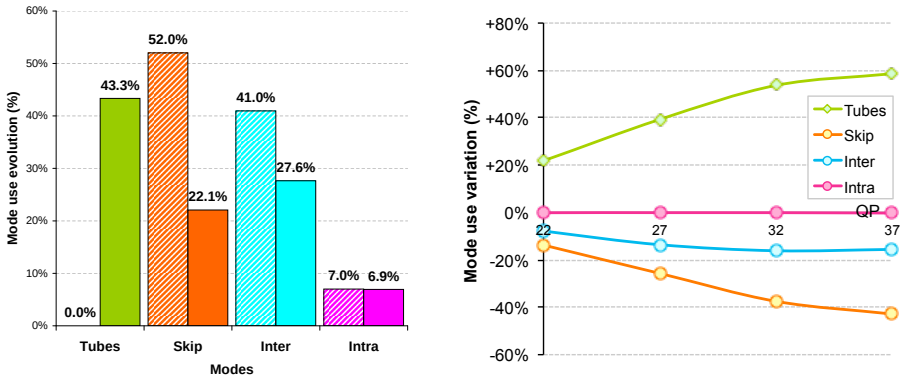
Coding mode maps are shown for both QPs 22 and 37. For the record, a QP of 22 is a widely accepted H.264/AVC quantization parameter for sequences which need to be encoded at a relatively high quality. On the contrary, a QP of 37 generally outputs highly compressed video bitstreams.

What immediately strikes us is how much the number of *macroblocks* predicted from the *motion tubes* is dependent on the QP value. At high bit-rates, the *motion tube* prediction is often relinquished to the benefit of more accurate *inter* coding mechanisms. On the contrary, the *motion tube* prediction is largely selected at low bit-rates. This can be easily explained from the fact that *inter* predictions are further improved from transmitted residual coefficients. At high bit-rates, these advanced inter coding modes are favoured by H.264/AVC's Rate-Distortion Optimization (RDO). Looking closely to the *macroblocks* coding mode maps, it can be seen that the largest part of the *macroblocks* predicted from the *motion tubes* correspond to *macroblocks* initially coded with *skip* or *direct* modes. However, some of the initial *inter macroblocks* are also replaced by the *motion tube* prediction, which is particularly encouraging.

7.3.2 An accurate investigation on the selection of the motion tubes

7.3.2.1 Overall coding modes selection rates

By providing the *motion tubes* prediction to H.264/AVC compression scheme, it was seen in previous section that the decisions taken by the coder were largely impacted. Let us now observe, for each available coding mode, the percentage of corresponding *macroblocks*. Figure 7.3.2a shows the evolution of the selection rate of each coding mode. For each mode, the left striped bar corresponds to the initial selection rate obtained with the reference H.264/AVC encoder, while the right solid bar corresponds to the resulting selection rate when *motion tubes* are embedded into the coder. To obtain these figures, selection rates from all sequences from the test set have been averaged.



(a) *Average over QPs 22, 27, 32 and 37* **(b)** *Evolution of the modes selection rates w.r.t. the QP*

Figure 7.3.2: *Influence of the* motion tubes *on the selection rates of the different coding modes*

Figure 7.3.2b confirms the observation made previously: the selection rate of the *motion tubes* is highly correlated to the QP. While only 20% of the *macroblocks* are encoded via the *motion tubes* at high bit-rates (QP=22), this selection rates rises up to 60% for low bit-rates (QP=37). Again, it can be seen (unsurprisingly) that *motion tubes* have little impact on the selection rate of *intra* modes. Indeed, *intra macroblocks* generally correspond to areas wherein motion estimation has failed, thus requiring to be spatially encoded. Besides the *intra* mode, all selection rates are linearly increasing or decreasing from QP 22 to QP 32, then become relatively stable for higher bitrates. However, the selection rate of *inter* modes seems to be less slightly correlated to the QP than the *skip* mode.

Unlike *intra* mode, *inter* and especially *skip*/*direct* modes are heavily impacted by the *motion tubes*. The amount of *macroblocks* encoded by any of the available *inter* modes (which require both motion and textural residues to be transmitted) is decreased by 25% (w.r.t. to the initial selection rate) which is significant enough to be noticed. As for *skip*/*direct* modes, their selection rate is drastically reduced, from approximately 50% down to 20%.

7.3.2.2 How the evolution of the modes selection rates is correlated to the QP

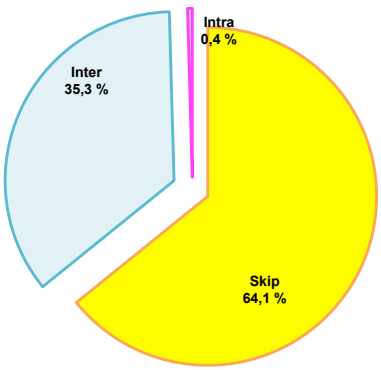
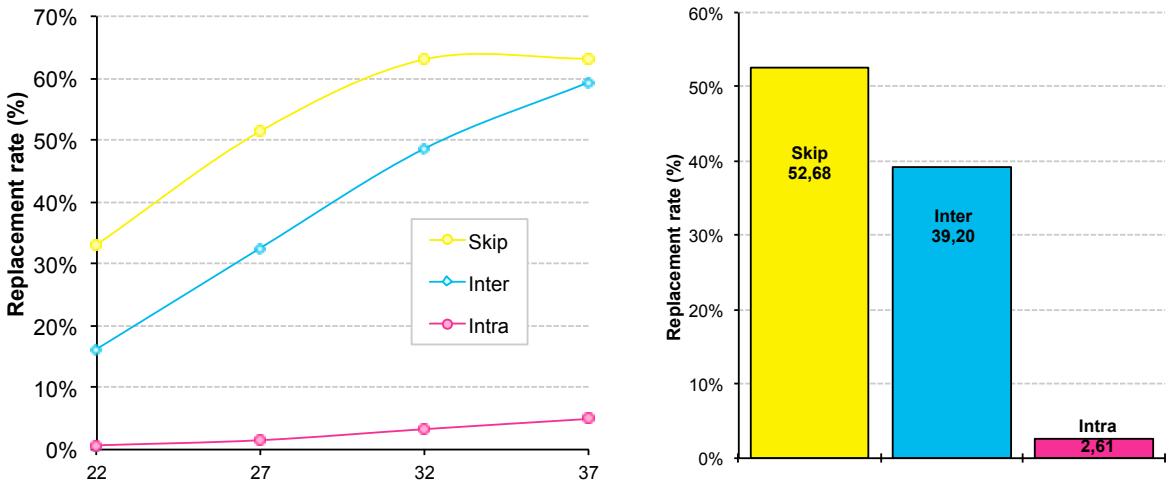


Figure 7.3.3: *Modes being replaced by the* motion tubes

Figure 7.3.3 shows, in average, which mode was previously used to code a *macroblock*, that is now coded by the *motion tubes*. It reads as follows: for all *macroblocks* now encoded by *motion tubes*, 35,3% of them were encoded by an inter coding mode in the initial coder. Here, it appears that the proportion of *macroblocks* encoded by the *motion tubes* originally coded through any classical inter-coding mode is far from negligible, which is very encouraging.

7.3.2.3 Replacement rates of the different coding modes

In order to precisely investigate the effect *motion tubes* have on classical coding modes, let us consider the problem the other way round. Instead of focusing on *macroblocks* which are encoded with the *motion tubes*, it is now proposed to measure in which proportions the occurrence of each classic mode is being replaced by *motion tubes*. Figure 7.3.4 shows how exactly each of the original decisions have been impacted by the *motion tubes*. Figure 7.3.4b reads as follows: from all the *macroblocks* initially encoded by mode *skip*, nearly 53% of them have been replaced by the *motion tube* prediction once these have been embedded into the coder. It reads as follows: for all *macroblocks* which were encoded by mode *skip* in the initial coder, 52,68% percent of them are now encoded by *motion tubes* in the modified coder.



(a) *Evolution of the mode replacement rates with respect to the QP*

(b) *Average replacement rates*

Figure 7.3.4: *Proportion of the initial decisions which have been turned into the favour of the* motion tubes

Several observation can be made from these figures. The ratio of *skip macroblocks* being replaced by the *motion tubes* prediction seems to reach its maximum when QP=32. This is in accordance with results showed in figure 7.3.2b. This tends to show that a maximum amount of *skip macroblocks* are turning in favour of the *motion tubes*.

As for *inter macroblocks*, the number of them being replaced by the *motion tubes* keeps increasing with the QP, which may be interpreted as a lack of efficiency from the *motion tubes* in areas typically encoded through *inter* modes. This is easily explained by the fact that *motion tubes* do not have any accurate textural correction mechanism, hence cannot compete with *inter* modes. This raises an interesting perspective: either provide the *motion tubes* with a textural correction mechanism, or integrate the *motion tubes* into all *inter* modes to benefit from their ability to correct the motion compensated prediction from a textural residue.

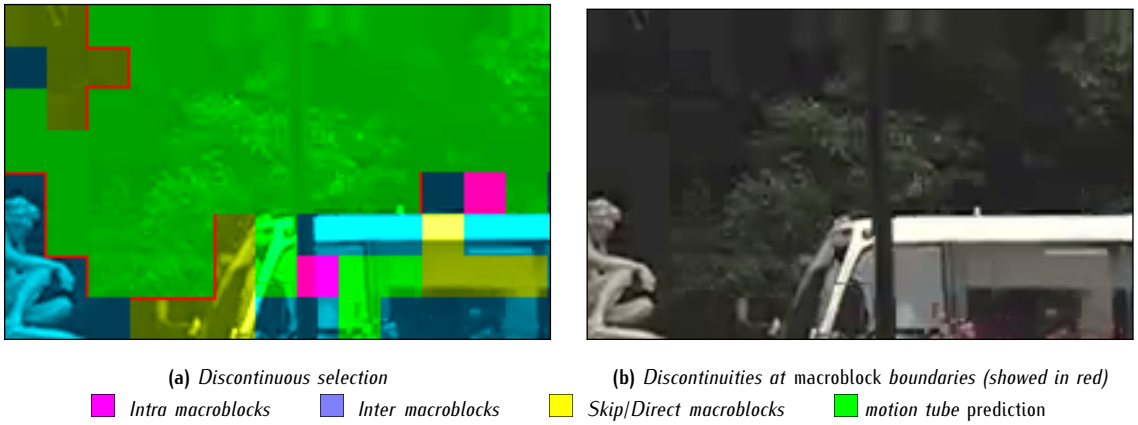
Finally, it also appears that, especially at low bitrates, a small proportion of the *intra macroblocks* are replaced by the *motion tube* prediction as well. However, the overall use of *intra* modes is constant whether *motion tubes* are used or not (see figure) 7.3.2b. This means that a similar amount of *macroblocks* initially coded by an *inter* mode, are now encoded by an *intra* mode. This might be explained by the lack of proper motion predictors: H.264/AVC predicts each motion vector from those of neighbouring *macroblocks*. With *motion tubes*, however, the *skip* motion vector is set to zero as *list 2* reference images are already motion compensated. In an attempt to address this issue, it was simply proposed to propagate through all tubes-*macroblocks* the motion vector from the last *inter macroblock*. At the end of a large region fully synthesized by the *motion tubes*, the propagated motion prediction is very likely not to be appropriate anymore. In these regions, as a perspective, one should replace H.264/AVC's motion predictions by those of the *motion tubes*.

7.3.3 Visual results and subjective evaluations

7.3.3.1 Different coding distortions for different coding modes

In transition areas where the coding mechanism has been changed from one *macroblock* to its neighbour, the encoded images may suffer from blocking effects. While the classical modes from H.264/AVC do not suffer that much from this problem, it appears that *motion tube* compression artefacts are quite different from those of the classical coding modes. The phenomenon is especially visible when a classical mode is providing a blurry texture, while neighbouring *motion tubes* reconstruction is much more faithful (at least, more textured), or vice-versa. This happens quite frequently as *motion tubes* may fail at reconstructing the images in some areas, while H.264/AVC will actually be able to provide a proper reconstruction. Conversely, H.264/AVC may fail where *motion tubes* might succeed.

Be that as it may, both scenarios are likely to introduce some blocking artefacts. The sudden change in reconstruction quality is also penalizing the overall perception. Figure 7.3.5 shows such areas on the top-left area of the seventh image of sequence *Bus*. Discontinuities are outlined in red on the mode selection map.



(a) *Discontinuous selection* **(b)** *Discontinuities at* macroblock *boundaries (showed in red)*

Figure 7.3.5: *How discontinuities in the selection introduces blocking effects and changes in textural quality*

7.3.3.2 A lack of temporal stability

While the selection maps are globally consistent across time, they are locally sporadic on transition regions. The same areas which were producing some blocking effects are also perturbing the temporal consistency of the sequences when they are played. Unfortunately, this effect cannot be illustrated on a piece of paper. Anyhow, a significant number of *macroblocks* located in these transitional areas are suffering from a succession of changes of coding technique. Both *motion tubes* and classical modes may provide the same objective quality, but end up in visually different reconstructions.

As a consequence, blocking artefacts previously mentioned keep appearing and disappearing, making *macroblocks* boundaries flickering, which is especially disturbing from a subjective perspective. This naturally introduces the next section: how to regularize the selection maps? Two main objectives may then be achieved:

1. from a coding perspective, the modified H.264/AVC coder may be greatly improved through the temporal regularization of its decision mechanism, hence using even more optimally the *motion tubes*;
2. from a *motion tube* perspective, above all, these selection maps and their regularization may turn into a huge step towards a life and death mechanism.

7.4 Accounting for the spatio-temporal nature of the motion tubes to drive their selection

Previous sections 7.2 and 7.3 provided invaluable information regarding the texture synthesis abilities of the *motion tubes*. In comparison to state-of-the-art compression techniques, it was shown that *motion tubes* can properly reconstruct a significant proportion of the image sequences, even though compression performances were not in favour of *motion tubes*.

However, it was also mentioned that H.264/AVC's decision mechanism is not suited to *motion tubes*. Standing back from previous investigation, it becomes obvious that H.264/AVC is effectively unable to catch the temporal dimension of the *motion tubes*. Indeed, the JSVM processes the sequences one frame at a time, and does not consider the whole GOP to drive its coding mode selection mechanism.

Section 7.4.1 will first provide a metric which measures how much consistent are H.264/AVC's decisions along the temporal dimension. Coupled with H.264/AVC's ability to assess the quality of the *motion tube* prediction, this additional metric turns out to be a valid quality-efficiency assessment metric for the *motion tubes*. Then, section 7.4.2 will propose an additional mechanism in charge of regularizing the selection of the *motion tubes* across time, thus encouraging the overall stability.

7.4.1 Towards an hybrid quality-stability metric for the motion tubes

Section 7.3.3.2 explained that encoded sequences suffer from a flickering effect, resulting from discontinuities of the selection map of the *motion tubes*. In order to evaluate the ability of a *motion tube* to provide a temporally coherent information, it is now proposed to measure how much consistent are H.264/AVC's decisions across time. This section will now introduce the concept of *stability* in the context of *motion tubes*. By quantifying the amount of temporal discontinuities in the mode selection maps, one may extract substantial information regarding the temporal coherence of the *motion tubes*, hence whether they should be terminated or not.

7.4.1.1 Preliminary study: a coarse stability measurement

Prior to the establishment of a tube-based temporal stability metric, one may ask how much exactly the selection maps are unstable across time. The modes selection maps $\mathcal{M}(x, y, t)$ are first binarized with respect to the *motion tubes*: `true` indicates that H.264/AVC has been using the *motion tubes* prediction, and `false` indicates that H.264/AVC has been using any of its classical coding modes. The temporal evolution of the modes selection map is categorized into two scenarios:

- ✓ **stable**: the *motion tubes* selection state is unchanged (from `false` to `false` or `true` to `true`);
- × **unstable**: the *motion tubes* selection state is changed (from `false` to `true` or `true` to `false`).

A coarse stability index is then given, for each time instant, by the percentage of *macroblocks* whose selection state has not been changed (first scenario). As the grid of the *macroblocks* and the positions and shapes of the *motion tubes* do not correspond, this index only provides an overall measurement of the stability of the selection of the *motion tubes*.

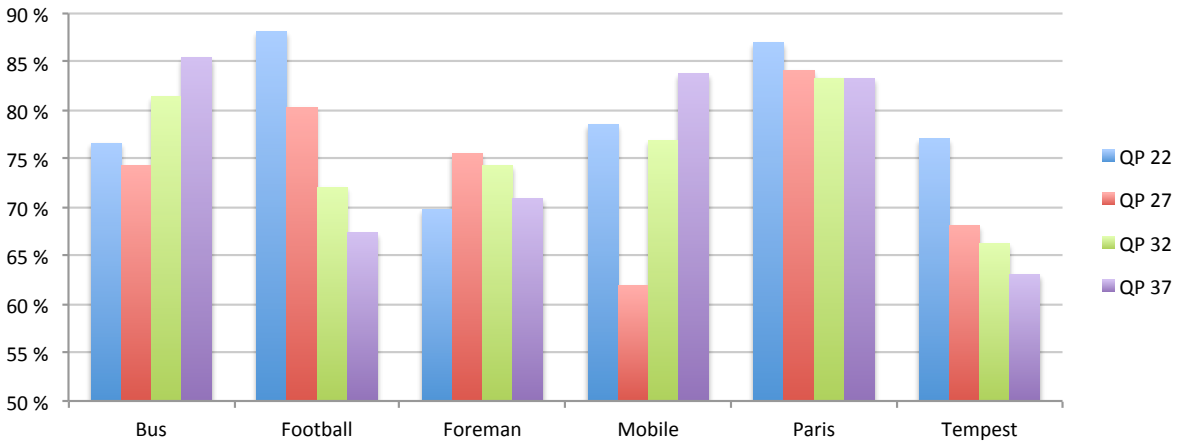


Figure 7.4.1: *Average coarse stability index for various sequences*

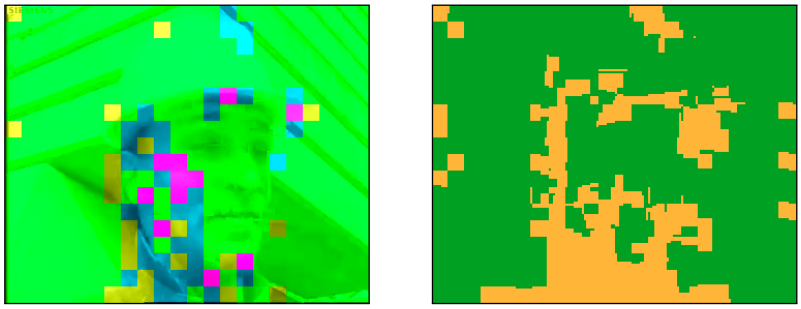
Figure 7.4.1 shows the average coarse stability for the usual sequences test set. As can be seen, the overall stability index is fluctuating between 60% and 90%. When the index is close to 100%, this means that the overall selection of the *motion tubes* is highly consistent and stable along the temporal axis. Conversely, when the index gets close to 50%, half of the *macroblocks* are changing of representation, hence likely to introduce some flickering artifact. In addition, it

can be seen that the stability index is not correlated to the QP: depending on the sequences, their relationships are very different. In the end, it appears that a large part of H.264/AVC's decisions introduce temporal discontinuities into the hybrid tube-AVC representation. This is responsible for the flickering effect mentioned in section 7.3.3.2. Following subsections will provide a more accurate stability metric to assess the stability of each *motion tube*.

7.4.1.2 Projecting the selection maps onto the motion tubes reference instant

Until now, the binarized mode selection maps $\mathcal{M}(x, y, t_{\text{cur}})$ only indicated whether *motion tubes* were or were not used to represent each *macroblock*. However, the grid of the *macroblocks* and the positions of the *motion tubes* do not correspond, preventing us from making a direct connection between the selection maps and the *motion tubes*. As a solution, it is proposed to project the selection map, for each time instant, onto the reference instant t_{ref} of the *motion tubes*. The projected selection map $\widetilde{\mathcal{M}}(x, y, t_{\text{cur}})$ is simply obtained through the backward motion compensation of $\mathcal{M}(x, y, t_{\text{cur}})$, and

$$\widetilde{\mathcal{M}}(x, y, t_{\text{cur}}) = w_{\text{cur}\to\text{ref}}\left(\mathcal{M}(x, y, t_{\text{cur}})\right) \tag{7.1}$$



(a) *Selection map $\mathcal{M}(x, y, t_4)$ for image I_4*

(b) *Projected selection map $\widetilde{\mathcal{M}}(x, y, t_4)$ for image I_4*

Figure 7.4.2: *Sequence* Foreman*: backward motion compensation of the selection map*

Provided that they all belong to the same family, *motion tubes* do not overlap at their reference instant, making possible to map each *motion tube* with its pixel-wise selection state. Figure 7.4.2 shows the selection map of the fifth image of sequence *Foreman*, and its projection at the *motion tubes* reference instant (a single family of *motion tubes* has been used). For each *motion tube*, green pixels from figure 7.4.2b are those that are being used by H.264/AVC, while orange pixels are those which are not used by H.264/AVC. From now on, the selection of each *motion tube* can be precisely computed, pixel by pixel, from the corresponding area of the projected selection maps.

7.4.1.3 Spatio-temporal selection rate and tube stability index

In order to quantify the temporal efficiency of each *motion tube*, two metrics are now introduced:

1. the **spatio-temporal selection rate** $\textsc{Sel}(t)$ indicates in which proportions each *motion tube* is selected by H.264/AVC;
2. the **spatio-temporal stability index** $\textsc{Stab}(t)$ indicates how stable are the selected areas across time.

a The spatio-temporal selection rate

Let $\mathbf{X}$ be an $M \times N$ *motion tube* instantiated at time instant t_{ref} and tracked across a GOP. For each time instant t_{cur}, its selection rate $\text{sel}(t_{\text{cur}})$ is given by the percentage of pixels from $\mathbf{\Omega}_{\mathbf{X}}(t_{\text{cur}})$ which are selected by H.264/AVC decision mechanism:

$$\textsc{Sel}(t_{\text{cur}}) = \frac{\text{card}\left(\mathrm{U}_{\mathbf{X}}(t_{\text{cur}})\right)}{M \cdot N} \tag{7.2}$$

where $\mathrm{U}_{\mathbf{X}}(t_{\text{cur}}) = \left\{(x, y) \in \widetilde{\mathcal{M}}\left(\mathbf{\Omega}_{\mathbf{X}}(t_{\text{ref}}), t_{\text{cur}}\right),\ \widetilde{\mathcal{M}}(x, y, t_{\text{cur}}) = \texttt{true}\right\}$ is the set of pixels from $\mathbf{X}$'s reconstruction which are selected by H.264/AVC. By computing the average value of $\textsc{Sel}(t_{\text{cur}})$ over the whole GOP, an overall spatio-temporal selection rate is obtained. Both instantaneous and average figures may need to be considered: the average selection rate may be used to decide whether to keep or not to keep a *motion tube*, while instantaneous figures may help detecting when *motion tubes* need to be deactivated, reactivated, or terminated.

b The spatio-temporal stability index

In order to quantify how much stable is the selection of each *motion tube* across time, it is proposed to measure how close to each other are the successive selection maps. It is proposed to compute for each time instant t_{cur} the number of pixels which are only selected at t_{cur} or only selected at $t_{cur}-1$. The stability index $\textsc{Stab}(t_{cur})$ writes as

$$\textsc{Stab}(t_{cur}) = \frac{\mathrm{card}\big(Y(t_{cur})\big)}{M \cdot N} = \frac{\mathrm{card}\big(U_X(t_{cur}) \,\Delta\, U_X(t_{cur}-1)\big)}{M \cdot N} \tag{7.3}$$

where $Y(t_{cur})$ is the set of pixels only selected at t_{cur} or only selected at $t_{cur}-1$, and Δ is the symmetric difference. Numerous other stability metrics could have also been considered; this one was chosen for its simplicity, and as a preliminary solution, but may be modified in future works to account for more specific evolutions of the selection. Figure 7.4.3 illustrates how $Y(t_{cur})$ is computed.

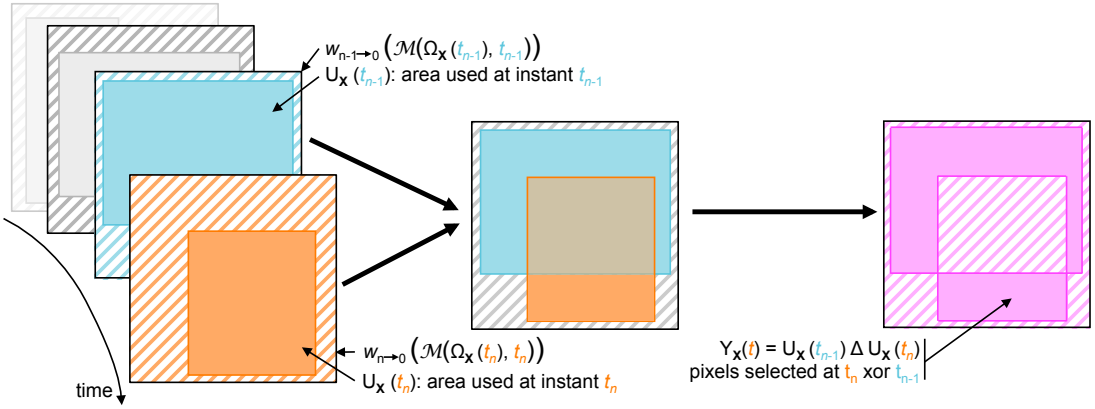
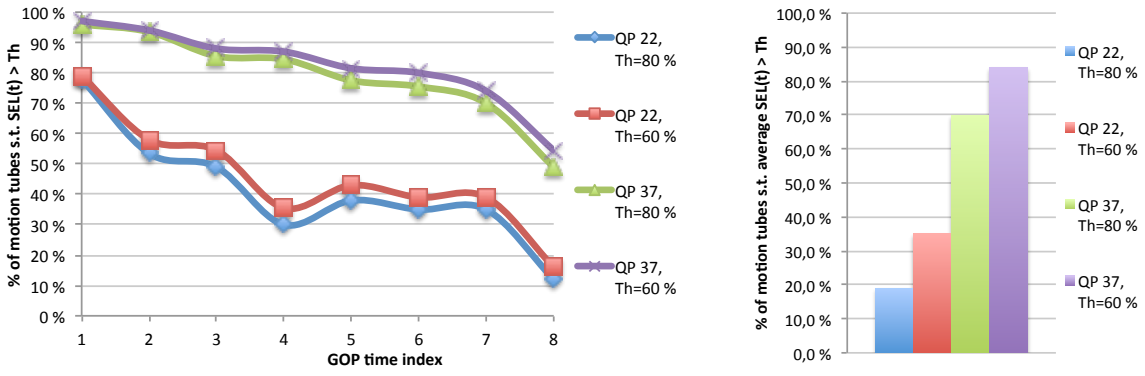


Figure 7.4.3: *Assessing the selection of each* motion tube *along the temporal axis*

7.4.1.4 A selection mechanism for the motion tubes based on the selection rate

As a first step towards a spatio-temporal selection mechanism, we now only keep *motion tubes* whose selection rate is higher than a given threshold Th. Figure 7.4.4 shows the proportion of *motion tubes* whose selection rate $\textsc{Sel}(t)$ is greater than 60 % and 80 %, at QPs 22 and 37 on sequence *Foreman* (similar results are observed for other sequences).

a Frame-by-frame versus tube-by-tube selections



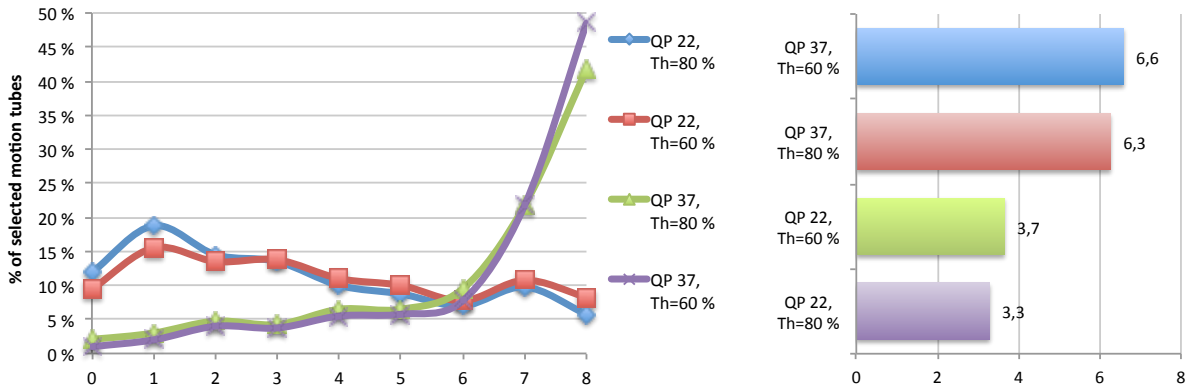
(a) *Instantaneous selection rate* **(b)** *Average selection rate*

Figure 7.4.4: *Percentage of* motion tubes *whose selection rate is higher than a threshold Th for sequence* Foreman

In figure 7.4.4a, *motion tubes* are selected on a frame by frame basis. The higher the QP, the higher the selection rate. Arbitrary thresholds have been used for validation purposes: further works should determine the best tradeoff between selection threshold and amount of synthesized textures.

In figure 7.4.4b, the selection is performed on a tube by tube basis. Only *motion tubes* whose average selection rate is higher than the threshold are kept. As can be seen, the overall selection rate is decreased in comparison to the frame by frame selection, thus providing a spatio-temporal decision mechanism.

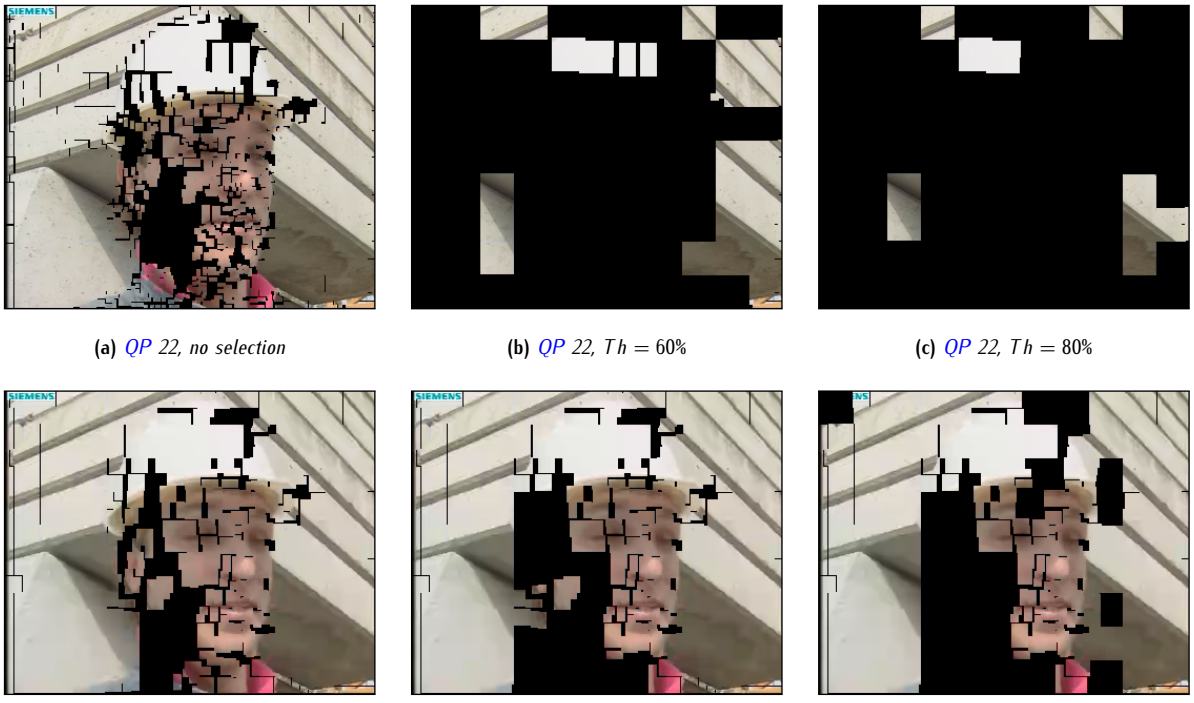
b Frame-by-frame selection: lifespan of selected motion tubes



(a) Motion tubes *duration histogram (time instants)* **(b)** *Average lifespan duration (time instants)*

Figure 7.4.5: *Impact of the selection on the lifespan of the* motion tubes

When *motion tubes* are selected on a frame-by-frame basis, their lifespan is impacted, thus decreasing the number of time instants they are active (in other words, their lifespan). Figure 7.4.5a shows the lifespan histogram, while figure 7.4.5b shows the average duration of the lifespan once *motion tubes* have been impacted by the selection. As can be seen, the lifespans are uniformly spread in between 0 and 8 time instants at high bit-rates (QP 22), while low bit-rates favour the use of *motion tubes* of long durations.



(a) *QP 22, no selection* **(b)** *QP 22,* $Th = 60\%$ **(c)** *QP 22,* $Th = 80\%$

(d) *QP 37, no selection* **(e)** *QP 37,* $Th = 60\%$ **(f)** *QP 37,* $Th = 80\%$

Figure 7.4.6: *Effect of the selection of the* motion tubes *on sequence* Foreman

c Visual results

Figure 7.4.6 shows how sequence *Foreman* is impacted when undesirable *motion tubes* have been removed from the representation. As could be expected, the most stable areas are located in the background, in other words, stable areas from the scene in terms of displacement and deformation. Whichever the QP, the selection is highly correlated to the location of the different elements from the scene. The higher the selection rate, the more unstable areas (areas undergoing complex deformation or displacement, areas suffering from occlusions, etc) are removed.

7.4.1.5 Measuring the motion tubes selection stability index: practical results

Figure 7.4.7 shows the histogram of the stability index STAB(t) of the *motion tubes* for sequence *Bus*. For each *motion tube*, STAB(t) has been averaged over the whole GOP. As can be seen, there are few *motion tubes* whose selection is not stable across time. In addition, this observation stands for all QPs. In other words, this means that the spatio-temporal selection rate may be, at least for now, sufficient enough to drive the selection of the *motion tubes*. As a perspective, however, this stability information may be advantageously used to refine the selection mechanism. Similar results are observed for other sequences from the test set.

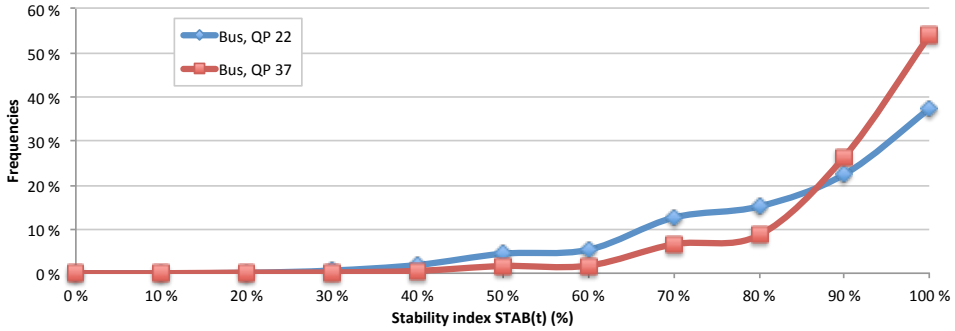


Figure 7.4.7: *Histogram of the stability index* STAB(t) *for sequence* Bus *at various QPs*

7.4.2 Improving the spatial coherence of the selection of the motion tubes

Previous sections introduced several selection mechanisms which may be used to drive the life and death of the *motion tubes*. At first, H.264/AVC video compression mechanism is used to assess the spatial quality of the reconstruction of the *motion tubes* by making them competing with other state-of-the-art coding mechanisms. The selection was then temporally processed to produce a spatio-temporal measurement of the quality and the stability of the *motion tubes*.

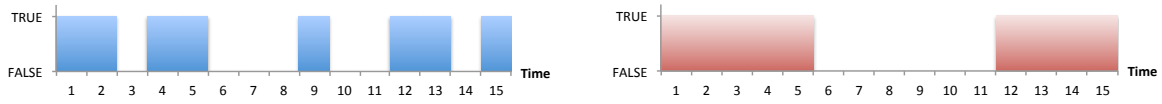
Preliminary visual results showed, in section 7.3.3.2, that the provided tube selection mechanism was suffering from the inability of the JSVM to optimize its decision along the temporal axis. This was later confirmed by the stability metrics provided in section 7.4.1. As a first step towards a spatio-temporal life and death mechanism, it is now proposed to regularize the selection maps in order to minimize the amount of discrepancies they hold along the temporal dimension.

7.4.2.1 Regularizing the tubes selection maps along the temporal axis

In order to minimize the amount of discrepancies from the selection maps, it is proposed to filter out inconsistent decisions detected along the temporal axis. The proposed filtering operation is illustrated in figure7.4.8, which plots the successive selection states of a *motion tube* X running across fifteen images. Two scenarios can be observed in figure 7.4.8a:

1. at t_3 and t_{14}, X is selected during several time instants, then deselected only for one instant, and reselected at next time instants
2. from t_6 to t_{11} included, X is not selected, except at t_9.

Both these situations should be avoided, hence minimizing the number of selection state changes, while still respecting the largest part of AVC's decisions. To this end, such scenarios are detected and corrected to end up in a temporally-coherent selection map as shown in figure 7.4.8b. In practice, the filtering operation is performed through a set of morphological operations (here, the structuring elements are unidimensional, and *close* then *open* operators were applied).



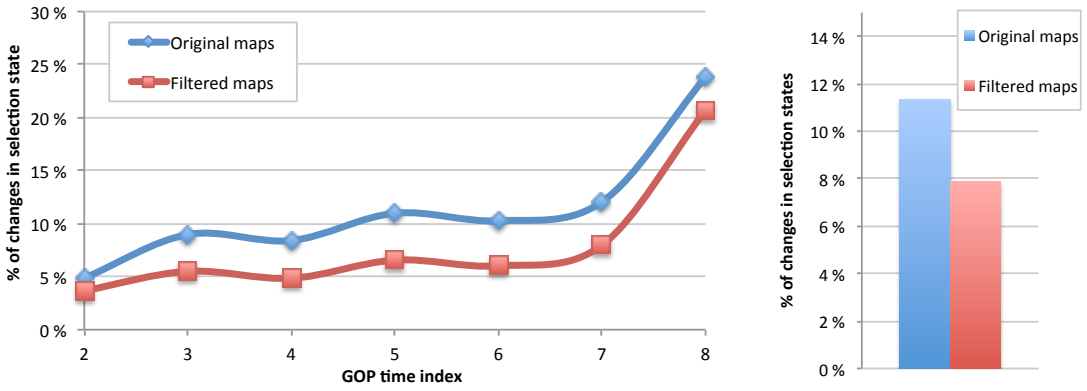
(a) *Original series of selection states* **(b)** *Filtered series of selection states*

Figure 7.4.8: *Filtering the selection maps along the temporal axis*

7.4.2.2 A significant reduction of unselections and reselections

In order to quantify the increase in temporal regularity, it is proposed to count the number of changes of selection states (from `true` to `false` or vice-versa) before and after the maps have been filtered. This is a very crude way to measure the gain in regularity, as the occurrence of a change in selection state is not bound to be a temporal discrepancy. As a perspective, one should input the filtered selection maps to H.264/AVC to further study their impact.

For now, figure 7.4.9 confirms the fact that filtering the selection maps did significantly reduce the number of selection changes, from 11.3 % to 7.9 % in average. These results have been obtained for sequences *Foreman* at QP 32. The original selection maps were computed from H.264/AVC's decisions: only *motion tubes* whose instantaneous selection rate $\textsc{Sel}(t)$ was greater than $Th = 80\%$ were activated. Similar results are observed for other QPs and sequences.



(a) *Instantaneous percentages* **(b)** *Average percentages*

Figure 7.4.9: *Percentage of* motion tubes *which are unselected or reselected in sequence* Foreman *at QP 32*

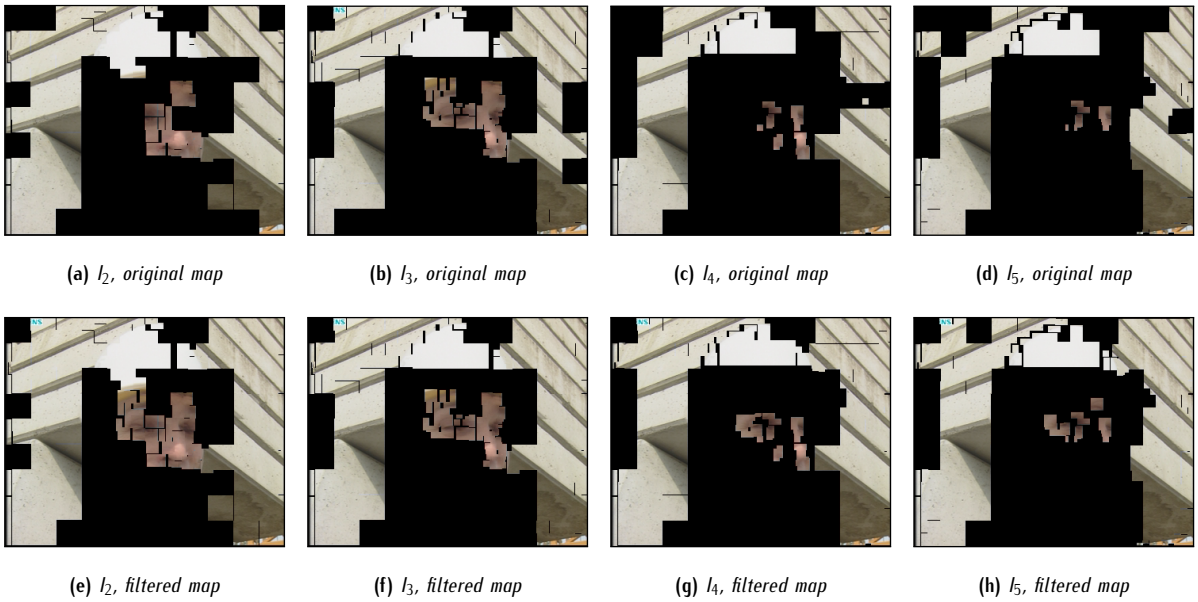
7.4.2.3 Temporal regularization of the selection: visual results

Even though the increase in stability is much more visible when actually playing the sequences, figure 7.4.10 still shows that the temporal evolution of the selection is much more stable once it has been filtered. The top row of images show the reconstructions of four successive images from sequence *Foreman* where the original tubes selection maps have been used. The bottom row show the same reconstructed images obtained once the selection maps have been filtered.

7.5 Conclusion

In previous chapters, all *motion tubes* were systematically tracked during the whole duration of the GOPs, whether the corresponding patches of textures could be tracked or not. Occlusions, complex deformations, or any other phenomena preventing an initial patch of texture from being tracked are especially problematic for the *motion tubes*. It is critical for the representation to be able to detect their occurrences and, in that event, deactivate or terminate the affected tubes.

In order to further validate the representation in regards to state-of-the-art coding tools, it was first proposed to compare the reconstruction provided by the *motion tubes* with the reconstruction provided by H.264/AVC compression standard. To this end, *motion tubes* were embedded into the JSVM. The decisions taken by the modified coder provided a large amount of invaluable informations regarding the *motion tubes* and their abilities. Several key features were highlighted:



(a) I_2, *original map* **(b)** I_3, *original map* **(c)** I_4, *original map* **(d)** I_5, *original map*

(e) I_2, *filtered map* **(f)** I_3, *filtered map* **(g)** I_4, *filtered map* **(h)** I_5, *filtered map*

Figure 7.4.10: *Effect of the regularization of the selection map on sequence* Foreman, *QP 32*

- *motion tubes* **were** eventually **used**, in very significant proportions, by the modified coder. This further **validates** the **concept** of *motion tubes*: they are able to provide a **proper representation** of a large part of the sequences;
- as could be expected, the more **stable** and **persistent** the images contents are across time, the more **suited** are *motion tubes*. Complex areas from the sequences, however, were generally better matched by classical coding tools.
- in terms of pure **compression**, however, the proposed hybrid compression scheme does **not** provide any **gains**; quite the contrary, actually. Still, the **coding cost** of the *motion tubes* can be **easily reduced** by **removing** the motion information of **unused tubes**: those poor compression performances may be drastically improved;
- the tube-based representation suffers from a **major issue**: its inability to **correct** the **predicted textures** through textural residues. For this reason, *motion tubes* were much more efficient at low bit-rates than at high bit-rates.

Besides these key points, embedding the *motion tubes* into H.264/AVC was also a first attempt at assessing the quality and the efficiency of the *motion tubes*. This information is crucial for an eventual decision mechanism in charge of the life and death of the *motion tubes*. The second half of this chapter was dedicated to this problem. Defining home made quality and efficiency metrics is definitely one of the most difficult problem raised by the proposed representation. By making the *motion tubes* competing with traditional coding techniques, one could at least know when and where it is better to use *motion tubes* instead of a more classical representation. From H.264/AVC's decisions, one may now be able to remove unused or inefficient *motion tubes* from the representation.

However, the decision mechanism of the JSVM is not optimal in regards to *motion tubes*. Indeed, *macroblocks* are usually processed on a frame by frame basis. Conversely, *motion tubes* need to be processed several time instant at once: the coding cost of a *motion tube* is spread over several time instants. In order to take the temporal axis into account, it was first proposed to measure the temporal coherence of the decisions taken by the JSVM in regards to the *motion tubes*, through their spatio-temporal selection rate $\textsc{Sel}(t)$ and the temporal stability of the selection $\textsc{Stab}(t)$. While the selection rate was clearly identified as a very efficient way to quantify the overall efficiency of a *motion tube*, the stability index showed that selected areas were locally stable across time.

Finally, a selection mechanism was designed based on the selection rate. By only keeping *motion tubes* whose overall or instantaneous selection rates are greater than an arbitrary threshold, one could discard *motion tubes* or portions of *motion tubes* which are not efficient enough. To further improve the selection mechanism, and clearly exhibiting and encouraging the temporal persistence of the selection, it was then proposed to regularize the selection maps along the temporal axis. Preliminary results showed that selection maps were visually much more stable. As an immediate perspective, this decision mechanism needs to be further assessed, and input to the modified coder in order to transmit required *motion tubes* only, hopefully reducing the loss in compression performances.

Conclusion

As a contribution to the field of video compression, this thesis provided a new representation for the image sequences. The representation focuses on the temporal axis and mostly aims at exhibiting the persistence of the textural information across time. Standing back on the literature, this work highlighted several issues specific to each of the available representations. *Motion tubes* then naturally emerged as potential candidates for a new representation which combines the advantages of several of its predecessors. Contributions of this thesis include a simple and advanced motion model, several mechanisms in charge of optimizing the synthesized images, and finally a first attempt at a life and death mechanism for the *motion tubes*.

Available representation and compression approaches: outcomes

Video compression aims at removing as much redundancies as possible from input sequences. Decorrelation techniques are in charge of identifying these redundancies, through numerous approaches which all focus on a specific type of redundancies (spatial or temporal, local or global, periodic or geometric, etc). Initially, images are input as a series of spatio-temporal samples. In order to exhibit the targeted redundancies, decorrelation techniques have introduced numerous alternatives to this initial representation.

Adressing spatial redundancies

The spatial domain has been extensively studied through still images representations, and numerous solutions have been provided to account for both local and global geometrical patterns, through the use of transforms. Among these transforms, the frequency transforms are the most popular ones, as they are both relatively low-computational and efficient. More recent transforms have been focusing on the space-frequency domain, and can be applied to signals of large dimensions and provide a continuous description of the image contents. The provided scalability is especially interesting in terms of representation features, but these representations suffer from their implementation complexity. Even more recently, space-frequency representations now locally model the geometrical patterns, but yet fail at efficiently compacting the images, or lack from scalability, and are generally computationally intensive.

Whichever way, the ability to locally describe the geometrical contents appears to be an essential feature. Consequently, the representation provided by this thesis locally accounts for the spatial contents through blocks of pixels. Not only will this enable local features to be accurately represented, but it will also guarantee the overall complexity to be reasonable.

How essential it is to efficiently capture the redundancies along the temporal axis

Most of the time, image sequences, account for the evolution of a scene across time. As a consequence, each image can be interpreted as a deformed version of its predecessor in the sequence. For this reason, the largest part of the redundancies is located along the temporal axis. Thus, video representations and compression schemes should especially pay attention to the temporal axis, through motion compensation.

Motion compensation has been extensively used for various purposes, including video analysis and (obviously) video compression. As a consequence, numerous approaches to motion compensation have been proposed. They all model the motion information (the motion field) in a specific way, and use various optimization techniques to compute their parameters. In an eye to compression, it is important for the motion field to be modelled with as few parameters as possible, while still providing a motion information which is accurate enough. With this in mind, geometrical motion models appeared as a reasonable tradeoff between motion accuracy and amount of information to transmit.

In particular, blocks and meshes were especially considered. With blocks, one may easily handle motion discontinuities, as there is not interdependencies between neighbouring blocks. However, they cannot handle continuities nor local deformations and stick to simple translations. Conversely, meshes naturally provide a continuous representation of the

motion field, and can model a large range of local deformations However, they handle motion discontinuities with difficulty. The representation provided by this thesis needs to account for both continuities and discontinuities of the motion field. Consequently, hybrid blocks/meshes motion representations were naturally advocated.

Available compression schemes: strenghts and weaknesses

More popular than ever, the classical approach to video compression is once again expected to be used in forthcoming HEVC video standard. Such a success is easily explained: its high compression performances, the relative simplicity of its implementation, and its ability to provide subjectively and objectively high reconstruction qualities. Also, it was built from a succession of incremental modifications, thus did not suffer from the addition of a disruptive technology delaying its application. However, it processes the temporal axis on a frame-by-frame basis, which is not in favour of a long-term decorrelation.

On the other hand, numerous disruptive approaches have been proposed as well, and address several issues raised by the classical representation. With MCTF and motion threads, they finally provided a way to process the temporal axis in a continuous manner. In addition, they often provide fully scalable schemes, which is nowadays considered to be an essential feature. However, they systematically perform the motion compensation, even though it might not produce a proper prediction, where an intra coding technique may be much more efficient. To paint a black picture of the situation, these techniques are often more computationally intensive than classical ones. Finally, they may increase the subjective quality, but generally lower the objective quality, which is still largely taken into account.

As a consequence, this thesis provided a representation which combines most of the advantages of both the classical compression scheme and the disruptive approaches as well. It provides a local description of the spatial contents, but processes the temporal axis in a continuous fashion. Our work most exclusively focused on the temporal axis, as it is our belief that most compression gains can be obtained by improving the temporal decorrelation.

Provided solution

With *motion tubes*, image sequences are interpreted as a set of patches of textures undergoing specific displacements and deformations along a given time interval. Indeed, each *motion tube* consists of an initial patch of texture, a lifespan during which the patch is available, and a set of warping operations describing its evolutions across time and space. In order to make their use easier, *motion tubes* were grouped into *families*: a reference image was partitioned into blocks, each of which initializing a *motion tube*.

With *motion tubes*, the motion information now consists of a set of spatio-temporal trajectories along with local deformations. The temporal persistence of the texture is then naturally exhibited by the representation. Several problems were raised by the representation, then addressed in dedicated chapters: the design of the motion model, the fact that *motion tubes* may not completely reconstruct the sequences, and finally the problem of their life and death.

The motion model: in between blocks and meshes

In order to efficiently track the displacements and deformations of individual patches of textures along the temporal axis, several features were requested: the ability to handle both continuities and discontinuities of the motion field, and the ability to represent both simple translations and more complex deformations. It was also strongly advised to entirely base the motion model on blocks, and avoid more complex geometric patterns such as meshes or even arbitrary regions. Naturally, the SOBMC appeared to be an appropriate model, as it hybrids a basic translational model (the BMC) and another model which handles slight deformations through simple translations of overlapped blocks of pixels (the OBMC). The OBMC, in particular, proved to be a low-computational version of the mesh-based CGI. In the end, the provided motion model is a variant of the SOBMC. It hybridizes four motion modes:

- a **disconnected** and **translational** model: a single motion vector is used to model the displacement of a patch of texture. As there is no interdependency with neighbouring patches, this **disconnected** mode naturally handles the discontinuities of the motion field;
- a **fully connected** model which handles **slight deformations**. Here, the motion vectors of neighbouring patches of textures are used to compute the deformation, such that a deformed patch of texture is kept **connected** to its neighbours, thus providing a continuous representation of the motion field;
- two **partially connected** models which respectively exhibit vertical and horizontal directions of connection, and also handle **slight deformations**.

In order to guarantee that the overall motion information is consistent across space and time, several regularization mechanisms were included. They notably include a rate-distortion optimization, along with a set of spatio-temporal motion predictors, in charge of discarding motion candidates which require too much information to be transmitted and often correspond to inconsistent parameters. Finally, the motion information was entropy coded through a state-of-the-art coder, the CABAC.

Dealing with textural synthesis issues

No matter how efficient is the motion model, synthesized images systematically suffer from various issues. With illumination changes, resolution losses, and other phenomena which alter the initial patch of texture, the textural information of a *motion tube* needs to be frequently refreshed in order to account for these changes. This thesis provided a crude solution to this issue through the introduction of *B-tubes*: instead of using a single textural reference instant, they source the texture from two reference instants. The final texture is then obtained by a weighted combination of both reference ones, which allows for textural variations to be crudely accounted for.

Another issue, much more problematic, is the fact that *motion tubes* are generally unable to entirely reconstruct the sequences. Two mechanisms were provided to address this problem. At first, *B-families* of *motion tubes* were used to register the textural information of a GOP through several families of *motion tubes*. A significant proportion of previously unreconstructed areas are now represented by additional *motion tubes*. Then, remaining holes were inpainted; the spatio-temporal information of the *motion tubes* can be used to drive this process.

Life and death of motion tubes

The last issue raised by the representation lies within its ability to decide when and where *motion tubes* need to be terminated. Indeed, there is no guarantee for a patch of texture to be effectively available throughout the whole duration of a GOP, neither for the provided motion model to be able to catch its displacement and deformation. May this happen, the corresponding *motion tube* needs to be terminated to not affect the overall compacity of the representation.

In order to detect any of these problematic scenarios, it was first proposed to assess the *motion tubes* through their integration into a state-of-the-art compression scheme: H.264/AVC. Since *motion tubes* were competing with traditional coding modes, the decisions of the modified coder could be analysed and showed when and where *motion tubes* were favourable. However, H.264/AVC's decisions are taken on a frame-by-frame basis, which does not correlate with the temporal nature of the *motion tubes*. As a first solution, it was proposed to measure the stability of the decisions along the temporal axis, and then to regularize these.

Discussion on the overall performance of the motion tubes

Representation features

This thesis proved that *motion tubes* are actually viable candidates to the representation of image sequences. Reconstructed images showed that, for a large proportion of the image sequences, *motion tubes* were able to properly synthesize the textures from a minimal amount of textural information. Their integration to H.264/AVC further proved how much interesting *motion tubes* may be in regards to more classic representations.

As expected, they also intrinsically exhibit the temporal persistence. In terms of representation, this turns out to be an invaluable information regarding the spatio-temporal content of the image sequences, and may greatly facilitate its analysis. Even though initial patches of textures cannot be interpreted as semantic regions, their evolutions will submit to those of the objects they belong to. A video object may and its temporal evolution may then be interpreted as an arbitrary set of *motion tubes*.

Compression features

The integration of the *motion tubes* into H.264/AVC showed that a significant amount of the *macroblocks* could be favourably coded by the *motion tubes*. However, the coding cost of the *motion tubes* was not taken into account in the mode selection process, as the instantaneous coding cost of a *motion tube* has little if no meaning. In the end, the gain in pure H.264/AVC bitrate dit not compensate for the coding of the *motion tubes*.

However, all *motion tubes* were transmitted, even though they were only partially or not selected at all by the modified coder. For this reason, it is essential for unused *motion tubes* to be discarded from the representation, thus reducing their coding cost. Hopefully, compression performances may turn in favour of the *motion tubes*. At the moment, however, there is no guarantee that their coding cost may be lowered to such an extent that the overall bitrate is decreased.

A viable representation which calls for numerous improvements

With *motion tubes*, this thesis provided a radical change to the way image sequences can be interpreted, processed, and compressed. From the start, they were designed in such a way that the proposed representation is not affected by problems which are inherent to existing schemes. However, such a disruptive approach comes to a cost: the tube-based representation is not mature enough yet to be advocated for practical use (*e.g.* as an answer to calls from normalization committees). Also, its ability to compact image sequences needs to be significantly increased for it to be seriously considered as an alternative.

Numerous issues are not yet addressed, or much too crudely (incompleteness of the representation, selection of the *motion tubes*). Also, modern compression schemes generally provide a set of additional tools to handle the large range of systems which can be used to shoot, transmit and display image sequences. Each of these tools, including scalability and multi-view coding, still require a significant amount of work to be designed. In its state, the proposed tube-based representation does not include any of these tools, even though a form of temporal scalability can be easily provided through *B-families* of *motion tubes*.

This thesis might have raised more questions than answers, and may be as well considered as a preliminary study which proved that *motion tubes* are a viable alternative to the problem of the representation and the compression of image sequences. However, the proposed representation is highly flexible, and exhibits numerous improvement axis. In terms of compression, we believe that *motion tubes* may match up to the traditional representation (H.264/AVC). Above all, *motion tubes* are subject to numerous perspectives, which, once completed, may turn the proposed representation into a valid and practical coding scheme. Next pages will summarize several of these perspectives.

Perspectives

While *motion tubes* abilities were thoroughly assessed by this thesis, there is still room for numerous improvements. Indeed, they threw back the whole compression problem into doubt, and forced us to reconsider the basic elements of a video compression scheme. A large part of our work has been dedicated to the construction of a simple and effective motion model; we consider it to be quite mature and do not plan any particular improvements worth mentioning here. The solutions we provided to complete and improve the synthesized images, conversely, are still relatively crude. Similarly, the life and death mechanism needs to be further evaluated, and most of all improved. Finally, it might also be very interesting to design additional features we believe *motion tubes* are capable of.

Towards optimal mechanisms in charge of compacting and transmitting the textures

A coding mechanism to transmit and refresh the textural information

Our work did not focus on the problem of the transmission of the textural information. With the integration of the *motion tubes* into H.264/AVC, we benefited from its intra coding mechanisms in charge of transmitting the textures. With *B-tubes*, the evolution of the textures was crudely accounted for, but without any guarantee that it actually improves the synthesized textures. However, for the representation to be completed, it is crucial to develop dedicated mechanisms which can precisely represent the textures and their evolution.

As an immediate perspective, one may modify the current integration into H.264/AVC to enable all inter coding modes to use the prediction provided by the *motion tubes*. Besides *skip* and *direct* modes, all other inter coding modes can refine the motion compensated prediction with a set of textural residues. Not only may this enable *motion tubes* to be precisely refreshed, but it also might drastically improve the overall gain in compression performances. In a similar way most *skip* and *direct macroblocks* have already been replaced by the *motion tubes* prediction, one may expect that remaining inter coding modes may favour this alternative motion prediction as well. In such case, the modified coder only needs to transmit the textural residues, and not the motion vectors anymore.

As a long term perspective, one may advocate for a coding scheme solely based on *motion tubes* to be designed. Their integration into H.264/AVC introduced numerous constraints which significantly limited the way *motion tubes* could be used. A whole new coding scheme will not be subject to these limitations anymore, but will require dedicated texture coding mechanisms which still need to be designed.

Dealing with multi-connected areas

While the problem of non-connected areas have been addressed through *B-families* of *motion tubes* and inpainting, the problem of multi-connected areas, conversely, was not considered. Wherever several *motion tubes* contribute to the reconstruction of the same area, the provided representation simply computed the final reconstruction as being the average of all contributions. This solution is not optimal, and preliminary tests show that significant gains can be obtained (approximatively $+0.5dB$ in average) if only the best contribution is used to synthesize these areas.

This calls to mind the problem of multiple description: how can we combine different versions of the same original signal in an optimal way? The literature, in the past few years, has been increasingly studying this problem. Provided solutions may be used to design an optimal composition operator for the *motion tubes*. As for the transmission of the texture and its evolutions, it may as well take into account the fact that multi-connected areas are areas where several blocks are overlapping. In such case, overlapped transforms (LOT, LBT) might be fairly interesting.

Maturing the life and death mechanism

Currently, the representation provided by this thesis mostly suffers from the crudeness of the selection mechanism in charge of the life and death of the *motion tubes*. This is especially important as it is responsible for the removal of all unneeded or inefficient *motion tubes*. The quality of the reconstructed images is affected by these erroneous *motion tubes*. Also, the overall coding cost may be significantly reduced once these tubes have been removed, especially because erroneous *motion tubes* are often those which require the most bitrate to be allocated.

Coder-dependent approach: further studies

As a short-term perspective, the provided selection mechanism needs to be completed and fully assessed. So far, H.264/AVC's decisions have been used to activate or deactivate each *motion tube*. More than ever, it is essential to alter the coding process of the *motion tubes*, such that discarded ones are not transmitted anymore.

In addition, the regularized selection maps also need to be assessed. So far, it has only been proved that it effectively reduced the occurrences of temporal discrepancies in the selection. Yet, there is no guarantee that it effectively improves the encoded sequences and reduces the flickering effect they are suffering from. As a solution, it is proposed to input the regularized selection maps to H.264/AVC, in order to drive its decision and force or prevent the use of *motion tubes* according to the selection state. Also, it may be fairly interesting to compute the impact of the regularization on the compression performances.

A coder-independent life and death mechanism

Ultimately, it would be very handy to design an home-made metric dedicated to the assessment of the *motion tubes*. Parameters which may need to be accounted for include: the distortion, the motion coding cost, the regularity and the smoothness of the trajectories, their amount of redundancy with respect to other *motion tubes*, to cite only a few of them.

The coder-dependent selection mechanism can act as a reference selection. One may then progressively build a quality and efficiency metric purely based on *motion tubes*, while ensuring that it correlates with the coder-dependent selection. This is another step towards a compression scheme solely based on *motion tubes*.

Additional features provided by the representation

Scalability

It is crucial for nowadays video compression schemes to integrate some scalability abilities. To this end, the image sequences are split into a base layer and several successive enhancement layers: in other words, image sequences are reconstructed from several contributions. In that, *motion tubes* are naturally able to provide several contributions for the same area, hence may naturally be organized into a fully scalable representation. As an immediate approach to scalability, one may consider that, in *B-families* of *motion tubes*, each additional family acts as an enhancement layer. Alternatively,

- **temporal scalability** and *motion scalability* can be provided through successive refinements of the trajectories of the *motion tubes*;
- **spatial scalability** can be provided through *motion tubes* of different sizes or resolutions;
- **quality scalability** can be provided through *motion tubes* of different qualities and/or through additional *motion tubes* holding residual textures.

3D video compression

Motion tubes aim at tracking the evolution of a patch of texture across time, as it is very likely to be available in several successive images. Similarly, in multiple view sequences, a patch of texture is likely to be observed in several successive views. In [CPML10], Colleu introduced a new representation of multiple view sequences through a polygon soup: the 3D scene is modelled by a set of variable-depth polygons (quads). The temporal axis, however, was not studied during the course of his study. To further compact his representation, one may instantiate, for each quad or group of quads, a *motion tube* in charge of describing the temporal evolution of the considered quads.

Eased video segmentation and description

Video analysis has been extensively focusing on finding solutions to extract semantic information. In particular, this information includes the shape, the displacements and the deformations of the elements of the scene. The required spatio-temporal segmentation still is problematic and often computationally intensive.

With variable-size *motion tubes*, it was provided a way to adapt the size and shape of the *motion tubes* according to the spatial contents and their temporal evolution. By grouping *motion tubes* into arbitrary sets which all share common spatio-temporal features, one might be able to identify and locate the different elements of the video in a continuous fashion.

Appendices

Appendix A

How OTMC and CGI motion modes are related to each other

While designing the motion model of the *motion tubes*, as was seen in chapter 5, both OBMC and CGI have been greatly influencing the final design. In practice, the provided motion model is a modifed version of the SOBMC (see section 2.4.3.2), though the LAOTMC motion mode lies halfway between the OTMC and the CGI (see section 5.3.4).

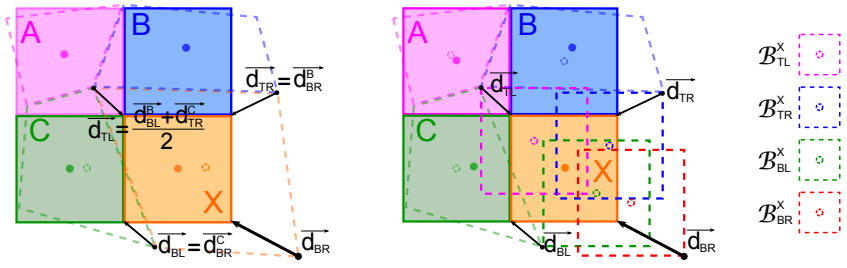
Section A.1 will demonstrate that OTMC and CGI are equivalent under certain hypothesis when performing *backward* motion compensation. From this, it will be deduced that OTMC is no more than a low-computational version of the CGI, thus corroborating the fact that the proposed LAOTMC mode is very likely to be close to the CGI. When it comes to *forward* motion section, however, section A.2 will show that CGI and OTMC are not equivalent anymore, as their respective weighting coefficient do not match. Then, it will be proposed to force the OTMC to use CGI weighting coefficients, which, in practice, does not prove to be efficient.

A.1 Backward motion compensation: identifying the CGI to the OTMC

In order to match the OTMC to the CGI, it is proposed to consider the deformation of a square block $\mathcal{B}_{\text{ref}}$ (see figure A.1.1). Its deformation is then described in two different ways:

1. with CGI, a quadrangular mesh is in charge of modeling the deformation;
2. with OTMC, four motion vectors are associated to each corner of the block.

Let $\mathcal{M}_{\mathcal{T}}$ be a *motion tube* initialized at t_{ref}, such that its support $\mathbf{\Omega}_{\mathcal{M}_{\mathcal{T}}}(t_{\text{ref}})$ is a square block $\mathcal{B}_{\text{ref}}$ at the very same instant. It is now proposed to compare CGI and OTMC compensation processes of $\mathbf{\Omega}_{\mathcal{M}_{\mathcal{T}}}(t_{\text{ref}})$ into $\mathbf{\Omega}_{\mathcal{M}_{\mathcal{T}}}(t_n)$. In both case, the displacements of the corners are given by motion vectors $\overrightarrow{d_{\text{TL}}}$, $\overrightarrow{d_{\text{TR}}}$, $\overrightarrow{d_{\text{BL}}}$ (the bottom-right motion vectors of three *motion tubes* from the causal neighbourhood) and $\overrightarrow{d_{\text{TR}}}$ (the bottom-right motion vector of $\mathbf{\Omega}_{\mathcal{M}_{\mathcal{T}}}(t_{\text{ref}})$).



(a) *CGI motion compensation* **(b)** *OTMC motion compensation*

Figure A.1.1: *A focus on CGI and OTMC approaches to motion compensation*

A.1.1 Overlapped Tube Motion Compensation: a reminder

With OTMC, the *backward* motion compensation consists of the projection and the weighting of four blocks $\mathcal{B}_{i,\text{cur}}$, $0 \leqslant i \leqslant 3$, onto a single reference block $\mathcal{B}_{\text{ref}}$. Let, $\mathcal{P}_{\text{ref}} = I_{\text{ref}}(P_{\text{ref}})$ be a pixel of $\mathcal{B}_{\text{ref}}$ located at coordinates $P_{\text{ref}} = [x_{\text{ref}}, y_{\text{ref}}]^T$. At t_{ref}, $\mathcal{P}_{\text{ref}}$ is predicted from a weighted average of four pixels from I_{cur}, whose locations are given by four displacement vectors $\overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{BL}}}(t_{\text{cur}})$ and $\overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$ -see equation (5.20)-. Hence, $\forall P_{\text{ref}} = [x_{\text{ref}}, y_{\text{ref}}]^T \in \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$,

$$\begin{aligned}\overline{I_{\text{ref}}}^{\text{OTMC}}(P_{\text{ref}}) = \quad & W_{\text{TL}}^{\text{OTMC}}(P_{\text{ref}}) \cdot I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + W_{\text{TR}}^{\text{OTMC}}(P_{\text{ref}}) \cdot I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TR}}}(t_{\text{cur}})\right) \\ & + W_{\text{BL}}^{\text{OTMC}}(P_{\text{ref}}) \cdot I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{BL}}}(t_{\text{cur}})\right) + W_{\text{BR}}^{\text{OTMC}}(P_{\text{ref}}) \cdot I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{BR}}}(t_{\text{cur}})\right)\end{aligned} \tag{A.1}$$

where $W_{\text{TL}}^{\text{OTMC}}(P)$, $W_{\text{TR}}^{\text{OTMC}}(P)$, $W_{\text{BL}}^{\text{OTMC}}(P)$ and $W_{\text{BR}}^{\text{OTMC}}(P)$ are the values of the four OTMC weighting window (see figure 5.3.5) at location $P = [x, y]^T$. From equation (A.1), it can be seen that the OTMC directly uses the motion vectors associated to the corners of $\mathcal{B}_{\text{ref}}$, and then interpolates the motion compensated intensity from four original pixellic intensities sourced from I_{cur}. As a single interpolation step is required, such a compensation technique requires a small amount of computations to be performed.

A.1.2 Working our way through Control Grid Interpolation

Unlike the OTMC, the Control Grid Interpolation directly sources the pixellic intensity from a single motion compensated location. The latter is obtained through an interpolation of the four motion vectors $\overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{BL}}}(t_{\text{cur}})$ and $\overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$. At position $P_{\text{ref}} = [x_{\text{ref}}, y_{\text{ref}}]^T$, the reference image I_{ref} is motion compensated by $I_{\text{cur}}(P_{\text{cur}})$, where $P_{\text{cur}} = [x_{\text{cur}}, y_{\text{cur}}]^T$ are the collocated coordinates of P_{ref} in $I_{\text{cur}}(P_{\text{cur}})$. P_{cur} results from a translation of P_{ref} by a displacement vector $\overrightarrow{d_{\text{CGI}}}(t_{\text{cur}})$ which is interpolated from $\overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $\overrightarrow{d_{\text{BL}}}(t_{\text{cur}})$ and $\overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$. In particular, $\forall P = [x_{\text{ref}}, y_{\text{ref}}]^T \in \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$

$$\begin{aligned}\overline{I_{\text{ref}}}^{\text{CGI}}(P_{\text{ref}}) &= I_{\text{cur}}(P_{\text{cur}}) \\ &= I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{CGI}}}(t_{\text{cur}})\right) \\ &= I_{\text{cur}}\left(P_{\text{ref}} + W_{\text{TL}}^{\text{CGI}}(P_{\text{ref}}) \cdot \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + W_{\text{TR}}^{\text{CGI}}(P_{\text{ref}}) \cdot \overrightarrow{d_{\text{TR}}}(t_{\text{cur}}) + W_{\text{BL}}^{\text{CGI}}(P_{\text{ref}}) \cdot \overrightarrow{d_{\text{BL}}}(t_{\text{cur}}) + W_{\text{BR}}^{\text{CGI}}(P_{\text{ref}}) \cdot \overrightarrow{d_{\text{BR}}}(t_{\text{cur}})\right)\end{aligned} \tag{A.2}$$

where $W_{\text{TL}}^{\text{CGI}}(P)$, $W_{\text{TR}}^{\text{CGI}}(P)$, $W_{\text{BL}}^{\text{CGI}}(P)$ and $W_{\text{BR}}^{\text{CGI}}(P)$ are the inner-mesh interpolation functions of the four control points. As resulting location may not coincide with an integer pixellic position, a further interpolation step may be required to obtain the appropriate pixellic intensity from neighbouring pixels. In the end, up to three interpolation steps may be required to motion compensate each pixel, which, in terms of complexity, is much more computationally demanding than the OTMC. Since influence functions are normalized, such that $W_{\text{TL}}(x, y) + W_{\text{TR}}(x, y) + W_{\text{BL}}(x, y) + W_{\text{BR}}(x, y) = 1$, equation (A.2) can be rewritten as:

$$\begin{aligned}\overline{I_{\text{ref}}}^{\text{CGI}}(P_{\text{ref}}) = I_{\text{cur}}\Big(\quad & W_{\text{TL}}^{\text{CGI}}(P_{\text{ref}}) \cdot \left[P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right] + W_{\text{TR}}^{\text{CGI}}(P_{\text{ref}}) \cdot \left[P_{\text{ref}} + \overrightarrow{d_{\text{TR}}}(t_{\text{cur}})\right] \\ & + W_{\text{BL}}^{\text{CGI}}(P_{\text{ref}}) \cdot \left[P_{\text{ref}} + \overrightarrow{d_{\text{BL}}}(t_{\text{cur}})\right] + W_{\text{BR}}^{\text{CGI}}(P_{\text{ref}}) \cdot \left[P_{\text{ref}} + \overrightarrow{d_{\text{BR}}}(t_{\text{cur}})\right] \quad \Big)\end{aligned} \tag{A.3}$$

A.1.2.1 Expanding the images into Taylor series

Intuitively, it can be seen that whenever the initial block $\mathcal{B}_{\text{ref}} = \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ is too much distorted, the shape of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$ will greatly vary whether CGI or OTMC motion models are used. As a consequence, CGI and OTMC cannot be identified under such circumstances. With small enough deformations, however, the shape of $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_n)$ will be similar whichever the provided motion model.

From this intuition, it is proposed to perform local approximations of the images which will ease the identification of the CGI to the OTMC. Let first assume that image $I_n(x, y)$ is infinitely derivable in the neighbourhood of point $P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) = [x_{\text{ref}} + dx_{\text{cur, TL}}, y_{\text{ref}} + dy_{\text{cur, TL}}]^T$. The Taylor series expansion of $I_n(x, y)$ around point $P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$ then writes as

$$\begin{aligned}I_{\text{cur}}^{\text{TL}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) &= I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + \sum_{p=1}^{\infty} \frac{1}{p!} \cdot \left\langle \overrightarrow{d_\varepsilon}, \nabla \right\rangle^p (I_{\text{cur}}) \left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) \\ &= I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_\varepsilon}^T \cdot \nabla\right](I_{\text{cur}}) \left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + o\left(\overrightarrow{d_\varepsilon}\right)\end{aligned} \tag{A.4}$$

where $\overrightarrow{d_\varepsilon} = [dx_\varepsilon, dy_\varepsilon]^T$ is a small displacement vector whose amplitude $\left\|\overrightarrow{d_\varepsilon}\right\|$ is inferior to R_{TL}, the radius of convergence of the Taylor series of $I_{\text{cur}}(x, y)$ around $P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$. Similarly, it is now assumed that $\overrightarrow{\nabla I_{\text{cur}}}(x, y)$ is infinitely derivable around point $P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}$. Consequently, the Taylor expansion of $\overrightarrow{\nabla I_{\text{cur}}}(x, y)$ then writes around this point as

$$\begin{aligned}\overrightarrow{\nabla I_{\text{cur}}}^{\text{TL}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) =& \overrightarrow{\nabla I_{\text{cur}}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) - \sum_{p=1}^{\infty} \frac{1}{p!} \cdot \left\langle \overrightarrow{d_\varepsilon}, \nabla \right\rangle^p \left(\overrightarrow{\nabla I_{\text{cur}}}\right)\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) \\ =& \overrightarrow{\nabla I_{\text{cur}}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) - \left[\left(\nabla \cdot \nabla^T\right) \cdot \overrightarrow{d_\varepsilon}\right](I_{\text{cur}})\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) + o\left(\overrightarrow{d_\varepsilon}\right)\end{aligned} \tag{A.5}$$

where $\nabla\nabla^T$ is the Hessian matrix of dimension two. Again, it is assumed that $\overrightarrow{d_\varepsilon}$'s amplitude is inferior to the radius of convergence $R_{\nabla\text{TL}}$. Replacing the gradient $\overrightarrow{\nabla I_{\text{cur}}}$ in equation (A.4) by its Taylor expansion from equation (A.5), the value of $I_{\text{cur}}(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon})$ is now given by

$$\begin{aligned}I_{\text{cur}}^{\text{TL}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) =& I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_\varepsilon}^T \cdot \nabla\right](I_{\text{cur}})\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) \\ & - \left[\overrightarrow{d_\varepsilon}^T \cdot \nabla \cdot \nabla^T \cdot \overrightarrow{d_\varepsilon}\right](I_{\text{cur}})\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) + o\left(\overrightarrow{d_\varepsilon}\right) \\ =& I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_\varepsilon}^T \cdot \nabla\right](I_{\text{cur}})\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}}) + \overrightarrow{d_\varepsilon}\right) + o\left(\overrightarrow{d_\varepsilon}\right)\end{aligned} \tag{A.6}$$

A.1.2.2 Local approximations of the pixel intensities around CGI control points

As it was earlier assumed that the deformation undergone by the initial block $\mathcal{B}_{\text{ref}} = \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ was rather small, it follows that P_{cur} (the collocated position of P_{ref}) and $P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$ (the translation of P_{ref} by the displacement of the top-left control point) are likely to be very close. In particular, assuming that

$$\left\|\overrightarrow{d_{\text{CGI}}}(t_{\text{cur}}) - \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right\| < \min\left(R_{\text{TL}}, R_{\nabla\text{TL}}\right) \quad , \tag{A.7}$$

one may apply equation (A.6) to $\overrightarrow{d_{\varepsilon,\text{TL}}} = \overrightarrow{d_{\text{CGI}}}(t_{\text{cur}}) - \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})$, which results in

$$I_{\text{cur}}^{\text{TL}}(P_{\text{cur}}) = I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TL}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_{\varepsilon,\text{TL}}}^T \cdot \nabla\right](I_{\text{cur}})(P_{\text{cur}}) - \left[\overrightarrow{d_{\varepsilon,\text{TL}}}^T \cdot \nabla \cdot \nabla^T \cdot \overrightarrow{d_{\varepsilon,\text{TL}}}\right](I_{\text{cur}})(P_{\text{cur}}) + o\left(\overrightarrow{d_{\varepsilon,\text{TL}}}\right) \tag{A.8}$$

Similarly, equivalent expressions for $I_{\text{cur}}(P_{\text{cur}})$ can be obtained from the Taylor expansions of $I_{\text{cur}}(x, y)$ around points $P_{\text{ref}} + \overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $P_{\text{ref}} + \overrightarrow{d_{\text{BL}}}(t_{\text{cur}})$ and $P_{\text{ref}} + \overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$; respectively $I_{\text{cur}}^{\text{TR}}(P_{\text{cur}})$, $I_{\text{cur}}^{\text{BL}}(P_{\text{cur}})$ and $I_{\text{cur}}^{\text{BR}}(P_{\text{cur}})$. Their radius of convergence are respectively written $\min(R_{\text{TR}}, R_{\nabla\text{TR}})$, $\min(R_{\text{BL}}, R_{\nabla\text{BL}})$ and $\min(R_{\text{BR}}, R_{\nabla\text{BR}})$.

$$\left\{\begin{array}{rcl} I_{\text{cur}}^{\text{TR}}(P_{\text{cur}}) & = & I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{TR}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_{\varepsilon,\text{TR}}}^T \cdot \nabla\right](I_{\text{cur}})(P_{\text{cur}}) + o\left(\overrightarrow{d_{\varepsilon,\text{TR}}}\right) \\ I_{\text{cur}}^{\text{BL}}(P_{\text{cur}}) & = & I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{BL}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_{\varepsilon,\text{BL}}}^T \cdot \nabla\right](I_{\text{cur}})(P_{\text{cur}}) + o\left(\overrightarrow{d_{\varepsilon,\text{BL}}}\right) \\ I_{\text{cur}}^{\text{BR}}(P_{\text{cur}}) & = & I_{\text{cur}}\left(P_{\text{ref}} + \overrightarrow{d_{\text{BR}}}(t_{\text{cur}})\right) + \left[\overrightarrow{d_{\varepsilon,\text{BR}}}^T \cdot \nabla\right](I_{\text{cur}})(P_{\text{cur}}) + o\left(\overrightarrow{d_{\varepsilon,\text{BR}}}\right) \end{array}\right. \tag{A.9}$$

where (resp.) $\overrightarrow{d_{\varepsilon,\text{TR}}} = \overrightarrow{d_{\text{CGI}}}(t_{\text{cur}}) - \overrightarrow{d_{\text{TR}}}(t_{\text{cur}})$, $\overrightarrow{d_{\varepsilon,\text{BL}}} = \overrightarrow{d_{\text{CGI}}}(t_{\text{cur}}) - \overrightarrow{d_{\text{cur, BL}}}$ and $\overrightarrow{d_{\varepsilon,\text{BR}}} = \overrightarrow{d_{\text{CGI}}}(t_{\text{cur}}) - \overrightarrow{d_{\text{BR}}}(t_{\text{cur}})$ are displacement vectors whose amplitudes are inferior to (resp.) $\min(R_{\text{TR}}, R_{\nabla\text{TR}})$, $\min(R_{\text{BL}}, R_{\nabla\text{BL}})$ and $\min(R_{\text{BR}}, R_{\nabla\text{BR}})$.

A.1.2.3 Towards an OBMC-like formulation of the CGI

Let now re-introduce the CGI inner-mesh interpolation functions. Assuming that $W_{\text{TL}}^{\text{CGI}}(x, y)$, $W_{\text{TR}}^{\text{CGI}}(x, y)$, $W_{\text{BL}}^{\text{CGI}}(x, y)$ and $W_{\text{BR}}^{\text{CGI}}(x, y)$ are normalized over $\mathcal{B}_{\text{ref}}$, the following weighted average

$$W_{\text{TL}}^{\text{CGI}}(P_{\text{ref}}) \cdot I_{\text{cur}}^{\text{TL}}(P_{\text{cur}}) + W_{\text{TR}}^{\text{CGI}}(P_{\text{ref}}) \cdot I_{\text{cur}}^{\text{TR}}(P_{\text{cur}}) + W_{\text{BL}}^{\text{CGI}}(P_{\text{ref}}) \cdot I_{\text{cur}}^{\text{BL}}(P_{\text{cur}}) + W_{\text{BR}}^{\text{CGI}}(P_{\text{ref}}) \cdot I_{\text{cur}}^{\text{BR}}(P_{\text{cur}}) \tag{A.10}$$

is equal to $I_{cur}(P_{cur})$. Further developing the calculations, the sum (A.10) is expanded into:

$$\begin{aligned} I_{cur}(P_{cur}) = \; & W_{TL}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{TL}}(t_{cur})\right) + W_{TR}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{cur,\,TR}}\right) \\ & + W_{BL}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{cur,\,BL}}\right) + W_{BR}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{cur,\,BR}}\right) \\ & + \left[\underbrace{\left(W_{TL}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{\varepsilon,TL}}^{T} + W_{TR}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{\varepsilon,TR}}^{T} + W_{BL}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{\varepsilon,BL}}^{T} + W_{BR}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{\varepsilon,BR}}^{T} \right)}_{\kappa} \cdot \nabla \right] (I_{cur})(P_{cur}) \\ & + o\left(\overrightarrow{d_{\varepsilon,TL}}\right) + o\left(\overrightarrow{d_{\varepsilon,TR}}\right) + o\left(\overrightarrow{d_{\varepsilon,BL}}\right) + o\left(\overrightarrow{d_{\varepsilon,BR}}\right) \end{aligned} \tag{A.11}$$

The weighted sum κ is then simplified into

$$\begin{aligned} \kappa = \; & \underbrace{\left(W_{TL}^{CGI}(P_{ref}) + W_{TR}^{CGI}(P_{ref}) + W_{BL}^{CGI}(P_{ref}) + W_{BR}^{CGI}(P_{ref}) \right)}_{1} \cdot \overrightarrow{d_{CGI}}^{T}(t_{cur}) \\ & - W_{TL}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{cur,\,TR}} - W_{TR}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{TR}}(t_{cur}) - W_{BL}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{BL}}(t_{cur}) - W_{BR}^{CGI}(P_{ref}) \cdot \overrightarrow{d_{BR}}(t_{cur}) \\ = \; & \overrightarrow{d_{CGI}}(t_{cur}) - \overrightarrow{d_{CGI}}(t_{cur}) \\ = \; & \overrightarrow{0} \end{aligned} \tag{A.12}$$

From this development, equation (A.11) is drastically simplified. Assuming that $o\left(\overrightarrow{d_{\varepsilon,TL}}\right)$, $o\left(\overrightarrow{d_{\varepsilon,TR}}\right)$, $o\left(\overrightarrow{d_{\varepsilon,BL}}\right)$ and $o\left(\overrightarrow{d_{\varepsilon,BR}}\right)$ are small enough compared to other terms, the CGI prediction $I_{ref}^{CGI}(P_{ref})$ is finally approximated to

$$\begin{aligned} \overline{I_{ref}}^{CGI}(P_{ref}) \approx \; & W_{TL}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{TL}}(t_{cur})\right) + W_{TR}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{TR}}(t_{cur})\right) \\ & + W_{BL}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{BL}}(t_{cur})\right) + W_{BR}^{CGI}(P_{ref}) \cdot I_{cur}\left(P_{ref} + \overrightarrow{d_{BR}}(t_{cur})\right) \end{aligned} \tag{A.13}$$

which is equivalent to the OTMC prediction -see equation (A.1)- assuming that the CGI inner-mesh interpolation functions are equal to the OBMC weighting windows:

$$\forall (x,y) \in \mathcal{B}_{ref} \quad , \quad \begin{cases} W_{TL}^{CGI}(x,y) = W_{TL}^{OTMC}(x,y) \\ W_{TR}^{CGI}(x,y) = W_{TR}^{OTMC}(x,y) \\ W_{BL}^{CGI}(x,y) = W_{BL}^{OTMC}(x,y) \\ W_{BR}^{CGI}(x,y) = W_{BR}^{OTMC}(x,y) \end{cases} \tag{A.14}$$

A.1.3 Small deformations: equivalence of the CGI and the OBMC

Following previous section, it is now proved that both CGI and OBMC perform equivalent *backward* motion compensation under several conditions:

1. $\forall (x,y) \in \mathbf{\Omega}$, the image domain, $I_{cur}(x,y)$ is infinitely derivable;
2. the deformation undergone by the reference block $\mathcal{B}_{ref} = \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{ref})$ is small enough;
3. the second derivatives of $I_{cur}(x,y)$ can be neglected in a local neighbourhood of P_{cur}.

The second condition requires quadrilateral $d\Omega_w(P_{ref})$, made of vertices $(P_{ref} + \overrightarrow{d_{TL}}(t_{cur}))$, $(P_{ref} + \overrightarrow{d_{TR}}(t_{cur}))$, $(P_{ref} + \overrightarrow{d_{BL}}(t_{cur}))$ and $(P_{ref} + \overrightarrow{d_{BR}}(t_{cur}))$, is fully included in the domain of convergence of the Taylor series of $I_{cur}(x,y)$ around P_{cur} and $d\Omega_w(P_{ref})$'s vertices (see figure A.1.2). Considering small deformations, it ensues that vectors $\overrightarrow{d_{TL}}(t_{cur})$, $\overrightarrow{d_{TR}}(t_{cur})$, $\overrightarrow{d_{BL}}(t_{cur})$, $\overrightarrow{d_{BR}}(t_{cur})$ and $\overrightarrow{d_{CGI}}(t_{cur})$ are close to each other, such that polygon $d\Omega_w(P_{ref})$ is very likely to be small enough to verify the condition. As for the third condition, it only requires $\left(\nabla\nabla^T\right)(I_{cur})(x,y)$ to be negligible over $d\Omega_w(P_{ref})$. In other words, it consists in approximating $I_{cur}(x,y)$ to an affine facet over $d\Omega_w(P_{ref})$.

In conclusion, the OTMCcan be seen as a low-computational version of the CGI: instead of performing the interpolation on a pixel-by-pixel basis, the OTMC computes once and for all the whole weighting coefficients which can then be systematically employed.

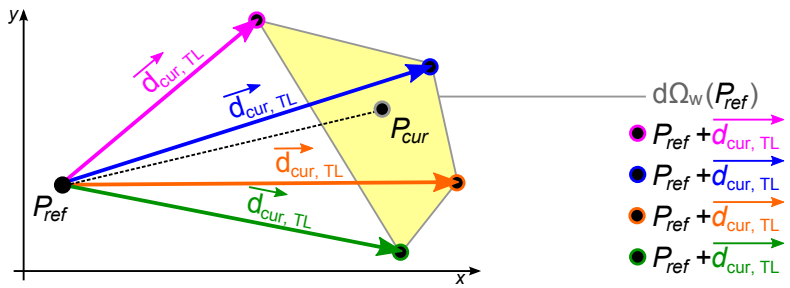


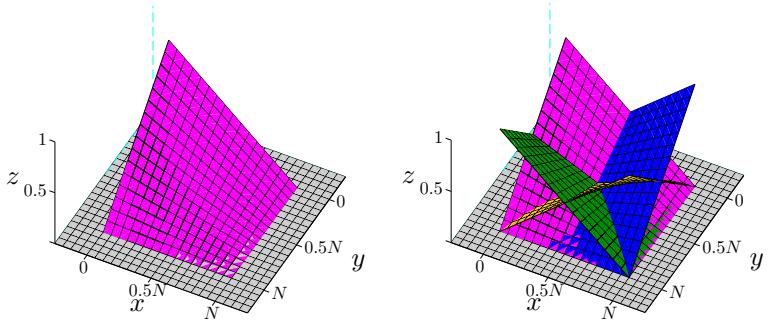
Figure A.1.2: *Motion compensation of P_{cur} into P_{ref}: quadrangular neighbourhood given by the motion vectors of the control points*

A.2 Forward motion compensation: a drift between OTMC and CGI coefficients

OTMC and CGI proved to be equivalent when motion compensating images in the *backward* direction from small deformations. To this end, the OTMC weighting windows and the influence functions of CGI's control points were matched over $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$ -see equation (A.14)-. In the *forward* direction, however, the OTMC does not provide a normalized projection as weighting windows do not properly overlap anymore. Furthermore, a few preliminary experiments showed that slight changes in the *forward* OTMC weighting coefficients may result in non negligible variations in synthesis quality.

A.2.1 Motion compensating CGI's interpolation functions and OTMC's weighting windows

Following previous demonstration, it is assumed that CGI's inner-mesh interpolation functions and OTMC's weighting windows are equivalent on the reference block $\mathcal{B}_{\text{ref}} = \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{ref}})$. In our experiments, a set of bilinear weights were used to balance the influence of each of the four corners of $\mathcal{B}_{\text{ref}}$. Figure A.2.1 shows corresponding weighting windows.



(a) *Top-left:* $W_{TL}^{CGI}(x,y) = W_{TL}^{OTMC}(x,y)$ **(b)** *Combined* backward *weighting windows*

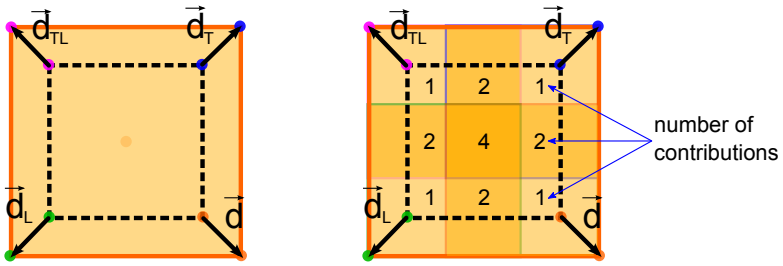
Figure A.2.1: Backward *motion compensation: equal OTMC's weighting windows and CGI's inner-mesh interpolation functions*

A.2.1.1 Enlarging the reference block: a simple deformation

Let first consider a simple deformation, by which the $N \times N$ reference block $\mathcal{B}_{\text{ref}}$ is enlarged into a $3/2N \times 3/2N$ block $\mathcal{B}_{\text{cur}} = \mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{cur}})$. The deformation is illustrated in figure A.2.2; it can be seen that $\mathbf{\Omega}_{\mathcal{M}_\mathcal{T}}(t_{\text{cur}})$ is similar whether CGI or OTMC motion model is used. It is now proposed to investigate the effects of such a deformation on the motion compensated weighting windows and inner-mesh interpolation functions.

A.2.1.2 Effects of the enlargement on the weighting coefficients

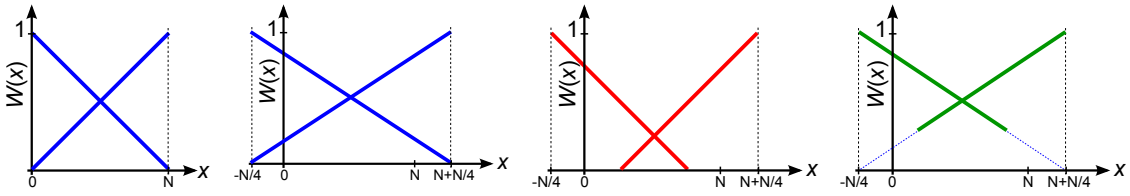
During the *forward* motion compensation, both OTMC weighting windows and CGI inner-mesh interpolation function are modified to account for the current deformation. While CGI accurately warps $I_{\text{ref}}(\mathcal{B}_{\text{ref}})$ onto $I_{\text{cur}}(\mathcal{B}_{\text{cur}})$, OTMC simply translates $I_{\text{ref}}(\mathcal{B}_{\text{ref}})$ on four different locations. In both cases, each motion compensated pixel from I_{cur} is balanced by the weighting coefficient of its collocated pixel in I_{ref}.



(a) *CGI motion compensation* **(b)** *OTMC motion compensation*

Figure A.2.2: *Enlargement of a block through CGI and OTMC approaches*

In other words, the *backward* weighting windows is motion compensated following the provided motion model. With CGI, *backward* inner-mesh interpolation functions are warped over $\mathcal{B}_{cur}$, such that motion compensated inner-mesh interpolation functions are smoothly stretched to match the shape of $\mathcal{B}_{cur}$. This is shown in figure A.2.3b. OTMC, on the other hand, does not deform its weighting windows, but simply translates it following the four motion vectors $\overrightarrow{d_{TL}}(t_{cur})$, $\overrightarrow{d_{TR}}(t_{cur})$, $\overrightarrow{d_{BL}}(t_{cur})$, $\overrightarrow{d_{BR}}(t_{cur})$. This is illustrated in figure A.2.3c.



(a) Backward *window* $W^{CGI}_{backward} = W^{OTMC}_{backward}$ **(b)** Forward *CGI window* $W^{CGI}_{forward}$ **(c)** Forward *OTMC window* $W^{OTMC}_{forward}$ **(d)** W^{OTMC}_{opt}*: a better OTMC window?*

Figure A.2.3: *Motion compensation of OTMC and CGI: sectional view of the weighting coefficients*

In addition, it was seen in chapter 5 that OTMC may not always synthesize a motion compensated pixel from four contributions, but only from the translations of $\mathcal{B}_{ref}$ available at the current pixel. With a simple enlargement, it can be seen from figure A.2.2b that $I_{cur}(\mathcal{B}_{cur})$ is synthesized from a variable number of contributions. Consequently, motion compensated weighting windows are normalized in various ways according to the available contributions.

A.3 Towards an optimal set of weights for the OTMC

It can be seen from chapter 5 that the deformations of the *motion tubes* were estimated in a *backward* fashion: the reconstruction of $I_{ref}(\mathcal{B}_{ref})$ from $I_{cur}(\mathcal{B}_{cur})$ was optimized. To this end, $I_{cur}(\mathcal{B}_{cur})$ was motion compensated backwards into $I_{ref}(\mathcal{B}_{ref})$. This only guarantee that estimated motion parameters are optimal with respect to provided *backward* weighting windows. Consequently, OTMC does not guarantee that a simple translation of each *backward* weighting window will provide optimal *forward* weighting coefficients.

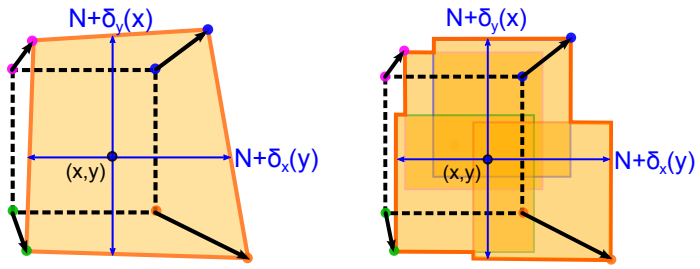
A.3.1 Applying CGI weights to OTMC contributions

As a first attempt towards the construction of an optimal set of *forward* weighting coefficients for the OTMC, it is now proposed to weight OTMC's contributions with CGI-like coefficients. Consequently, it is proposed to spread the OTMC weighting window over the whole domain of $\mathcal{B}_{cur}$, as illustrated in figure A.2.3d. Hence, for any point $P_{cur} = [x_{cur}, y_{cur}]$ of I_{cur} and its collocated position $P_{ref} = [x_{ref}, y_{ref}]$ in I_{ref}, the *forward* OTMC coefficients are now given by $W^{OTMC}_{opt}(x, y)$, such that

$$W^{OTMC}_{opt}(x_{cur}, y_{cur}) = W^{OTMC}_{backward}\left(\frac{N}{N+\delta_x(y_{ref})}\cdot x_{ref}, \frac{N}{N+\delta_y(x_{ref})}\cdot y_{ref}\right) \quad (A.15)$$

where $W^{\text{OTMC}}_{\text{backward}}(x, y)$ is one of the four *backward* weighting windows. As for $\delta_x(y)$ and $\delta_y(x)$, they are respectively the horizontal and vertical variations of the dimension of the deforming block (see figure A.3.1). Horizontal and vertical stretching dimensions are interpolated from those of the block's edges; an accurate approximation when deformations are small. They are given for any position (x, y) by

$$\begin{cases} \delta_x(y) = \dfrac{N-y}{N} \cdot \underbrace{(dx_{\text{cur, TR}} - dx_{\text{cur, TL}})}_{\text{top } x \text{ variation}} + \dfrac{y}{N} \cdot \underbrace{(dx_{\text{cur, BR}} - dx_{\text{cur, BL}})}_{\text{bottom } x \text{ variation}} \\ \delta_y(x) = \dfrac{N-x}{N} \cdot \underbrace{(dy_{\text{cur, BL}} - dy_{\text{cur, TL}})}_{\text{left } y \text{ variation}} + \dfrac{x}{N} \cdot \underbrace{(dy_{\text{cur, BR}} - dy_{\text{cur, TR}})}_{\text{right } y \text{ variation}} \end{cases} \tag{A.16}$$



(a) *CGI motion compensation* **(b)** *OTMC motion compensation*

Figure A.3.1: *Variations in dimensions of the deforming block*

In areas synthesized from all four OTMC contributions, we now have $W^{\text{OTMC}}_{\text{opt}}(x, y) \approx W^{\text{CGI}}_{\text{forward}}(x, y)$. As a consequence, the modified OTMC can now be identified to the CGI through an similar approach to what was demonstrated regarding the *backward* motion compensation in section A.1: the modified OTMC weights are now near-optimal from a CGI perspective. However, the compensation complexity is increased as $W^{\text{OTMC}}_{\text{opt}}$ needs to be computed for each pixel.

A.3.2 Practical experiments, conclusions, perspectives

In practice, however, the modified OTMC weighting windows do not provide any improvement. On the contrary, they lower the resulting PSNR on synthesized areas. Table A.3.1 shows the influence of the OTMC weighting windows on the synthesized images for several sequences. This can be due to numerous factors, including:

- initial hypothesis which allowed us to identify the OTMC to the CGI may not be true (too large deformations, lack of linearity of the images, ...);
- the provided modified OTMC weights are too crudely computed from the four motion vectors;

Sequence	SPSNR		Gain
	with initial OTMC weights	with modified OTMC weights	
Bus	24.08 dB	23.90 dB	-0.18 dB
City	31.60 dB	31.45 dB	-0.15 dB
Crew	31.80 dB	31.68 dB	-0.12 dB
Foreman	31.92 dB	31.77 dB	-0.15 dB
Mobile	25.62 dB	25.40 dB	-0.21 dB

Table A.3.1: *Influence of the OTMC weights on the PSNR*

As a perspective, however, deforming *motion tubes* may directly be motion compensated using a CGI motion model. CGI is truly more computationnaly demanding than the OTMC; however, most computations are due to the motion estimation process which performs a *backward* motion compensation for each set of motion parameters which needs to be evaluated. When synthesizing the images, a single motion compensation step is required. Consequently, it may be interesting to further investigate an non-symmetric scheme which relies on the OTMC at the estimation stage, and on the CGI at the synthesis stage.

Appendix B

Hybridizing TMC and OTMC motion modes: different scenarios

HYBRIDIZING THE DIFFERENT MOTION MODES provided in chapter 5 (in particular, in section 5.2.2.5), one may build a large variety of deformation patterns. Considering a simple group of four *motion tubes*, $4 \times 4 \times 4 \times 4 = 256$ different hybridization scenarios are conceivable. While section 5.2.2.5 already illustrated a few of them through figures 5.2.4, 5.2.5 and 5.2.6, this annex will further illustrate the variety of hybridization scenarios.

Again, the current *motion tube* is called **X**; its causal neighbours are respectively written **A**, **B** and **C** (respectively the top-left, top and left neighbours). Table 1 lists the different scenarios under consideration: forthcoming sections will illustrate each of them. The scenarios hybridize the four available motion and connection modes (disconnected TMC mode, and connected OTMC, top OTMC and left OTMC modes) in various ways. Obviously, the list is not exhaustive as only 16 out of the 64 possible scenarios are considered; however, the chosen scenarios are quite representative of the overall possibilities of the hybrid motion model.

Scenario index	***Motion tubes* deformation and connection modes**			
	A (top-left)	**B** (top-right)	**C** (bottom-left)	**X** (current)
Disconnected scenarios				
1	TMC	TMC	TMC	TMC
2	OTMC	OTMC	OTMC	TMC
3	TMC	Left OTMC	Top OTMC	TMC
4	TMC	TMC	Left OTMC	TMC
Full-connected scenarios				
5	TMC	TMC	TMC	OTMC
6	OTMC	OTMC	OTMC	OTMC
7	TMC	Left OTMC	Top OTMC	OTMC
8	TMC	TMC	Left OTMC	OTMC
Top-connected scenarios				
9	TMC	TMC	TMC	Top OTMC
10	OTMC	OTMC	OTMC	Top OTMC
11	TMC	Left OTMC	Top OTMC	Top OTMC
12	TMC	TMC	Left OTMC	Top OTMC
Left-connected scenarios				
13	TMC	TMC	TMC	Left OTMC
14	OTMC	OTMC	OTMC	Left OTMC
15	TMC	Left OTMC	Top OTMC	Left OTMC
16	TMC	TMC	Left OTMC	Left OTMC

Table 1: *A list of representative scenarios wherein the current* motion tube ***X*** *is connected to its neighbours in various ways*

B.1 Disconnected TMC scenarios

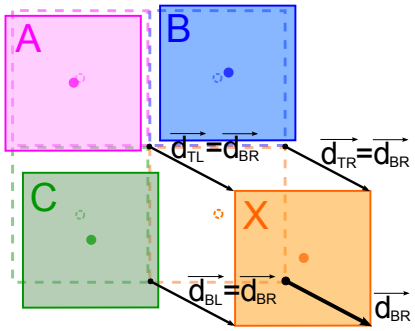


Figure B.1.1

Scenario 1
- *A, B, C and X: TMC*

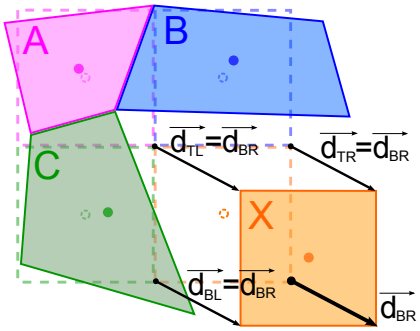


Figure B.1.2

Scenario 2
- *A, B and C: OTMC*
- *X: TMC*

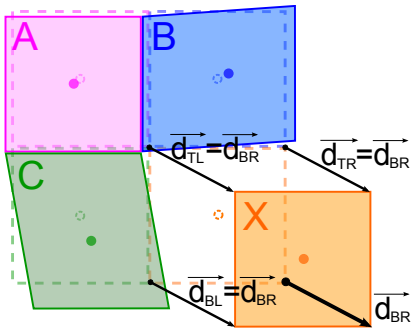


Figure B.1.3

Scenario 3
- *A and X: TMC*
- *B: left OTMC*
- *C: top OTMC*

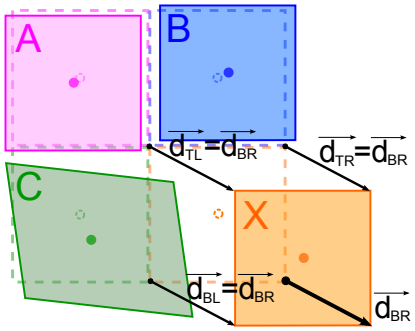
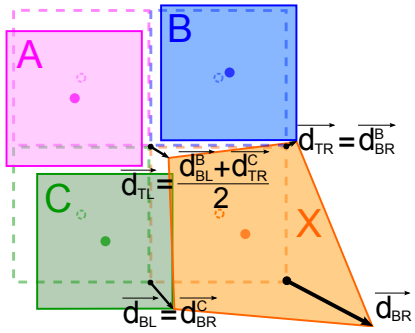


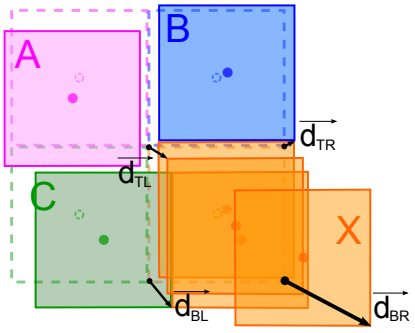
Figure B.1.4

Scenario 4
- *A, B and X: TMC*
- *C: left OTMC*

B.2 Full-connected scenarios



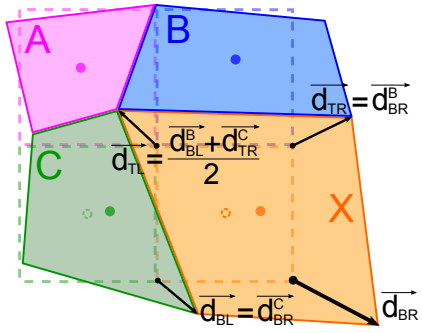
(a) *Idealistic deformation*



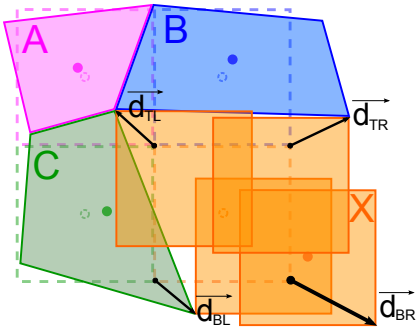
(b) *OBMC-like compensation*

Figure B.2.1

Scenario 5
- *A, B and C: TMC*
- *X: OTMC*



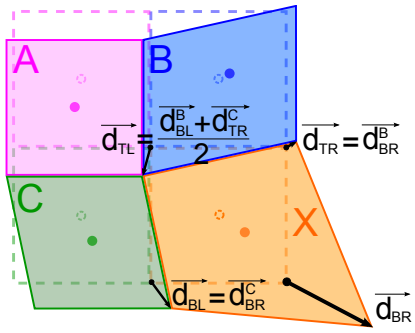
(a) *Idealistic deformation*



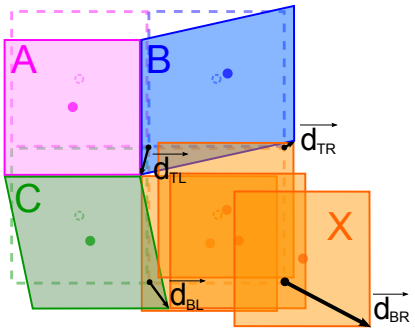
(b) *OBMC-like compensation*

Figure B.2.2

Scenario 6
- *A, B, C and X: OTMC*



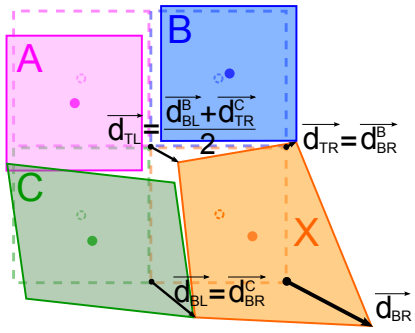
(a) *Idealistic deformation*



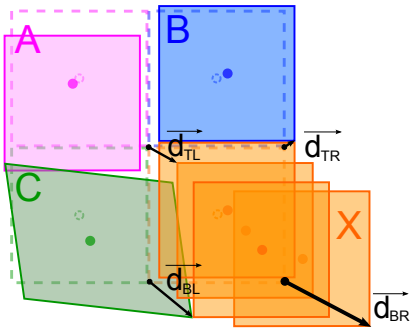
(b) *OBMC-like compensation*

Figure B.2.3

Scenario 7
- *A: TMC*
- *B: left OTMC*
- *C: top OTMC*
- *X: OTMC*



(a) *Idealistic deformation*

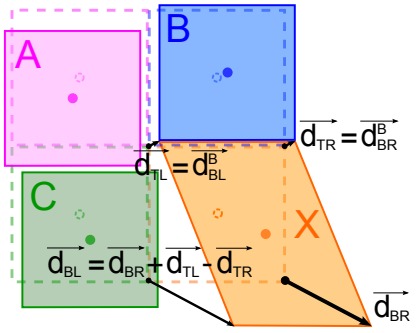


(b) *OBMC-like compensation*

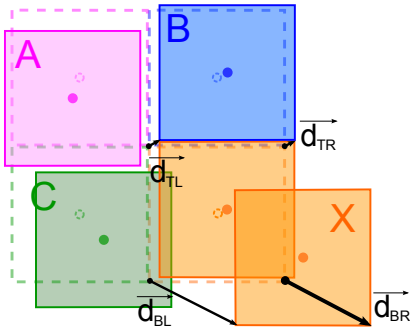
Figure B.2.4

Scenario 8
- *A and B: TMC*
- *C: left OTMC*
- *X: OTMC*

B.3 Top-connected scenarios



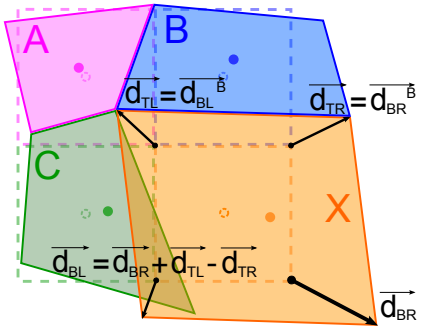
(a) *Idealistic deformation*



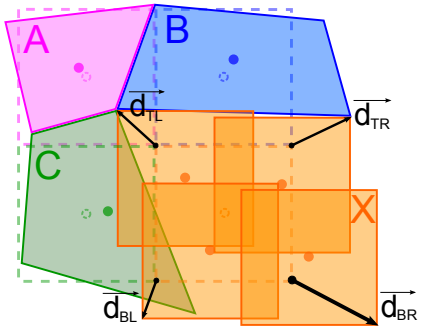
(b) *OBMC-like compensation*

Figure B.3.1

Scenario 9
- *A, B and C: TMC*
- *X: top OTMC*



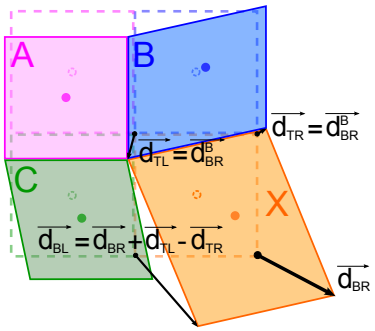
(a) *Idealistic deformation*



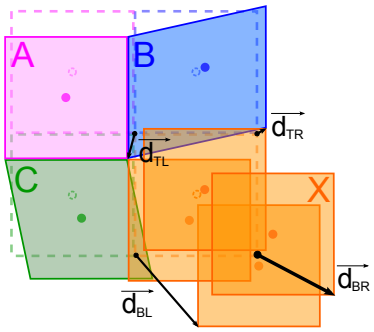
(b) *OBMC-like compensation*

Figure B.3.2

Scenario 10
- *A, B and C: OTMC*
- *X: top OTMC*



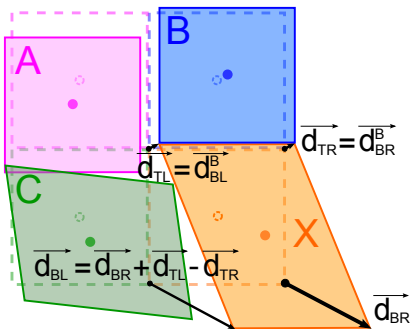
(a) *Idealistic deformation*



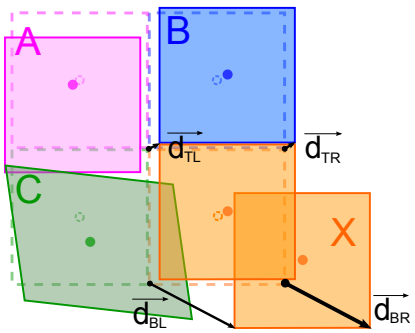
(b) *OBMC-like compensation*

Figure B.3.3

Scenario 11
- *A: TMC*
- *B: left OTMC*
- *C and X: top OTMC*



(a) *Idealistic deformation*

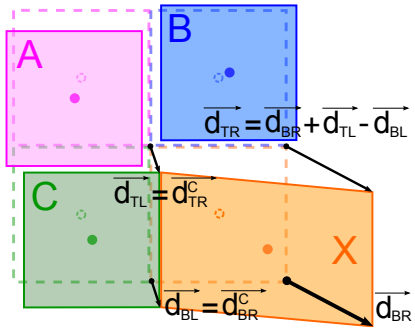


(b) *OBMC-like compensation*

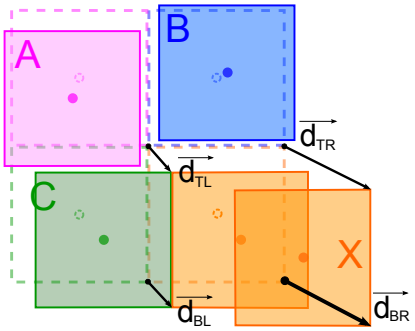
Figure B.3.4

Scenario 12
- *A and B: TMC*
- *C: left OTMC*
- *X: top OTMC*

B.4 Left-connected scenarios



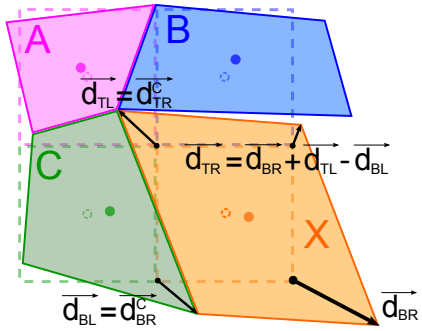
(a) *Idealistic deformation*



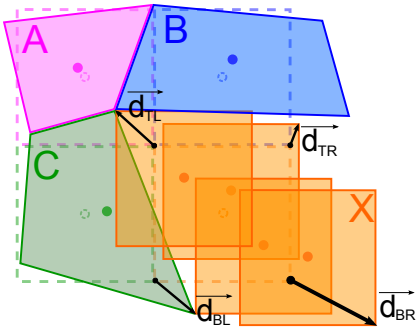
(b) *OBMC-like compensation*

Figure B.4.1

Scenario 13
- *A, B and C: TMC*
- *X: left OTMC*



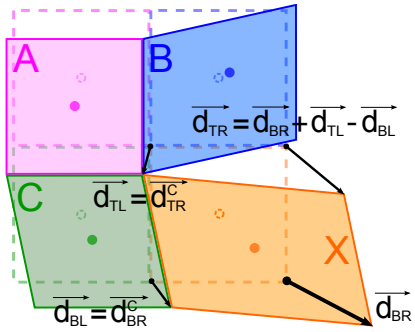
(a) *Idealistic deformation*



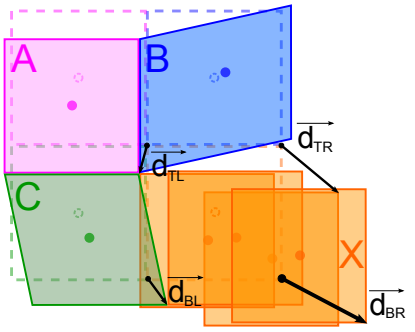
(b) *OBMC-like compensation*

Figure B.4.2

Scenario 14
- *A, B and C: OTMC*
- *X: left OTMC*



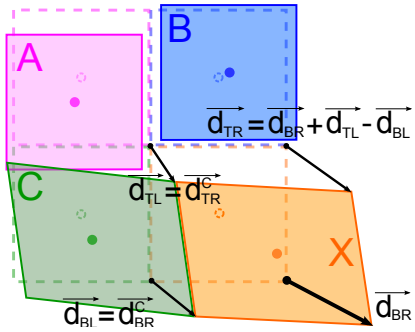
(a) *Idealistic deformation*



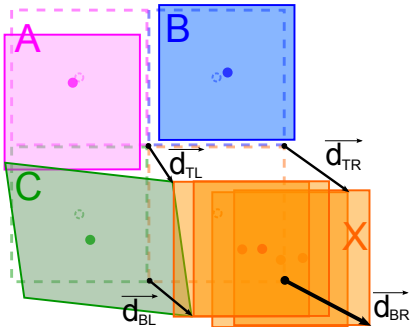
(b) *OBMC-like compensation*

Figure B.4.3

Scenario 15
- *A: TMC*
- *C: top OTMC*
- *B and X: left OTMC*



(a) *Idealistic deformation*



(b) *OBMC-like compensation*

Figure B.4.4

Scenario 16
- *A and B: TMC*
- *C and X: left OTMC*

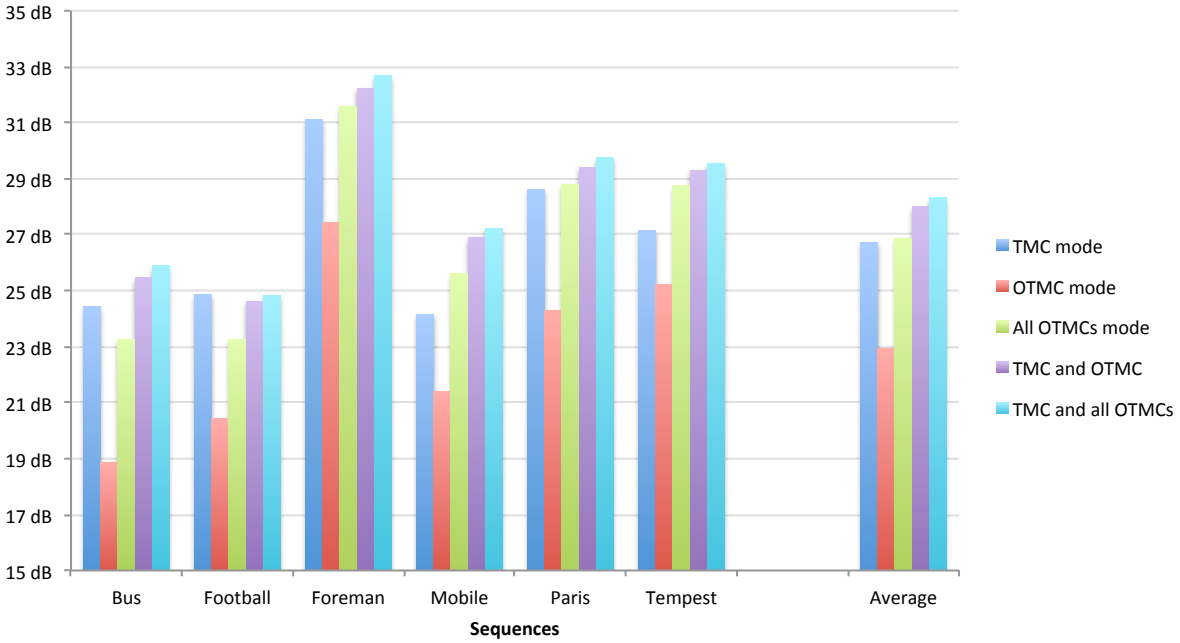
Appendix C

Motion model performances: detailed results

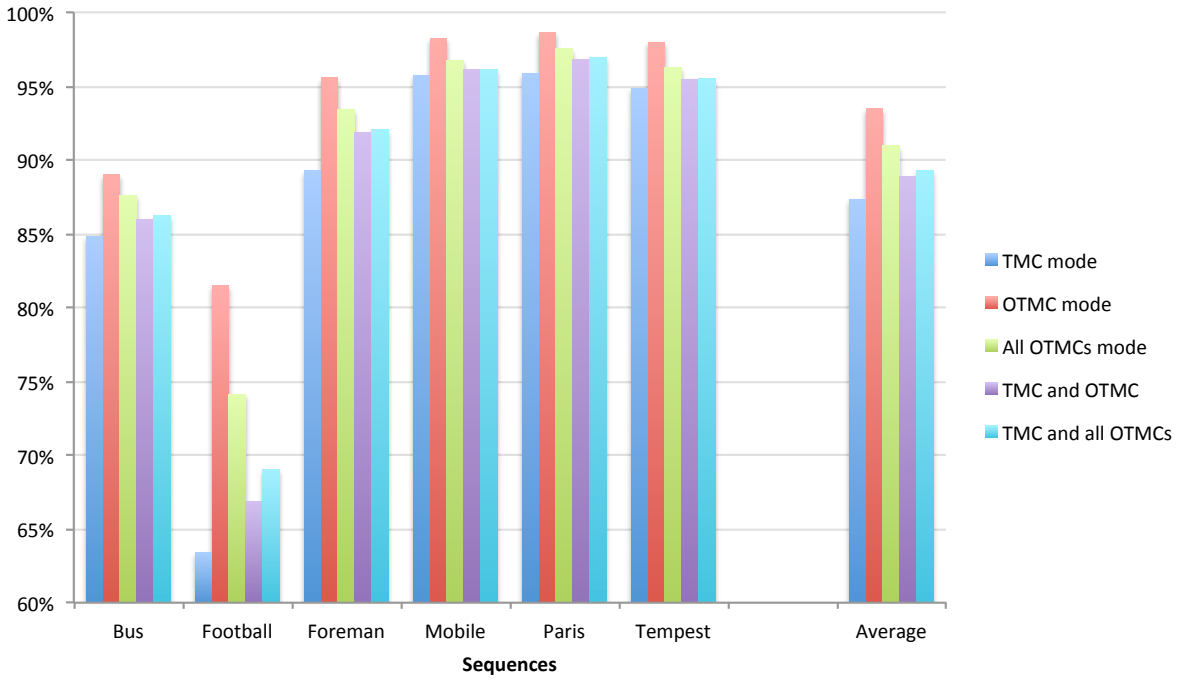
C.1 Hybridization of TMC, OTMC, left OTMC and top OTMC motion modes

Sequence	Motion model				PSNR I_{ref}	SPSNR I_{cur}	Rec. rate
	TMC	OTMC	Left OTMC	Top OTMC			
Bus	✓	×	×	×	23.33 dB	24.43 dB	84.84 %
	×	✓	×	×	19.30 dB	18.86 dB	89.01 %
	×	✓	✓	✓	23.09 dB	23.27 dB	87.57 %
	✓	✓	×	×	24.38 dB	25.49 dB	86.00 %
	✓	✓	✓	✓	24.82 dB	25.88 dB	86.24 %
Football	✓	×	×	×	24.15 dB	24.86 dB	63.38 %
	×	✓	×	×	21.38 dB	20.44 dB	81.49 %
	×	✓	✓	✓	23.91 dB	23.27 dB	74.14 %
	✓	✓	×	×	24.43 dB	24.63 dB	66.87 %
	✓	✓	✓	✓	24.82 dB	24.82 dB	69.03 %
Foreman	✓	×	×	×	30.47 dB	31.12 dB	89.27 %
	×	✓	×	×	27.84 dB	27.44 dB	95.56 %
	×	✓	✓	✓	31.58 dB	31.58 dB	93.43 %
	✓	✓	×	×	31.77 dB	32.22 dB	91.87 %
	✓	✓	✓	✓	32.34 dB	32.68 dB	92.05 %
Mobile	✓	×	×	×	23.84 dB	24.15 dB	95.70 %
	×	✓	×	×	21.70 dB	21.40 dB	98.24 %
	×	✓	✓	✓	25.57 dB	25.59 dB	96.75 %
	✓	✓	×	×	26.57 dB	26.92 dB	96.12 %
	✓	✓	✓	✓	26.87 dB	27.23 dB	96.12 %
Paris	✓	×	×	×	27.87 dB	28.63 dB	95.87 %
	×	✓	×	×	24.63 dB	24.30 dB	98.64 %
	×	✓	✓	✓	28.69 dB	28.80 dB	97.52 %
	✓	✓	×	×	28.95 dB	29.41 dB	96.79 %
	✓	✓	✓	✓	29.34 dB	29.75 dB	96.95 %
Tempest	✓	×	×	×	26.85 dB	27.13 dB	94.84 %
	×	✓	×	×	25.49 dB	25.23 dB	97.98 %
	×	✓	✓	✓	28.59 dB	28.74 dB	96.25 %
	✓	✓	×	×	28.95 dB	29.30 dB	95.48 %
	✓	✓	✓	✓	29.20 dB	29.54 dB	95.50 %

Table C.1.1: *Hybridization of the four motion modes: resulting SPSNRs and reconstruction rates - detailed results of table 5.3.1 from section 5.3.5*



(a) *Synthesized images reconstruction SPSNRs*



(b) *Synthesized images reconstruction rates*

Figure C.1.1: *Hybridization of the four motion modes: resulting SPSNRs and reconstruction rates - detailed results of figure 5.3.12 from section 5.3.5*

C.2 Influence of the LAOTMC on the synthesized images

Sequence	LAOTMC use	PSNR I_{ref}	SPSNR I_{cur}	Rec. rate
Bus	No LAOTMC split	23.85 dB	23.42 dB	92.37 %
	LAOTMC split down to 4 × 4	23.09 dB	23.27 dB	87.57 %
	LAOTMC split down to 2 × 2	22.92 dB	23.42 dB	85.74 %
Football	No LAOTMC split	24.03 dB	22.69 dB	82.02 %
	LAOTMC split down to 4 × 4	23.91 dB	23.27 dB	74.14 %
	LAOTMC split down to 2 × 2	23.73 dB	23.55 dB	69.17 %
Foreman	No LAOTMC split	31.00 dB	30.39 dB	96.65 %
	LAOTMC split down to 4 × 4	31.58 dB	31.58 dB	93.43 %
	LAOTMC split down to 2 × 2	31.49 dB	31.85 dB	91.97 %
Mobile	No LAOTMC split	25.60 dB	25.47 dB	97.88 %
	LAOTMC split down to 4 × 4	25.57 dB	25.59 dB	96.75 %
	LAOTMC split down to 2 × 2	25.52 dB	25.65 dB	96.46 %
Paris	No LAOTMC split	28.28 dB	28.00 dB	98.92 %
	LAOTMC split down to 4 × 4	28.69 dB	28.80 dB	97.52 %
	LAOTMC split down to 2 × 2	28.54 dB	28.88 dB	96.99 %
Tempest	No LAOTMC split	28.44 dB	27.99 dB	97.98 %
	LAOTMC split down to 4 × 4	28.59 dB	28.74 dB	96.25 %
	LAOTMC split down to 2 × 2	28.54 dB	28.77 dB	96.00 %

Table C.2.1: *LAOTMC: resulting SPSNRs and reconstruction rates – detailed results from section 5.3.5.2*

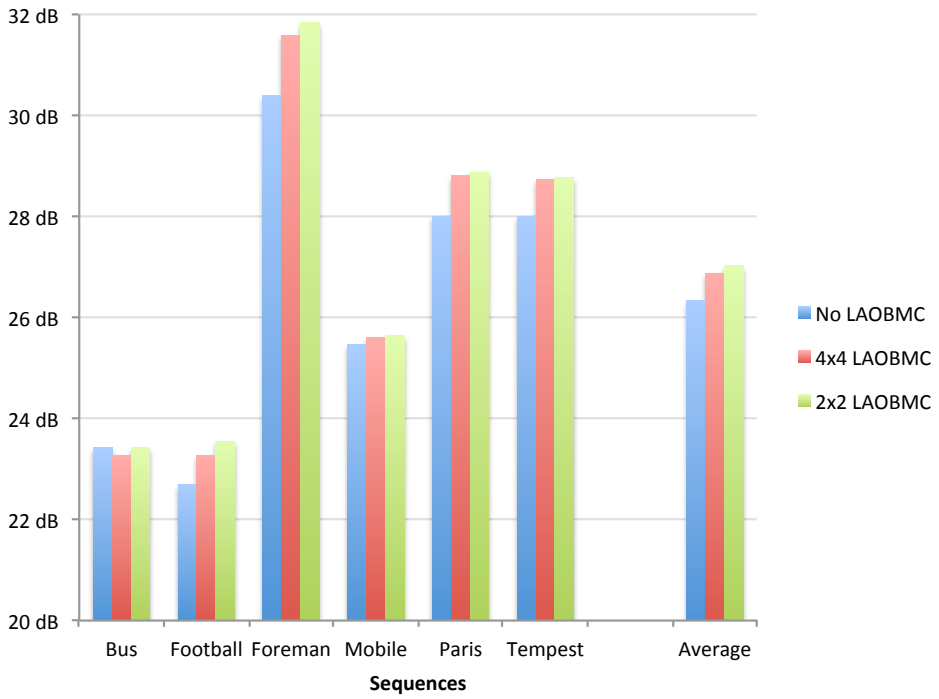


Figure C.2.1: *LAOTMC: influence on the reconstruction SPSNRs*

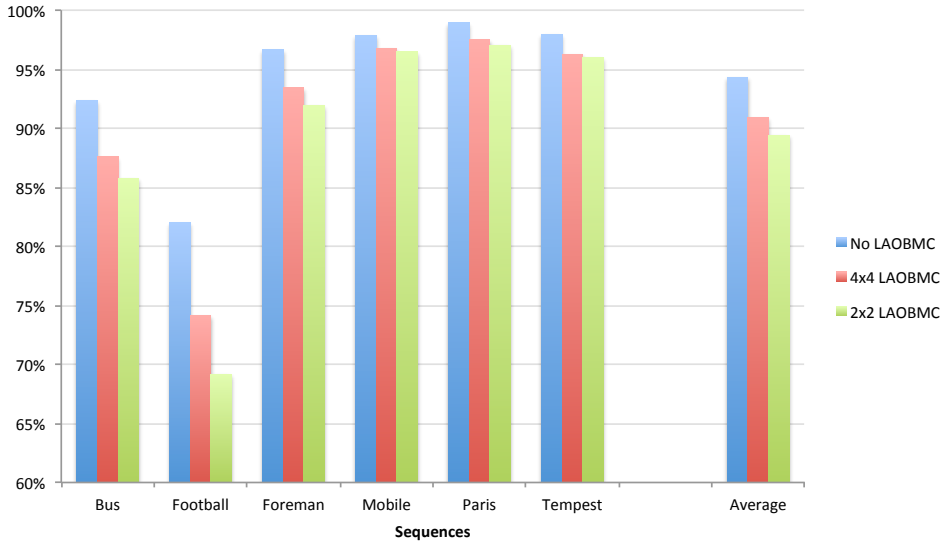


Figure C.2.2: *LAOTMC: influence on the reconstruction rates*

C.3 Influence of the regularization on the synthesized images

C.3.1 Multigrid versus rate-distortion regularization

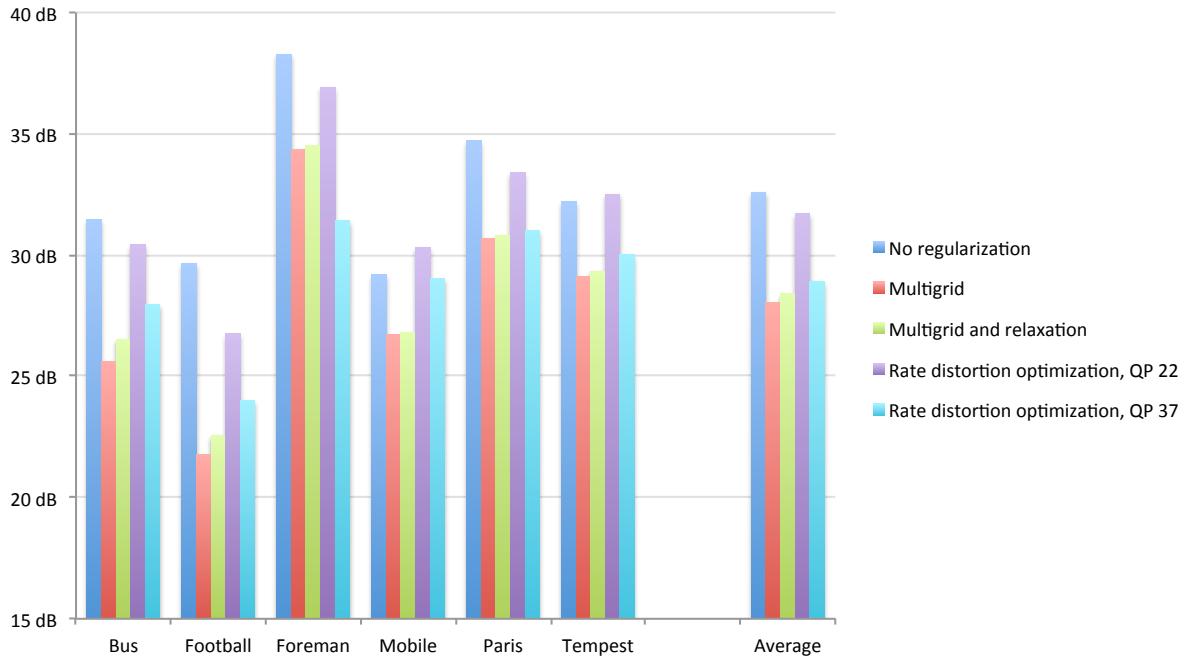


Figure C.3.1: *Influence of the regularization on the reconstruction SPSNRs for various sequences - detailed results of section 5.4.3.3*

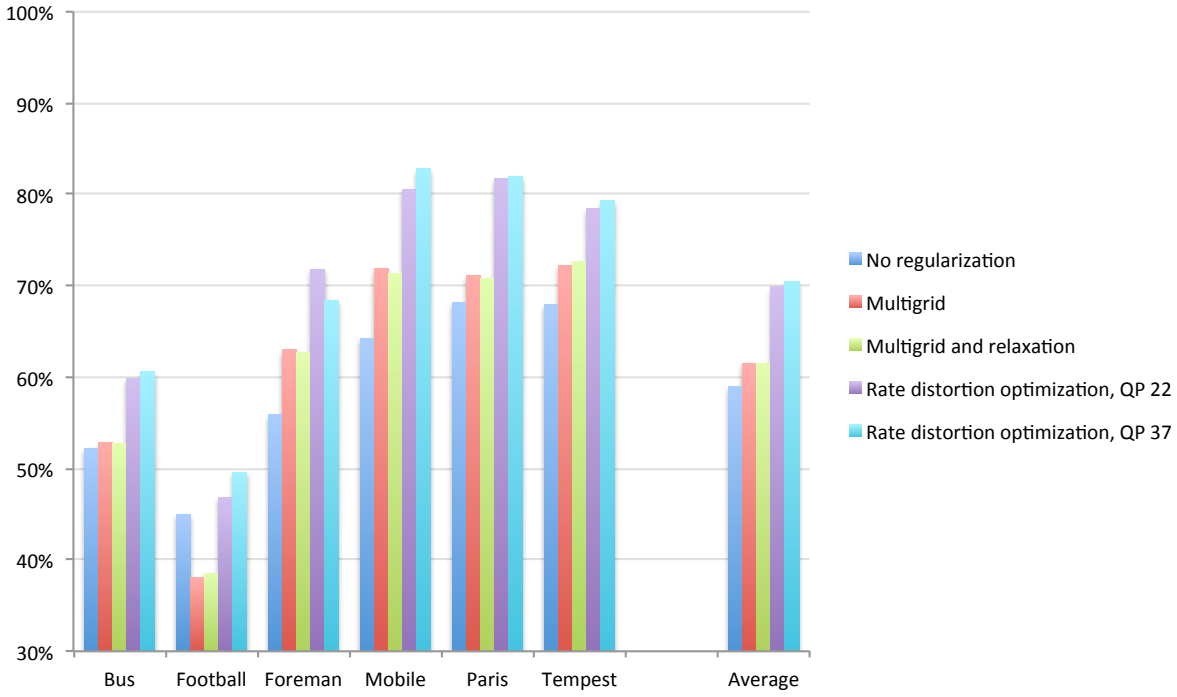


Figure C.3.2: *Influence of the regularization on the reconstruction rates for various sequences - detailed results of section 5.4.3.3*

C.3.2 Rate-distortion regularization: QP and motion bitrates

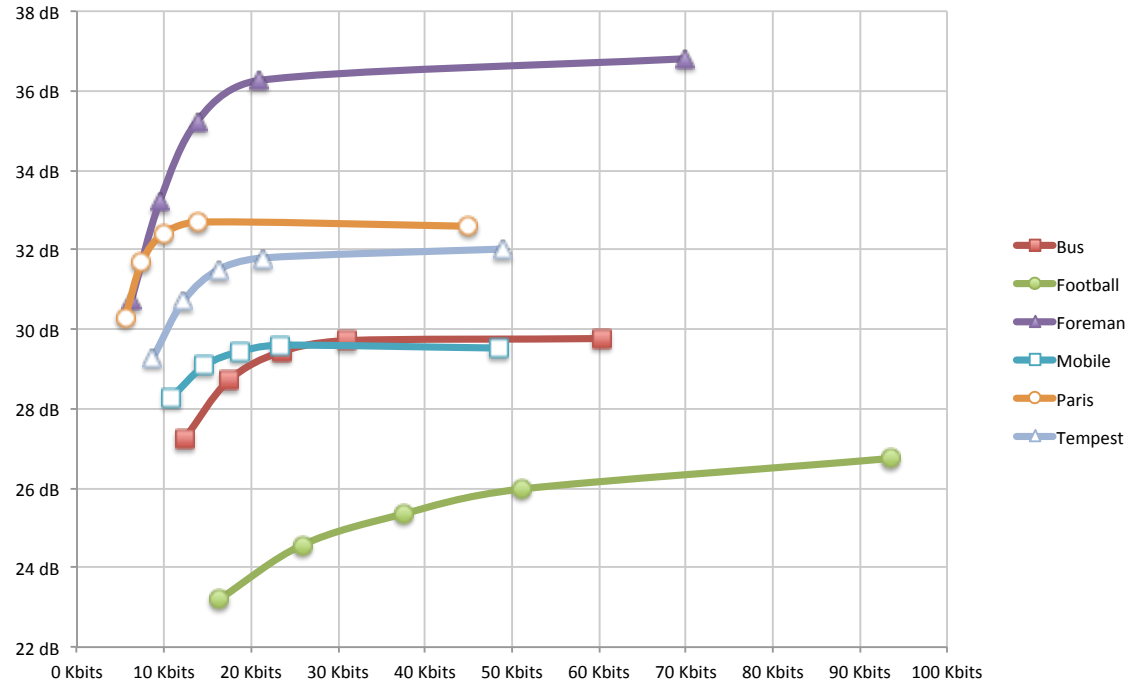


Figure C.3.3: *Rate-distortion regularization: influence on the reconstruction SPSNRs for various sequences - detailed results of section 5.4.3.3. For each curve, the R-D points correspond, from right to left, to: no RDO, then RDO at successive QPs 22, 27, 32 and 37.*

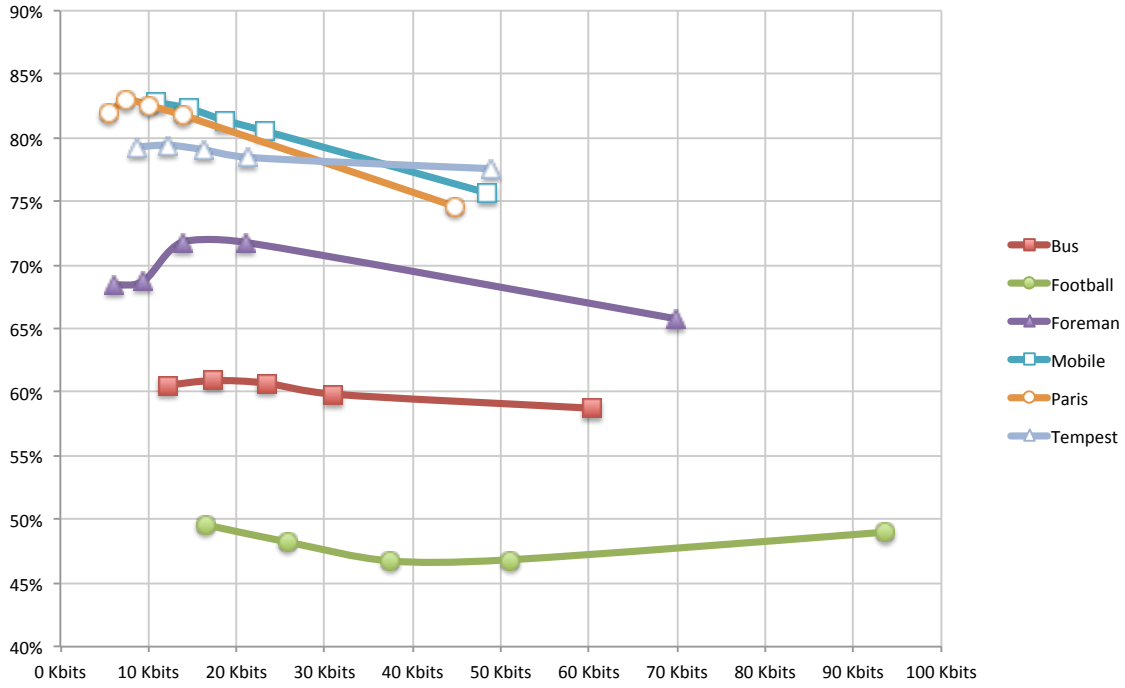


Figure C.3.4: *Rate-distortion regularization: influence on the reconstruction rates for various sequences - detailed results of section 5.4.3.3. For each curve, the R-D points correspond, from right to left, to: no RDO, then RDO at successive QPs 22, 27, 32 and 37.*

C.4 Variable-sized motion tubes: rate-distortion performances

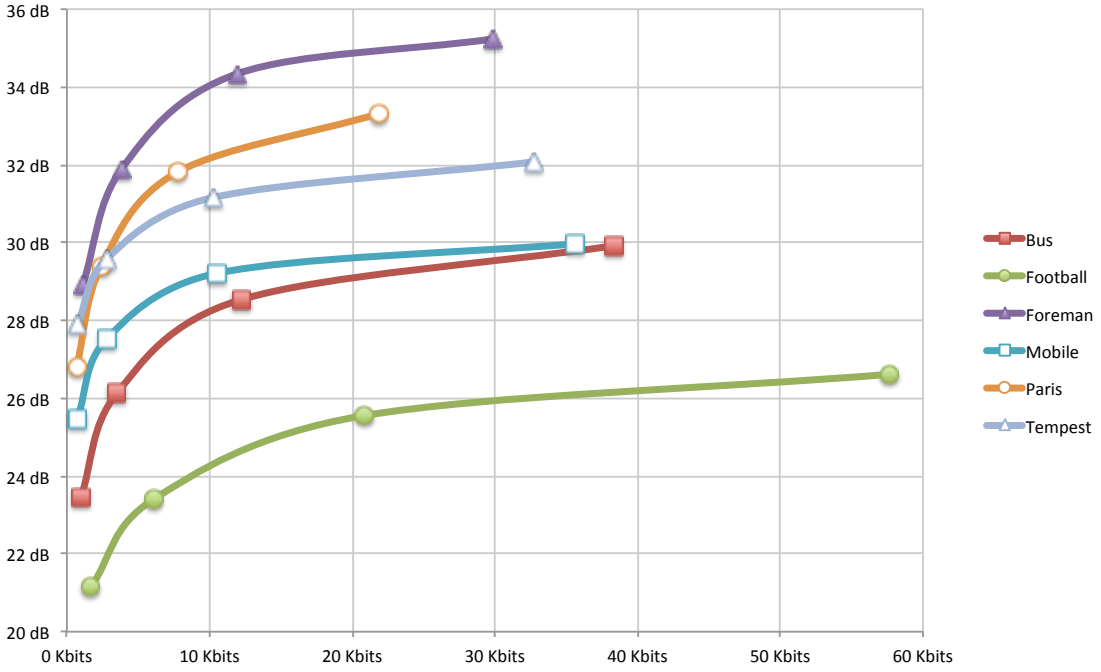


Figure C.4.1: *Influence of the size of the* motion tubes*: resulting SPSNRs for various sequences - detailed results of section 5.6.3. For each curve, the R-D points correspond, from right to left, to:* 32×32, 16×16, 8×8 *and* 4×4 motion tubes.

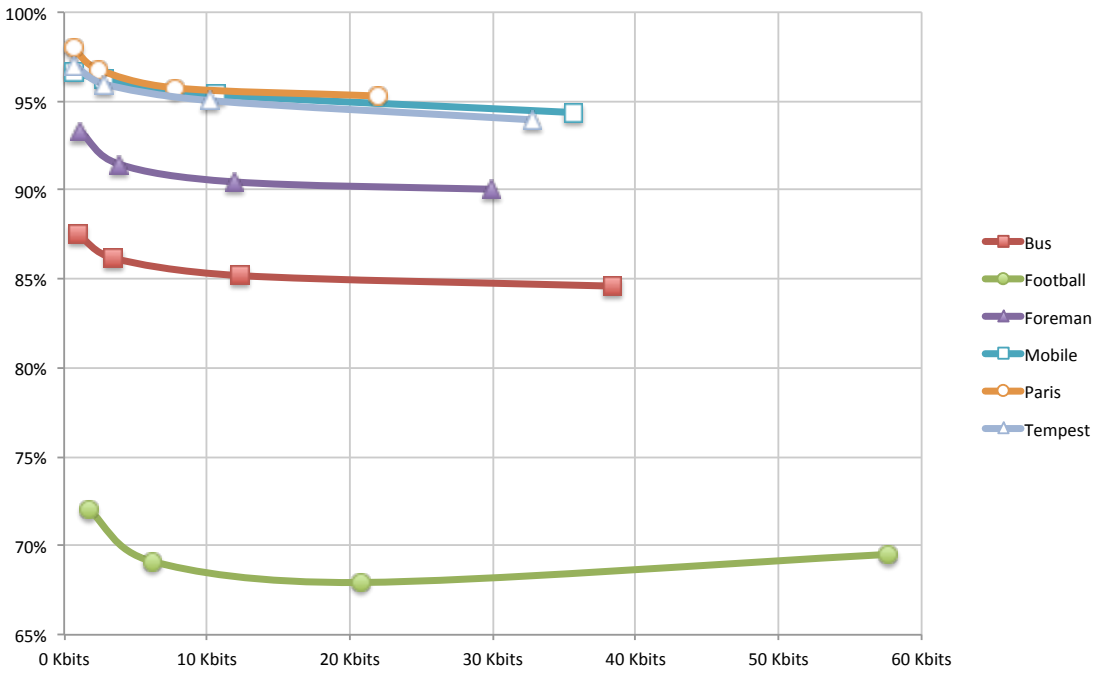


Figure C.4.2: *Influence of the size of the* motion tubes*: resulting reconstruction rate for various sequences – detailed results of section 5.6.3. For each curve, the R-D points correspond, from right to left, to:* 32×32, 16×16, 8×8 *and* 4×4 motion tubes.

Appendix D

Texture synthesis improvements: detailed results

D.1 Bi-predictive motion tubes: detailed performances

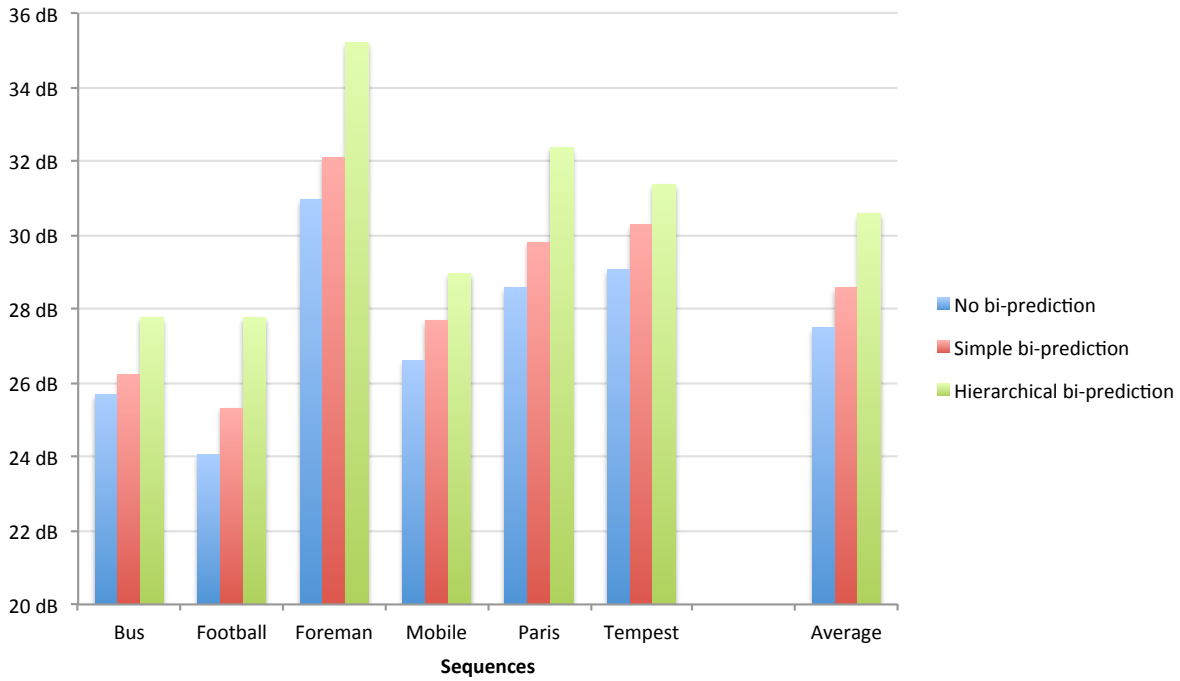


Figure D.1.1: *Influence of the* B-tubes *bi-prediction on the PSNR of reconstructed areas - detailed results of section 6.2.3*

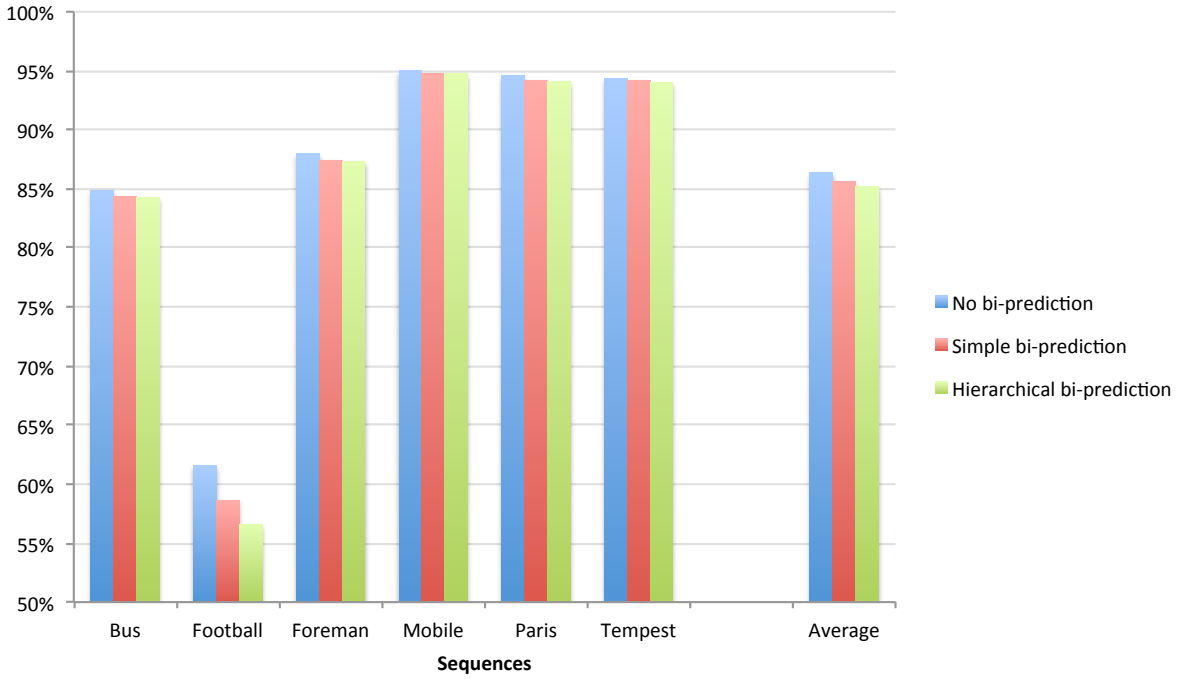


Figure D.1.2: *Influence of the* B-tubes *bi-prediction on the reconstruction rate - detailed results of section 6.2.3*

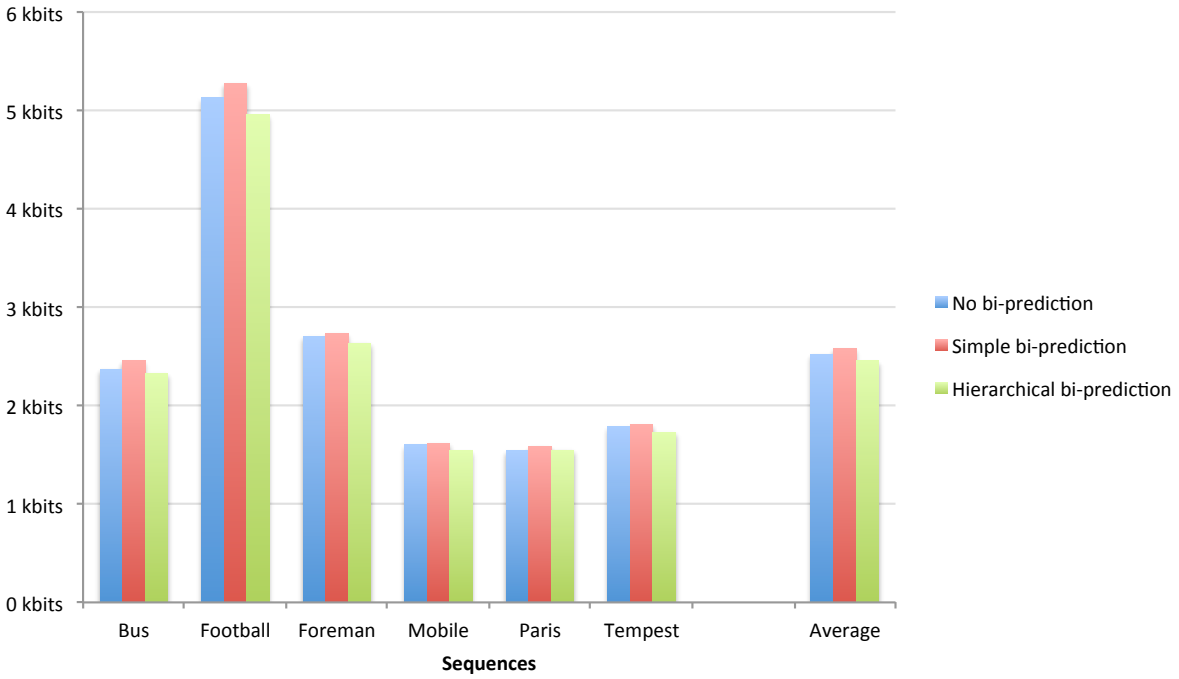


Figure D.1.3: *Influence of the* B-tubes *bi-prediction on the motion bitrate - detailed results of section 6.2.3*

D.2 Inpainting: detailed performances

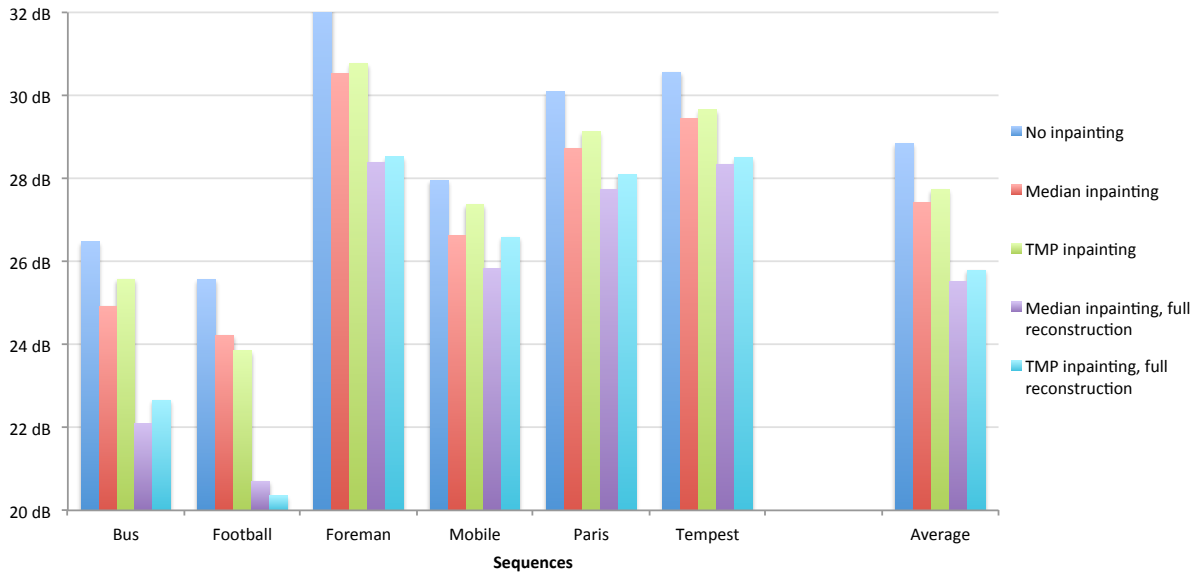


Figure D.2.1: *Influence of inpainting on the PSNR of reconstructed areas – detailed results of section 6.4.2.4.*

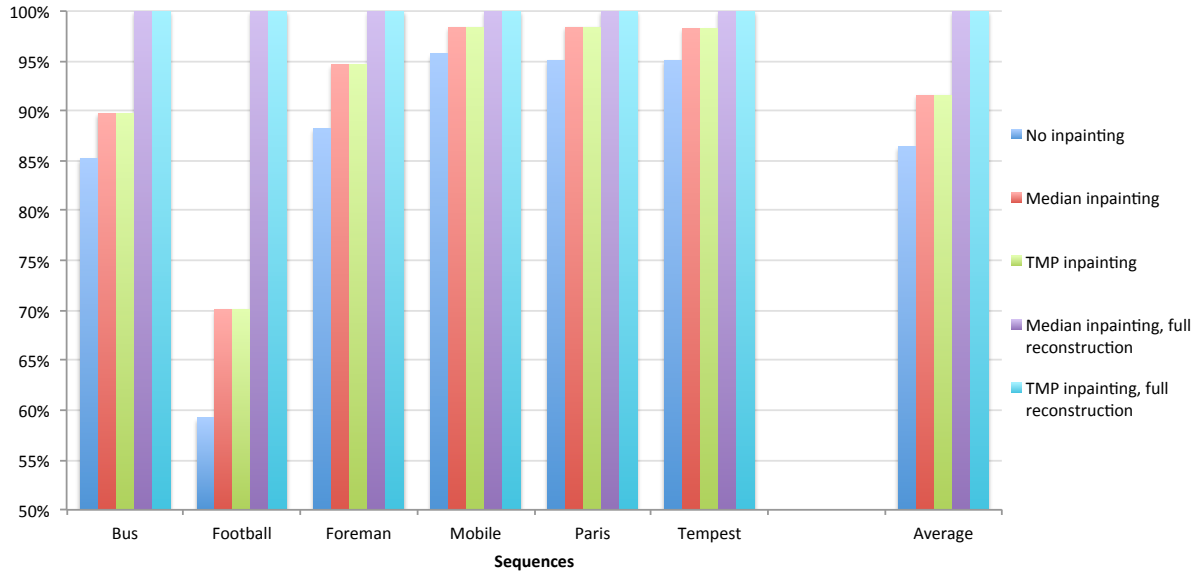


Figure D.2.2: *Influence of the inpainting on the reconstruction rate – detailed results of section 6.4.2.4.*

Appendix E

Résumé

Dans un monde toujours plus connecté, où l'on parle même d'*hyper-connectivité*, de plus en plus d'informations sont échangées. En effet, un nombre toujours croissant d'utilisateurs a maintenant accès à de nombreux services qui nécessitent la transmission de données à travers un réseau de communication. Au delà de cet aspect purement démographique, la multitude de terminaux communicants aujourd'hui disponibles, ainsi que la disparité des systèmes de communication sont également responsables d'un accroissement spectaculaire de la quantité de données échangées. A titre d'exemple, on peut citer le formidable essort des téléphones mobiles qui permettent désormais d'accéder à internet, et ainsi d'échanger des volumes de données importants, via les nouveaux standards de communications mobiles (3G, 4G).

Les contenus vidéo, en particulier, constituent la majeure partie du volume de données échangées. En effet, les contenus et les usages, eux aussi, évoluent. De plus en plus diversifiés (WebTV, VOD, P2P), les contenus audiovisuels sont toujours plus accessibles et génèrent donc un traffic croissant. Par ailleurs, leur qualité s'améliore elle aussi, à travers l'arrivée récente de la haute-définition (HD), ainsi que de la 3D qui perce depuis peu également. Ces améliorations ont un impact direct sur le traffic réseau, puisque les contenus associés sont beaucoup plus volumineux.

En conséquence, une augmentation considérable du traffic réseau est à prévoir. Selon CISCO, le traffic réseau mondial devrait tripler d'ici 2014 [CIS10]. Quant à la vidéo, si elle n'occupe que 40% du traffic à l'heure actuelle, on estime qu'elle représentera 90% de celui-ci en 2014 [CIS10]. Malgré les progrès constants en matière de transmission, on estime que les infrastructures réseaux ne pourront probablement pas supporter une telle charge.

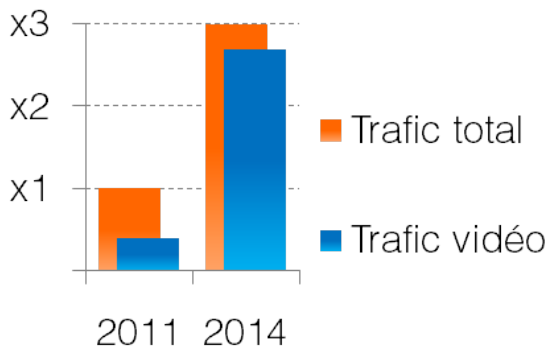


Figure 1: *2011 - 2014 : Evolution du traffic réseau et de la proportion représentée par les données vidéo [CIS10]*

A ce titre, il est plus que jamais capital de travailler au développement de techniques qui permettront de réduire au maximum la quantité d'informations nécessaires à la transmission de données vidéo. En d'autres termes, la compression vidéo est encore et toujours d'actualité. D'autre part, devant la multiplication des contenus, notre capacité à les analyser, les décrire et les indexer va s'avérer cruciale. C'est dans ce contexte que les travaux présentés dans cette thèse se situent. En effet, notre objectif est de construire une nouvelle manière de représenter les vidéos,

- en vue **d'améliorer** les **performances de compression** par rapport aux approches existantes,
- tout en garantissant un **niveau sémantique** aussi **élevé** que possible, facilitant ainsi d'éventuelles étapes d'analyse, de description et d'indexation.

Cette annexe vise à fournir un résumé du contenu anglais disponible dans le document. Elle s'articule en sept sections :

la section E.1 propose un bref état de l'art qui se concentre sur les différentes approches qui ont été employées dans le cadre de la compression vidéo,

la section E.2 introduit le concept de la représentation proposée, à travers l'utilisation de "tubes de mouvement",

la section E.3 s'intéresse au modèle de mouvement, adapté aux tubes, que nous avons proposé,

la section E.4 se concentre sur diverses approches visant à maximiser la cohérence spatio-temporelle de la représentation, à travers une structuration systématique et l'emploi de techniques de régularisation,

la section E.5 introduit la méthodologie de validation que nous avons suivie, à travers l'intégration des tubes au sein d'un codeur standardisé, ITU-T H.264/AVC,

la section E.6 s'intéresse au problème de la vie et de la mort des tubes et propose une approche préliminaire permettant d'optimiser leur utilisation pour garantir un maximum de compacité,

la section E.7 conclut ce résumé et introduit brièvement quelques unes des nombreuses perspectives que nos travaux proposent.

E.1 La compression vidéo : approches existantes

E.1.1 La chaîne de compression

La figure E.1.1 illustre certaines opérations de base réalisées par la majorité des codeurs vidéo. Si le mode de partitionnement de l'information dépend principalement de la représentation choisie, un certain nombre d'opérations sont généralement communes à la plupart des codeurs:

- **l'estimation et la compensation en mouvement** permet d'aligner les différents éléments (pixels, blocs ou objets) le long de l'axe temporel;
- **la transformation** permet d'exprimer un ensemble de valeurs à transmettre dans un espace où elles sont naturellement plus compactes –à noter que le terme *transformation* est à prendre au sens large du terme, et peut désigner une transformation mathématique ou une quelconque étape qui modifie la représentation des valeurs de manière à les compacter–;
- la **quantification** permet de réduire la dynamique des coefficients à transmettre ; non réversible, cette opération introduit des pertes et des distorsions, mais permet de contrôler le débit final;
- **le codage entropique** permet de compacter l'information binaire obtenue en s'appuyant sur les propriété statistiques de celle-ci.

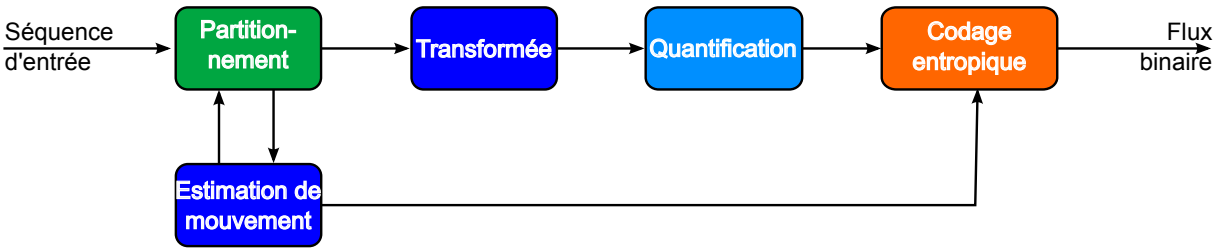


Figure E.1.1: *Une chaîne de compression typique*

E.1.2 Représentation et niveau sémantique

Avant même de sélectionner les outils de codage qui permettront de compresser l'information vidéo, il est crucial de choisir avec soin la manière dont on veut la représenter. Celle-ci peut être considérée comme un simple amas de pixel, ou à l'inverse comme la projection d'une scène 3D et donc composée d'objets sémantiques. Le choix de la représentation dirigera ensuite le choix des outils de codage appropriés. La complexité de la solution proposée est étroitement liée à la représentation choisie : si la sémantique est un atout déterminant en vue de l'analyse, elle est aussi relativement

complexe à construire. La figure E.1.2 illustre ce concept en ordonnant les principales représentation suivant leur niveau sémantique.

Figure E.1.2: *Echelle des représentations*

E.1.2.1 Représentations photométriques

Les représentations photométriques considèrent les séquences d'images comme un simple amas de pixel, et n'associent pas de sémantique particulière à leur contenu. En conséquence, un partitionnement en blocs réguliers [Ric03, WS03] ou régions arbitraires [PL99] est souvent utilisé.

Quel que soit le partitionnement utilisé, ces approches n'optimisent les décisions de codage que sur critères débit-distortion, et ce de façon objective. Du fait de sa simplicité d'utilisation, la représentation photométrique constitue aujourd'hui l'approche classique.

E.1.2.2 Représentations sémantiques

A l'opposé, les représentations sémantiques considèrent les vidéos comme étant la projection sur le plan de la caméra d'un ensemble d'objets en mouvement dans une scène. A l'instar de cette interprétation, les séquences d'images sont partitionnées en régions sémantiques (objets ou parties d'objet) dont les déplacements et les déformations éventuels sont suivis au cours du temps. On parle alors d'approches basées régions ou basées modèles [YW94, SGPK94, SMP+97, SM99, XLLZ01]. Parfois, la sémantique est introduite via une étape de codage supplémentaire, en aval d'un codage photométrique (ex: MPEG-4 Part 2 [ISO00a]).

Les objets étant ensuite représentés par une ou plusieurs régions, ils peuvent être traités indépendamment les uns des autres et décrits à l'aide de modèles 2D ou 3D. Une fois ces objets transmis au décodeur, ce dernier peut alors reconstituer la scène. En pratique, cependant, la viabilité de ces codeurs est difficile à justifier :

- une fois paramétrisés, l'information nécessaire à la transmission de ces objets est parfois prohibitive, en particulier à bas débits ;
- l'étape de segmentation nécessaire à l'identification des objets ou des régions est souvent trop complexe en vue d'applications temps-réel.

E.1.2.3 Approches pseudo-sémantiques

A mi-chemin entre l'approche photométrique et l'approche sémantique, les approches pseudo-sémantiques offrent un compromis sémantique/complexité. Si le codage s'effectue toujours au niveau pixel, une étape préliminaire de partitionnement permet d'intégrer un certain degré de sémantique à la représentation, ce qui est susceptible d'exhiber un maximum de corrélation et de cohérence, tant spatiales que temporelles.

A l'instar des approches sémantiques, les approches pseudo-sémantiques ne permettent généralement pas de décrire la scène de manière parfaite ; elles se content de construire une représentation visuellement acceptable. A titre d'exemple, on pensera aux approches par analyse-synthèse (cf section suivante) et à certains codeurs régions [PL99, DBBR07] qui ne segmentent pas la vidéo sur critères sémantiques.

E.1.3 Approches classiques et approches en rupture

E.1.3.1 Approche classique

Si de nombreuses solutions de codage ont été proposées, les standards de compression se succèdent et se ressemblent : depuis les premières versions du standard MPEG, la représentation n'a pas ou peu évoluée et reste photométrique. Seuls les outils de codage évoluent et offrent toujours plus de performances. HEVC, le successeur du standard actuel (H.264/AVC) ne déroge pas à la règle et ne remet pas en cause cette approche.

Ainsi, l'axe temporel est décrit de manière discontinue : la vidéo est découpée en images qui sont traitées une à une. Dans le domaine spatial, le contenu est décrit de manière locale (ou discontinue) : chaque image est découpée en *macroblocs*. Les performances de compression obtenues sont très élevées : le découpage des images en blocs permet d'effectuer une optimisation débit-distorsion au niveau local. D'autre part, cette approche est relativement facile à implanter sur de nombreux systèmes.

E.1.3.2 Approches en rupture

Les codeurs régions [SM99, XLLZ01, FBDC07], déjà évoqués précédemment, sont certes idéaux en termes de sémantique, mais sont difficiles à mettre en œuvre en pratique. Ils ne seront pas abordés plus en détails.

Les codeurs par ondelette 3D [Ohm94, CW99, ST01, PPB01] effectuent une transformée en ondelette le long de la trajectoire du mouvement, et une transformée dans le domaine spatial. Cette solution exhibe donc la continuité des informations, et parfois leur scalabilité, tant spatiales que temporelles [XXLZ01, XWX+04].

Les codeurs par analyse-synthèse [TZ94, WXCM99, Cam04, LG08], enfin, cherchent à décorréler la texture et le mouvement. Hautement flexible, cette approche pose un nouveau problème : comment répartir le débit entre le mouvement et la texture ?
Certains outils sont représentatifs de ces codeurs : la compensation en mouvement globale [YW94, PMCM01, Cam04], par exemple, consiste à projeter toutes les images d'un groupe d'images sur une seule image de référence. Elle nécessite l'utilisation d'un modèle de mouvement avancé (par opposition à celui proposé par l'approche classique, i.e. purement translationnel). Plus récemment, des outils de synthèse de texture ont été proposés [DH04, ZSWL07a, NNHW07, MADB10] ; à l'heure actuelle ces approches souffrent de leur complexité, ne conviennent pas à tous les types de textures, et surtout sont difficiles à mettre en œuvre le long de l'axe temporel.

E.1.3.3 Discussion

A la lumière de cette revue, un schéma alternatif se dessine : construire une représentation à mi-chemin entre l'approche classique (photométrique) et certaines des approches en rupture dont l'intérêt à été prouvé. C'est donc vers un schéma pseudo-sémantique que nous nous dirigeons : une description locale (discontinue) du contenu spatial, à l'image de l'approche classique, couplée à une description continue de son évolution temporelle (à l'image des approches par analyse-synthèse). En terme de compression, enfin, nous tâcherons de rester aussi proche que possible des codeurs standardisés, à travers un partitionnement en blocs.

E.2 Le tube de mouvement : concept

E.2.1 Persistance temporelle des texture

Lorsque l'on observe une séquence d'images naturelles, il est facile d'observer à quel point les images successives se ressemblent : une même zone de texture est généralement présente dans plusieurs images consécutives. En effet, l'information de texture est portée par le fond et les objets de la scène ; ceux-ci persistent au cours du temps. Cependant, les mouvements de la caméra ou ceux des objets de la scène se traduisent par des déplacement ou des déformations de texture le long de l'axe temporel. Ce phénomène est illustré dans la figure E.2.1 : le patch vert subit une simple translation, tandis que le patch bleu subit une translation ainsi qu'une rotation dues au mouvement du ballon.

Ainsi, une séquence d'images peut être, dans une certaine mesure, considérée comme un ensemble de patchs de textures, qui se déplacent et se déforment le long de trajectoires temporelles. En supposant qu'il soit possible de déterminer cet ensemble de patchs et leurs trajectoires, on peut alors représenter les séquences d'images par cet ensemble, et plus par une simple succession d'images. Il est important de noter, toutefois, que toutes les textures ne peuvent pas être

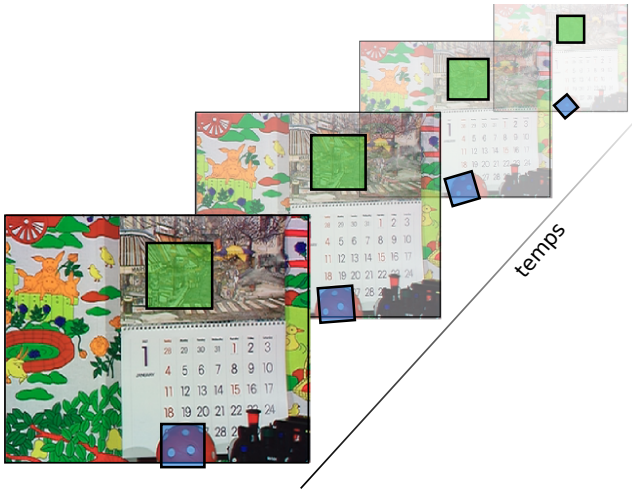
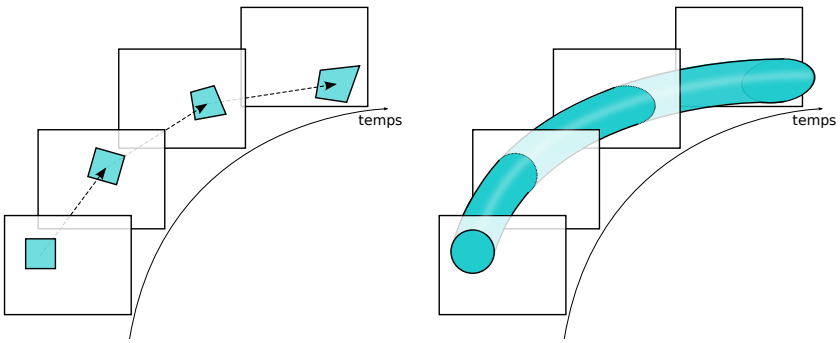


Figure E.2.1: *Persistance temporelle des texture dans une séquence d'images*

représentées dans un tel cadre : les objets particulaires, les liquides et les effets de transparence, entre autres, sont impossibles à suivre au cours du temps simplement via déformation et déplacement d'une texture initiale. En conséquence, nos travaux se concentrent sur des séquences où de tels effets sont absents.

E.2.2 Du patch de texture au tube

Comme on peut le voir sur la figure E.2.2, la succession des formes prises par un patch de texture rappelle un tube dont la section se déforme au cours du temps. Pour cette raison, il a été choisi de baptiser ces patchs de textures qui sont suivis des *tubes de mouvement*. On notera que ce concept a déjà été proposé dans de précédentes études [PBC07, BDRB07], dans lesquelles il était utilisé à des fins d'analyse, et non pas de représentation comme c'est le cas pour nos travaux.



(a) *Un patch de texture en déplacement*

(b) *Représentation simpliste d'un tube de mouvement*

Figure E.2.2: *Du patch au tube*

Ainsi, le tube de mouvement se conceptualise à travers quatre composantes :

1. un patch de texture,
2. une trajectoire spatio-temporelle,
3. des déformations appliquées le long de la trajectoire,
4. et une durée de vie : le tube apparaît à un certain instant, et disparaît à un autre instant.

Dès lors, la séquence d'image peut être représentée par un ensemble de tubes qui peuvent être connectés les uns aux autres, ou inversement, déconnectés. De plus, chacun de ces tubes peut apparaître ou disparaître à n'importe quel instant.

Enfin, ils peuvent également se recouvrir ou se découvrir, introduisant ainsi des zones *multi-connectées* (représentées par plusieurs tubes) ou des zones *non-connectées* (représentées par aucun tube). Ces diverses possibilités sont illustrées dans la figure E.2.3.

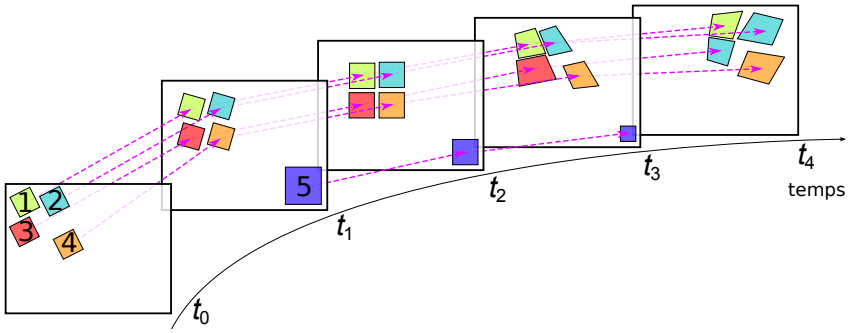


Figure E.2.3: *Une séquence d'image partiellement reconstruite par quelques tubes*

E.2.3 Problèmes soulevés par la représentation

Si les tubes de mouvement sont susceptibles d'améliorer la cohérence temporelle de la représentation, ils introduisent toutefois de nouvelles problématiques. Premièrement, quel modèle de déplacement et de déformation est le plus adapté aux tubes de mouvement ? Deuxièmement, sur quels critères doit-on se baser pour déterminer la durée de vie des tubes ; en d'autres termes, comment mesurer la qualité ou l'efficacité d'un tue vis-à-vis de la représentation ? Troisièmement, comment gérer les recouvrements (qui introduisent de la redondance) et les découvrements (qui doivent être comblés) des tubes ? Enfin, comment décrire l'évolution temporelle de la texture elle-même, du fait (par exemple) des changements d'illumination ou de pertes de résolution ? C'est à certaines de ces questions que nos travaux vont tenter de répondre.

E.3 Un modèle de mouvement basé tubes

Suivre des patchs de texture dans le temps nécessite l'utilisation d'un modèle de déplacement ainsi que d'un modèle de déformation afin de déterminer la trajectoire et la déformation subits par les patchs. Dans le domaine spatial, il s'agira notamment de se donner la possibilité de capturer tant les continuités que les discontinuités du champ de mouvement. Dans le domaine temporel, les modèles devront être en mesure d'exhiber la cohérence et la continuité du mouvement.

Si de nombreux modèles existent, certaines contraintes vont limiter nos choix. En effet, le contexte de compression implique l'utilisation de modèles aussi compacts que possible, et dont l'estimation des paramètres est relativement peu complexe en vue d'implantations matérielles.

D'autre part, les tubes eux-même vont guider notre choix : puisqu'ils partitionnent le domaine image en *patchs*, il devient alors assez naturel d'utiliser un modèle par mise en correspondance de *motif* (ex: bloc, mailles, régions). Reste donc à déterminer les propriétés du *motif* envisagé. Enfin, il s'agira également de mettre en place des contraintes de régularité afin d'encourager la cohérence de la représentation.

E.3.1 Blocs et mailles : un compromis entre capacité et complexité

Si de nombreuses solutions par mise en correspondance ont été proposées dans le cadre de l'estimation de mouvement, les solutions basées blocs et celles basées mailles prédominent. Ces deux approches diffèrent non seulement par leur complexité, mais aussi par leur capacités. Antagonistes de par leurs capacités, blocs et mailles sont tous deux intéressants pour les tubes de mouvement. La figure E.3.1 en synthétise les caractéristiques.

A la lumière de ce comparatif, il apparait clairement que le modèle par blocs est particulièrement intéressant de par sa simplicité ainsi que sa capacité à représenter les ruptures du champ de mouvement. Le modèle par maillage, lui, même s'il permet de représenter naturellement les continuités du mouvement, reste trop complexe dans le cadre de notre étude. Le modèle par blocs recouvrants, enfin, propose une alternative au mailles : pour un niveau de complexité réduit, il permet lui aussi de représenter de légères déformations, et donc les continuités du champ de mouvement, au prix d'un effet de flou. L'annexe A démontre en particulier que l'Overlapped BMC (OBMC) peut effectivement être considéré comme une

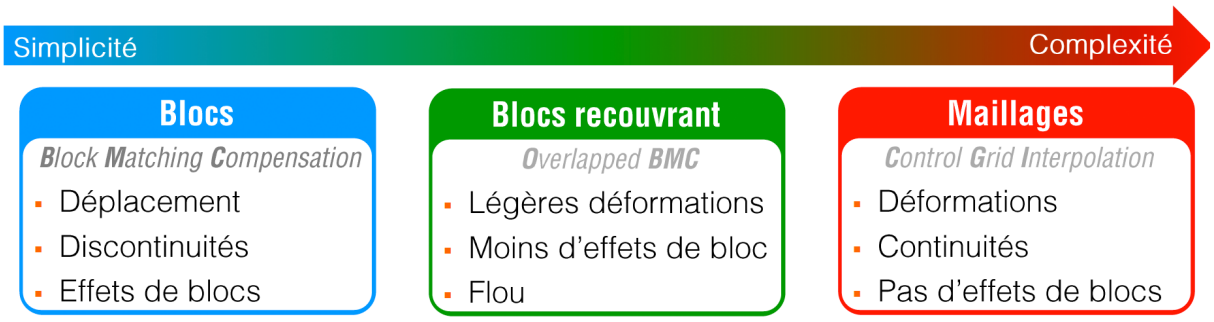


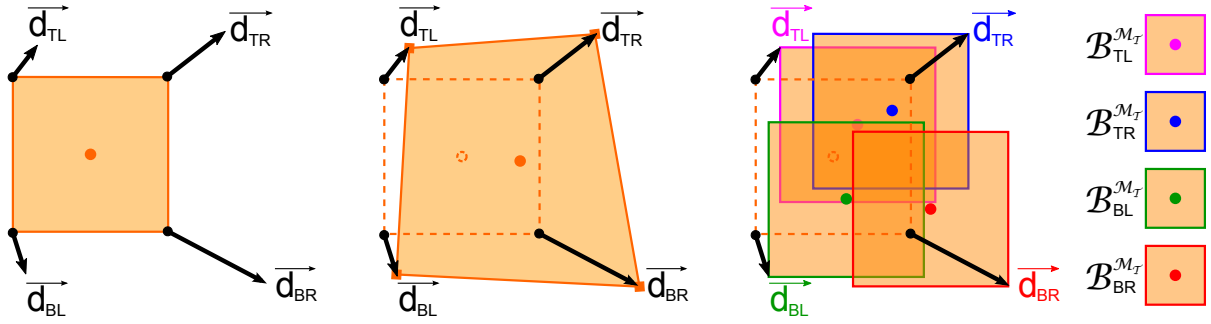
Figure E.3.1: *Complexité et caractéristiques des modèles de mouvement par blocs et maille : comparatif*

version basse complexité du modèle par maillage. C'est donc logiquement que notre approche va se concentrer sur le modèle par blocs et le modèle par blocs recouvrants (OBMC).

E.3.2 Un modèle hybride basé blocs et blocs recouvrants

Afin de représenter continuités et disonctinuités du mouvement, ainsi que déplacements et déformations des patchs, nos travaux se basent sur l'utilisation d'un modèle hybride connu sous le nom de Switched OBMC (SOBMC) [IM00], qui mèle un modèle purement translationnel (le Block Motion Compensation (BMC) [JJ81]) et un modèle qui permet de représenter des déformations simples (l'OBMC [NO92, AKOK92, OS94, CHJ+06]).

E.3.2.1 Construction du modèle de mouvement



(a) *Bloc de texture initial* **(b)** *Déformation idéale par maillage* **(c)** *Approximation de la déformation par OBMC*

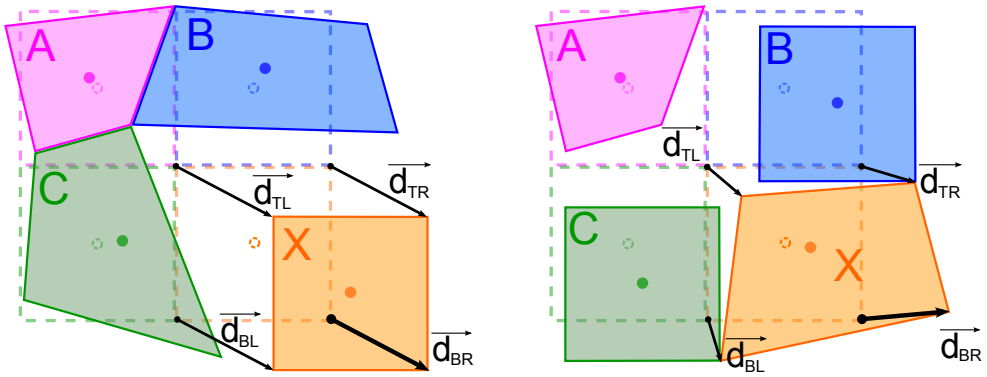
Figure E.3.2: *Modèle de déformation proposé pour les tubes*

Chaque tube est donc initialisé comme étant un bloc rectangulaire. A l'instar d'une maille, il est muni de quatre vecteurs de mouvement en chacun de ses coins (qui jouent donc le rôle de points de contrôle, au sens d'un maillage). A chaque instant, ces quatre vecteurs vont définir le déplacement et la déformation subits par le tube. Dans le cas général, le bloc initial est translaté aux quatre positions définies par les vecteurs de mouvement. Une fenêtre de pondération propre à chaque coin permet ensuite de sommer les différentes contributions pour un même point. La figure E.3.2 illustre le modèle proposé. Deux cas de figures sont alors envisagés :

- soit le **modèle translationnel** est utilisé, et les quatre vecteurs sont égaux (cela revient donc à un simple BMC) ; on peut ainsi représenter les **déconnections** entre tubes, faisant ainsi apparaître les **ruptures** du champ de mouvement ;
- soit le **modèle recouvrant** est utilisé, et les vecteurs haut-gauche, haut-droit et bas-gauche sont donnés par les tubes voisins à gauche et en haut; on peut ainsi représenter les **connections** entre tubes, et faire apparaître les **continuités** du champ de mouvement.

La figure E.3.3 montre comment ces deux modèles peuvent être couplés. Quatre tubes **A**, **B**, **C** et **X** sont initialement connectés (traits pointillés). A gauche, la figure E.3.3a illustre le cas où **A**, **B** et **C** sont gardés connectés, tandis que **X**

est déconnecté et subit un simple déplacement. A droite, la figure E.3.3b illustre le cas où **X** est connecté à ses voisins **B** et **C**, malgré les disparités de mouvement de **A**, **B** et **C**.

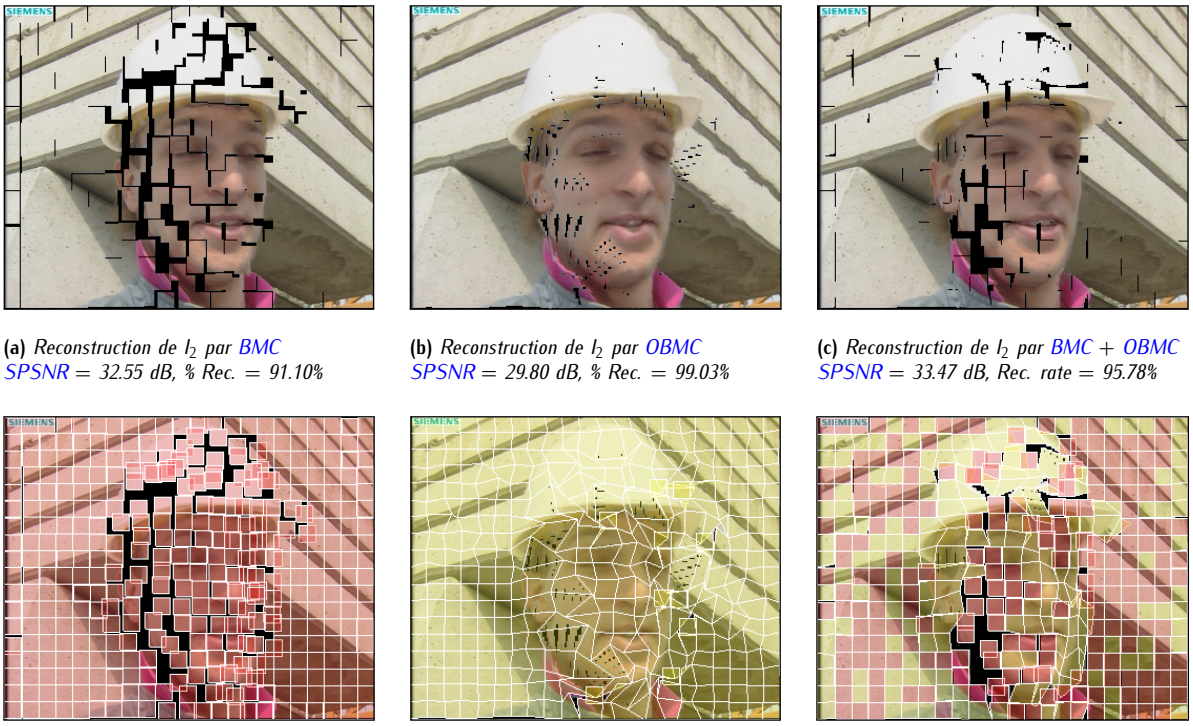


(a) *A, B et C: OBMC; X: BMC* **(b)** *A et X: OBMC; B et C: BMC*

Figure E.3.3: *Différentes combinaisons des deux modèles de mouvement*

E.3.2.2 Compensation en mouvement : résultats

La figure E.3.4 illustre les capacités des modèles de mouvement sélectionnés à compenser les images en mouvement. La séquence *Foreman* est d'abord partitionnée à t_0 en tubes à section carrée disjoints et connectés. Deux instants de temps de plus tard, on peut voir l'image $\widehat{I_2}$ compensée en mouvement ainsi que le déplacement ou la déformation des tubes. Les légendes de la figure E.3.4 mentionnent également le PSNR des zones reconstruites (noté SPSNR) ainsi que le taux de reconstruction (% Rec.).



(a) *Reconstruction de I_2 par BMC*
SPSNR = 32.55 dB, % Rec. = 91.10%

(b) *Reconstruction de I_2 par OBMC*
SPSNR = 29.80 dB, % Rec. = 99.03%

(c) *Reconstruction de I_2 par BMC + OBMC*
SPSNR = 33.47 dB, Rec. rate = 95.78%

(d) *Découpage de $\widehat{I_2}$ en tubes (BMC)*

(e) *Déformation des tubes sur $\widehat{I_2}$ (OBMC)*

(f) *Déformation et déplacement des tubes sur $\widehat{I_2}$ (BMC + OBMC)*

Figure E.3.4: *Modèles de mouvement BMC et OBMC : performances respectives et hybridation*

Cet exemple, représentatif de l'ensemble des séquences étudiées, montre à quel point l'hybridation des deux modèles BMC et OBMC permet de construire une compensation en mouvement plus cohérente et de meilleure qualité que celle produite par chacun des deux modèles pris individuellements.

E.3.3 Régularisation spatio-temporelle du mouvement

Si le modèle de mouvement hybride introduit par nos travaux permet de représenter avec efficacité tant les ruptures que les continuités du champ de mouvement, et donc les déplacements et les déformations de la texture à travers le temps, il n'est soumis pour l'instant à aucune contrainte spatio-temporelle et produit ainsi une description assez peu cohérente du mouvement à travers le temps. Notre principale motivation, quant à l'utilisation des tubes de mouvement, est de décrire la persistence temporelle des textures, et donc d'exhiber au maximum la cohérence de l'information à travers le temps et l'espace (le domaine image).

E.3.3.1 Minimisation sous contrainte de régularité

Pour cette raison, il est indispensable d'intégrer des critères de régularité au mécanisme d'estimation de mouvement. Non seulement la régularisation va garantir un maximum de cohérence à la représentation, mais elle va aussi permettre d'optimiser la compacité de l'information de mouvement. Généralement, le principe de la régularisation, appliquée à l'estimation de mouvement, consiste à minimiser une fonctionnelle J qui est donnée par la somme d'un terme d'attache aux données D et d'un terme de régularité R (exemple : la différence entre le vecteur de mouvement $\overrightarrow{d}$ courant et ses voisins $\overrightarrow{d_i}$),

$$J = \underbrace{\mathrm{Err}\left[I_{\mathrm{cur}}(P) - I_{\mathrm{ref}}(P+\overrightarrow{d})\right]}_{\text{Terme d'attache aux données } D} \quad + \quad \mu \underbrace{\left\| \sum_i \overrightarrow{d} - \overrightarrow{d_i} \right\|}_{\text{Terme de régularité } R} \tag{E.1}$$

où I_{cur} est l'image courante et I_{ref} l'image de référence, et μ un coefficient de pondération.

Si de nombreuses techniques de minimisation sous contrainte existent, une des plus populaires est celle des multiplicateurs de Lagrange. Le système à minimiser s'écrit alors

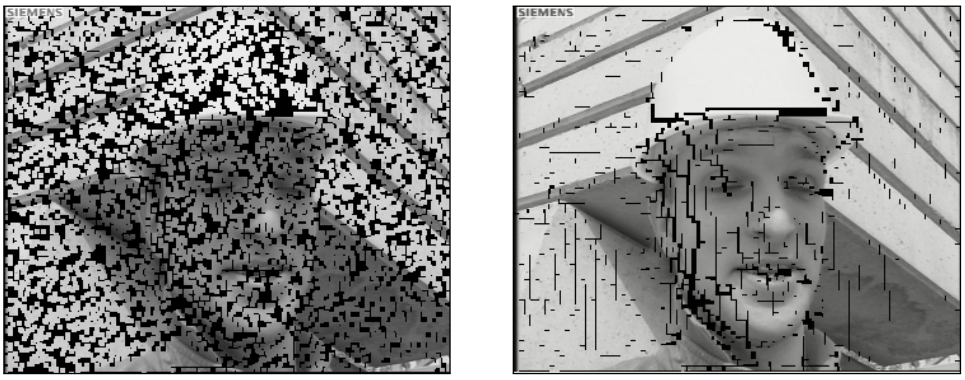
$$\textbf{min} \underbrace{\mathrm{Err}\left[I_{\mathrm{cur}}(P) - I_{\mathrm{ref}}(P+\overrightarrow{d})\right]}_{\mathrm{D}\ D} \quad + \quad \lambda \underbrace{R\left(\overrightarrow{d} - \overrightarrow{d_{\mathrm{pred}}}\right)}_{\text{Terme de régularité } R} \tag{E.2}$$

où λ est le multiplicateur de Lagrange, $\overrightarrow{d_{\mathrm{pred}}}$ est une prédiction de $\overrightarrow{d}$ et R est le coût de codage du résidu de mouvement. La régularisation du champ de mouvement est maintenant garantie par l'utilisation de prédicteurs de mouvement spatio-temporels $\overrightarrow{d_{\mathrm{pred}}}$: les vecteurs de mouvement dont le coût de codage (et donc le terme de régularité) sera le plus faible sont ceux définis par les prédicteurs.

E.3.3.2 Des prédicteurs spatio-temporels adaptés aux tubes

De part sa nature, la représentation des séquences d'images en tubes permet de construire une large variété de prédicteurs, tant dans la dimension spatiale (le domaine image) que dans la dimension temporelle. Le mouvement des tubes voisins, par exemple, constitue souvent un excellent prédicteur spatial. Quand à la dimension temporelle, elle est utilisée pour prédire le mouvement d'un tube à l'instant courant à l'aide du mouvement de ce même tube aux instants temporel précédents et/ou suivants. La notion de trajectoire peut ici être exhibée à travers une estimation temporellement hiérarchique du mouvement : la trajectoire est d'abord estimée entre deux instants de temps distants, puis successivement raffinée à mesure que les instants intermédiaires sont estimés à leur tour. D'autre part, l'estimation de mouvement est elle aussi régularisée via une approche de type EPZS, où les vecteurs de mouvement candidats sont également obtenus à l'aide de prédicteurs spatio-temporels.

Cet ensemble de prédicteurs permet alors de régulariser efficacement le mouvement et la déformation des tubes, et ainsi de proposer une représentation cohérente de la séquence. La figure E.3.5 montre comment la régularisation améliore singulièrement la cohérence des images reconstruites.



(a) *Sans régularisation*
SPSNR = 41.94 dB, % Rec. = 68.12%

(b) *Avec régularisation*
SPSNR = 39.81 dB, % Rec. = 93.45%

Figure E.3.5: *Impact de la régularisation sur la sequence* Foreman

E.4 Structuration de la représentation

Si la section précédente abordait le modèle de mouvement et de déformation proposé pour les tubes de mouvement, elle n'indiquait pas comment la représentation dans son ensemble est structurée. Comment découper (spatialement) les images en tubes de manière à ce que la plus grande partie de la séquence puisse être reconstruite ? Comment s'assurer que les zones non connectées (synthétisés par aucun tube) soient aussi peu nombreuses que possible ? C'est à ces deux questions que la présente section va répondre à travers deux solutions simples et efficaces.

E.4.1 La famille de tubes

Afin de simplifier la gestion des tubes, nous introduisons d'abord le concept de famille de tubes de mouvement. Une famille de tubes est un ensemble de tubes qui sont initialisés au même instant de référence, et qui sont suivis à travers le même groupe d'images (GOP). La structuration spatiale et temporelle de la représentation se base sur l'utilisation d'une ou plusieurs de ces familles. La figure E.4.1 illustre la notion de famille de tubes. A l'instant de référence, l'image est découpée en blocs de pixels (ou patchs) non recouvrants. Chacun de ces blocs est ensuite utilisé pour initialiser un tube de mouvement qui sera suivant à travers le GOP.

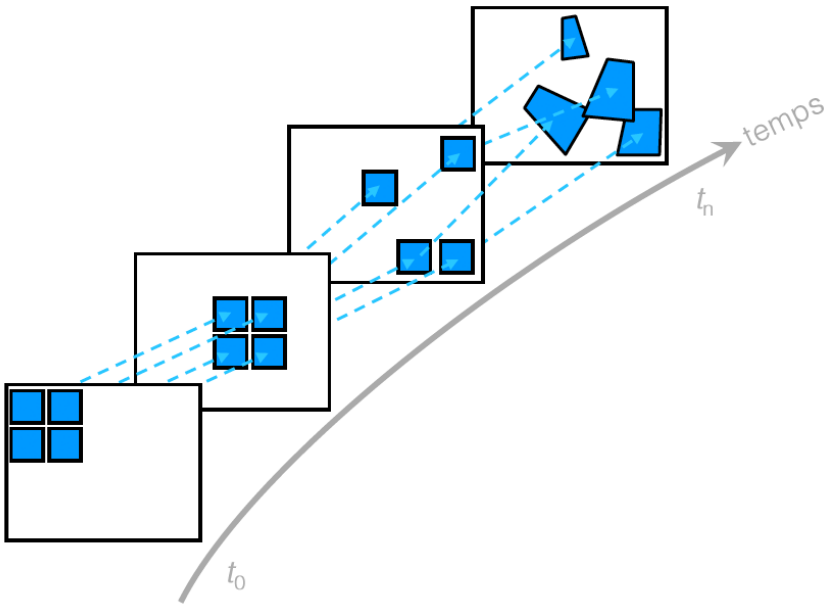
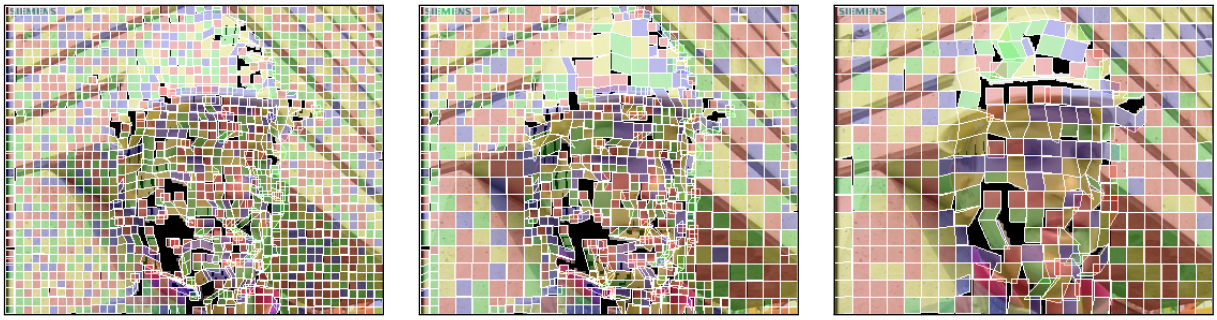


Figure E.4.1: *Une famille de tube initialisée à l'instant t_0 et suivie jusque t_n*

E.4.2 Adaptation au contenu spatial : granularité des tubes

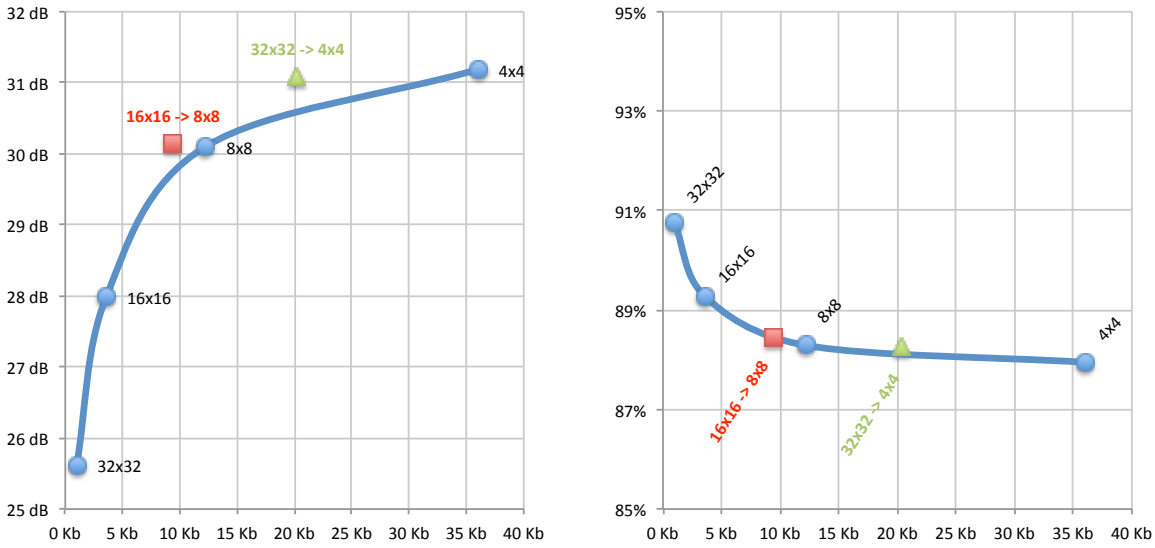
Si nos premières expérimentations partitionnent les images en patchs à dimensions fixes et égales, notre schéma a ensuite évolué vers une approche adaptative. En effet, une question a rapidement émergé : quelle est la dimension optimale pour les tubes ? Les zones dont le mouvement est uniforme se prêtent à l'utilisation de tubes d'assez grande taille, tandis que celles dont le mouvement est plus complexe requièrent des tubes de plus faibles dimensions. La figure E.4.2 présente le découpage d'une image de la séquence Foreman proposée par des tubes de différentes tailles, fixes ou variables.



(a) 8×8 *tubes sur* $\widehat{I_2}$ **(b)** *Tubes à taille variable sur* $\widehat{I_2}$ **(c)** 16×16 *tubes sur* $\widehat{I_2}$

Figure E.4.2: *Tubes à taille variales: application à la séquence* Foreman

Pour ces raisons, il a été proposé de déterminer une partition optimale via optimisation Lagrangienne. Au coût de codage qui incorporait déjà le coût du résidu de mouvement, on ajoute le coût du partitionnement, permettant ainsi de déterminer le meilleur compromis entre taille du tube, le coût du mouvement, et la qualité de l'image reconstruite. La figure E.4.3 montre que malgré le surcoût de codage lié à la transmission de la partition optimale, l'utilisation des tubes à taille variable propose une meilleure reconstruction pour un même débit, ou inversement un débit moindre pour une même qualité de reconstruction. Quant au taux de reconstruction, il n'est ni amélioré ni réduit par les tubes à taille variable.



(a) *SPSNR vs débit* **(b)** *Taux de reconstruction vs déit*

Figure E.4.3: *Tubes à taille variables: courbes débit-distortion et débit-reconstruction*

E.4.3 Adaptation au contenu temporel : plusieurs familles de tubes

Si le découpage spatial des images est désormais assuré par les tubes à taille variable, une seule et même famille de tubes ne suffit que rarement à représenter l'ensemble d'un groupe d'images. En effet, tout patch de texture invisible ou

absent à l'instant de référence n'est alors représenté par aucun tube lorsqu'il apparaît aux autres instant. C'est donc très naturellement que nous avons proposé l'utilisation de familles de tubes additionnelles, chacune initialisée à des instants de temps différents. La figure E.4.4 en illustre le principe à travers l'utilisation de trois familles (rouge, bleue et verte).

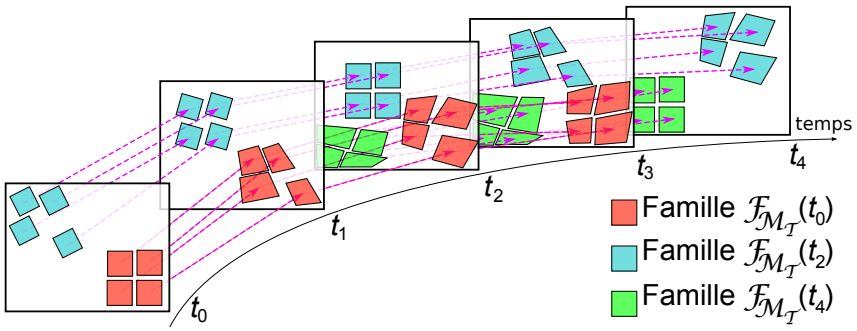
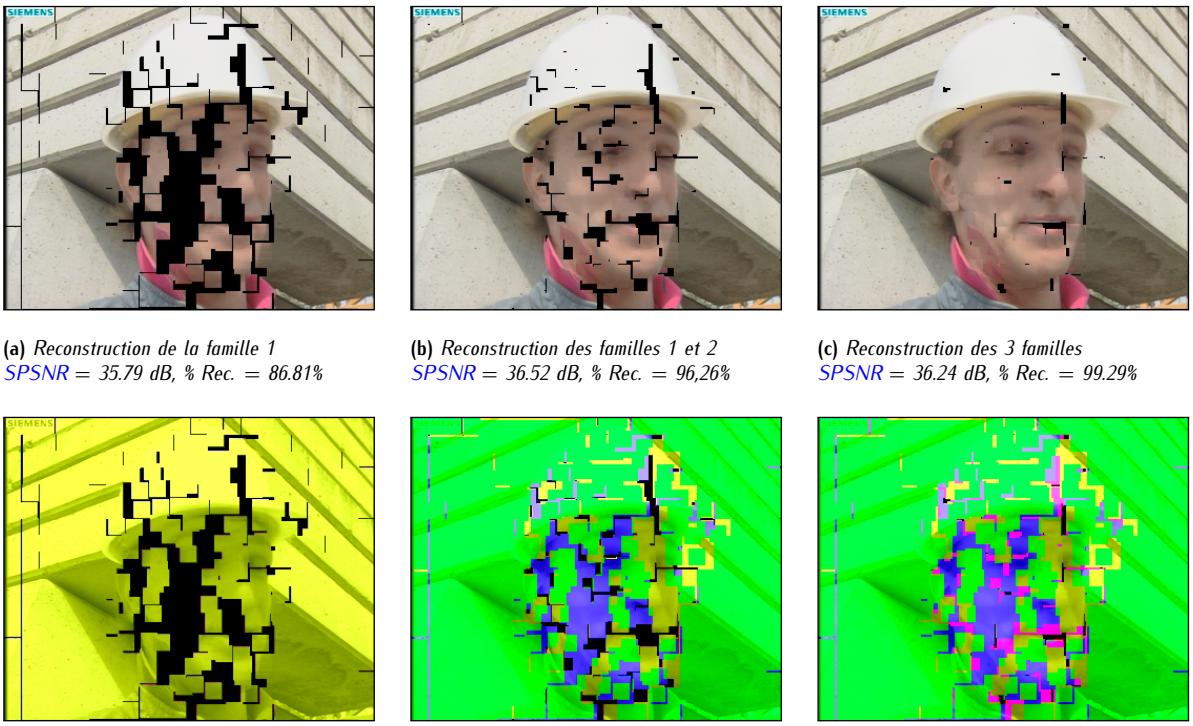


Figure E.4.4: *Plusieurs familles pour reconstruire un groupe d'images*

En pratique, l'utilisation de plusieurs familles contribue effectivement à augmenter le taux de reconstruction. La figure E.4.5 montre comment la reconstruction de la séquence *Foreman* est assurée par plusieurs familles de tubes. Une première famille de tubes est initialisée à t_0 (zones en jaune) et suivie jusque t_8. Une seconde famille est alors initialisée à t_8 et suivie vers t_0 et reconstruit d'avantage l'image (zones en bleu lorsque seule la seconde famille est utilisée, et en vert lorsqu'une pondération des deux familles est utilisée). Enfin, une troisième et dernière famille, initialisée en t_4 et suivie vers les deux extrémités du GOP, permet de combler la plupart des zones non connectées restantes. Bien entendu, les "trous" restants peuvent être comblés via des mécanismes de synthèse de texture ou de structure tels que l'inpainting.



(a) *Reconstruction de la famille 1*
SPSNR = 35.79 dB, % Rec. = 86.81%

(b) *Reconstruction des familles 1 et 2*
SPSNR = 36.52 dB, % Rec. = 96,26%

(c) *Reconstruction des 3 familles*
SPSNR = 36.24 dB, % Rec. = 99.29%

(d) *Zones reconstruites par la famille 1*

(e) *Zones reconstruites par les familles 1 et 2*

(f) *Zones reconstruites par les familles 1, 2 et 3*

Figure E.4.5: *Reconstruction de la sequence* Foreman *à l'aide de trois familles de tubes*

E.5 Validation du modèle proposé

Si l'objectif principal de nos travaux est de proposer une nouvelle représentation pour les séquences d'images, il ne s'agit pas d'oublier que nous nous situons dans le contexte de la compression. Aussi, il est important de s'assurer que la représentation est effectivement compacte et compétitive face aux représentations existantes : en d'autres termes, que la représentation est viable. Dans cet objectif, nos travaux ont cherché à mettre en compétition la représentation par tubes de mouvement à l'approche standard de compression qu'est H.264/AVC. Toutefois, nous nous sommes concentrés sur l'aspect représentation, et n'avons cherché qu'à vérifier que la reconstruction des images proposées par les tubes est avantageuse vis-à-vis de la représentation classique par blocs. La figure E.5.1 représente le schéma de codage modifié pour permettre l'insertion du modèle par tubes de mouvement.

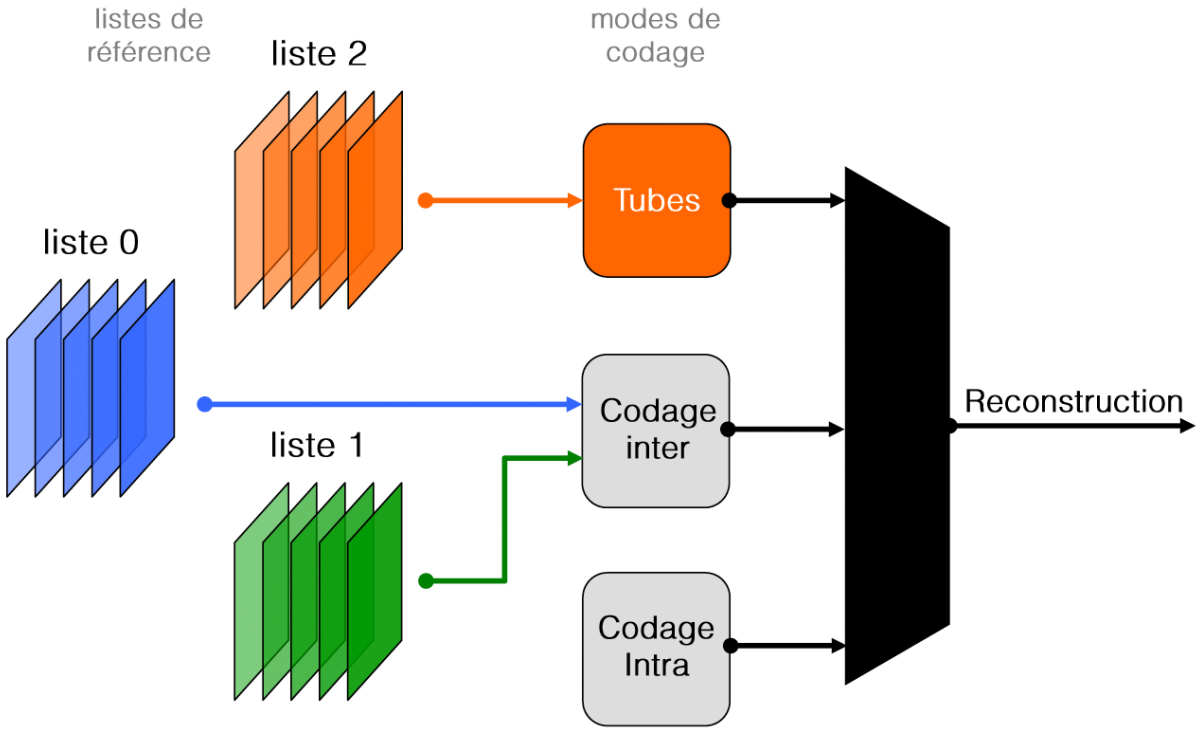


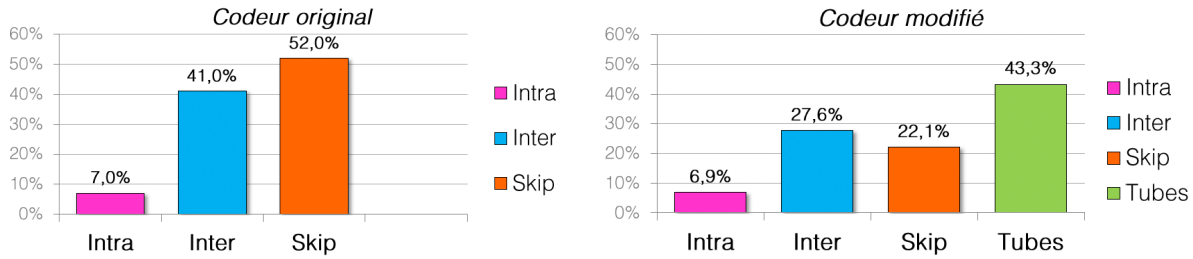
Figure E.5.1: *Schéma de codage AVC modifié par l'insertion du modèle par tubes*

Afin de mener à bien cette étape de validation, nous avons intégré le mode de prédiction par tubes de mouvement au sein du codeur H.264/AVC. Dans le codeur modifié, la décision d'utiliser ou de ne pas utiliser les tubes ne se base que sur la distorsion et ne tient pas compte du coût de codage des tubes. En effet, le manque d'optimisation du codage des tubes de mouvement les rend inintéressants dans un contexte débit-distorsion. La figure E.5.2 montre cependant que les tubes sont très largement utilisés par le codeur modifié, et donc que la représentation est viable en terme de distorsion. A terme, il s'agira bien entendu de mettre en place un mécanisme de codage adapté aux tubes afin de proposer effectivement des performances de compression intéressantes, qui pour l'instant présentent peu d'intérêt tant le codage et la gestion des tubes sont peu optimisés.

E.6 Vie et mort des tubes

Si la représentation par tubes de mouvement permet effectivement d'exhiber la persistance et la cohérence temporelle de l'information, elle repose pour l'instant sur l'hypothèse selon laquelle chaque patch de texture est disponible pendant la totalité du groupe d'image représenté. Or, en pratique, une grande partie de ces patchs n'est pas disponible pendant toute la durée d'un GOP et ne peuvent donc pas être suivis.

Afin d'optimiser la compacité de la représentation, il s'agit alors de mettre en place un mécanisme de vie et de mort des tubes. En d'autre termes, des outils permettant de supprimer une partie ou l'intégralité d'un tube si le patch de texture correspondant disparaît ou devient impossible à suivre. On améliore ainsi la compacité de la représentation en



(a) *Codeur initial* **(b)** *Codeur modifié*

Figure E.5.2: *Taux d'utilisation des modes de prédiction H.264/AVC et des tubes dans les codeurs étudiés ; moyenne sur six séquences et quatre débits*

utilisant moins de tubes pour représenter la séquence. En terme de compression, on évite également la transmission de mouvements incohérents, lorsque le suivi échoue, qui sont typiquement difficiles à coder.

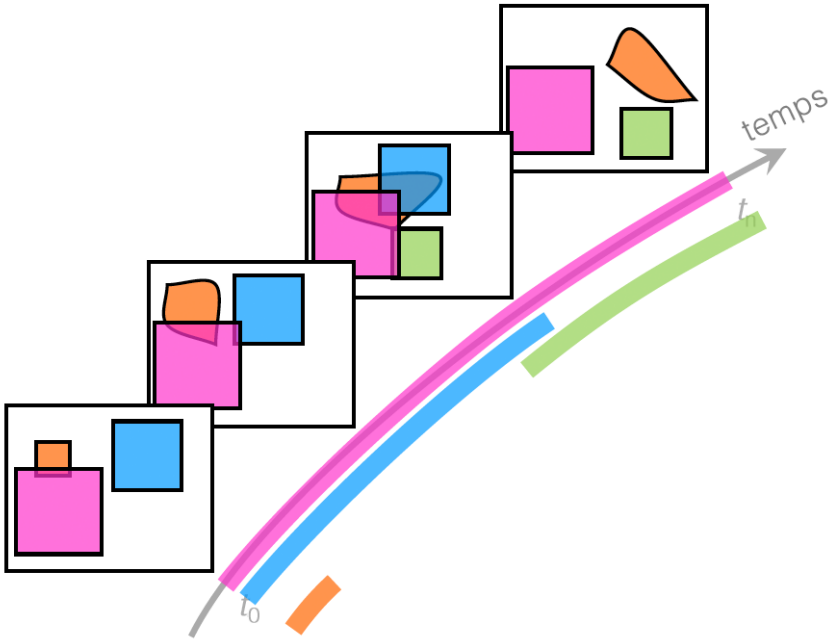


Figure E.6.1: *Disponibilité d'un patch de texture à travers un GOP : quatre scenarii à envisager*

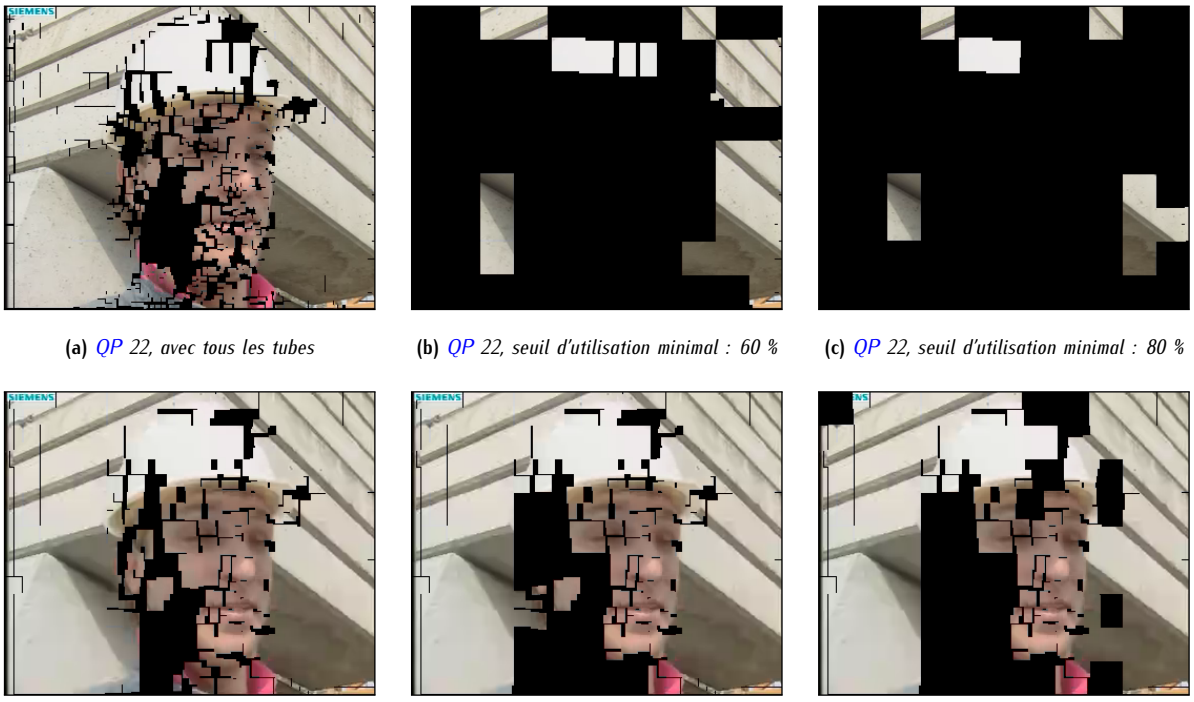
Idéalement, un mécanisme de vie et de mort devrait être en mesure de détecter plusieurs cas de figure (voir figure E.6.1):

- les patchs qui sont disponibles pendant l'intégralité du GOP (en rose sur la figure),
- les patchs qui disparaissent (ex: occultations) au cours du GOP (en bleu sur la figure),
- les patchs qui apparaissent (ex: désoccultation) au cours du GOP (en vert sur la figure),
- et les patchs qui ne peuvent pas être suivis (en orange sur la figure).

En pratique, cela requiert la construction d'un critère de qualité et/ou d'efficacité propre aux tubes : ce critère permettra ainsi de mesurer à quel point un tube est efficace au sens de la représentation, et donc de le continuer ou de l'arrêter, de le garder ou le supprimer selon les situations. Or, il est difficile de définir un tel critère tant les paramètres qui régissent le mouvement et la déformation des tubes sont nombreux. Dans le domaine spatial, la position, la distorsion, et la déformation, pour n'en citer que trois, sont à prendre en compte. De plus, certains paramètres temporels jouent également un rôle primordial, dont la trajectoire et la régularité du mouvement. Enfin, si les critères mentionnés précédemment sont purement objectifs, on peut également envisager l'utilisation de critères subjectifs.

Devant la multitude critères à envisager, nos travaux ont visé à proposer une solution préliminaire qui se base sur les

choix opérés par le codeur H.264/AVC modifié. En étudiant l'utilisation des tubes par ce dernier, on ne vient garder que ces tubes dont l'utilisation spatiale et/ou temporelle est supérieure à un certain seuil. Dans la figure E.6.2, on observe bien que la sélection des tubes privilégie ceux qui correspondent aux zones uniformes et dont le suivi est aisé. A l'inverse, les zones plus complexes comme le visage de *Foreman* sont elles moins sélectionnées.



(a) *QP 22, avec tous les tubes* **(b)** *QP 22, seuil d'utilisation minimal : 60 %* **(c)** *QP 22, seuil d'utilisation minimal : 80 %*

(d) *QP 37, avec tous les tubes* **(e)** *QP 37, seuil d'utilisation minimal : 60 %* **(f)** *QP 37, seuil d'utilisation minimal : 80 %*

Figure E.6.2: *Sequence* Foreman *: sélection des tubes selon le taux d'utilisation de ces derniers par le codeur H.264/AVC modifié*

Si dans le domaine spatial, le choix du codeur peut s'avérer judicieux, ce n'est absolument pas le cas dans le domaine temporel : les décisions prises par le codeur se font instant par instant, et ne considèrent donc pas la dimension temporelle et persistante des tubes. Ainsi, rien ne garantit que le mécanisme de sélection proposé aboutisse à des décisions cohérentes le long de l'axe temporel. C'est effectivement le cas : la solution préliminaire devra ensuite être améliorée en filtrant la sélection au cours du temps.

E.7 Conclusions et perspectives

Avec les tubes de mouvement, nos travaux introduisent une nouvelle représentation pour les séquences d'images dont l'objectif principal est d'exhiber la persistance et la cohérence temporelle de la texture. D'autre part, les tubes de mouvement forment une représentation pseudo-sémantique et flexible, tout en maintenant la complexité à un niveau raisonnable.

Notre première contribution consiste en l'élaboration d'un modèle de mouvement qui permet de décrire tant les déplacements que les déformations des tubes, ainsi que leurs connections et leurs déconnections. D'autre part, des étapes de régularisation permettent au mouvement estimé d'être relativement cohérent tant spatialement que temporellement.

Notre seconde contribution a posé les bases de la représentation dans son ensemble à travers l'emploi de familles de tubes. Dans le domaine spatial, la dimension des tubes a été adaptée aux propriétés du mouvement. Dans le domaine temporel, l'emploi de plusieurs familles de tubes permet de maximiser le taux de reconstruction et donc de réduire les zones non prédites.

Fort de cette première représentation, nous avons cherché à en vérifier la viabilité à travers son intégration dans le standard de compression H.264/AVC. Certes le manque d'optimisation de la transmission des tubes se traduit en performances de compression significativement réduites, mais les résultats obtenus montrent d'abord et avant tout que les tubes sont effectivement capables de représenter un groupe d'images. Les perspectives d'améliorations en termes de

compression sont nombreuses ; on pensera en tout premier lieu à supprimer les tubes qui sont inutiles et non sélectionnés, réduisant ainsi leur coût de codage.

Maintenant dotés d'un outil d'évaluation, nous avons alors pu nous pencher sur le problème de la vie et de la mort des tubes. La solution proposée est préliminaire, et se base sur les décisions opérées par le codeur H.264/AVC modifié. En perspective, il est crucial de développer un critère permettant d'évaluer la qualité et/ou l'efficacité d'un tube, qu'il soit objectif ou subjectif. Celui-ci servira alors à déterminer quels sont les tubes qui doivent être gardés et quels sont ceux qui doivent être supprimés. D'autre part, il est également important de régulariser cette sélection le long de l'axe temporel.

Enfin, si les tubes proposent une nouvelle manière de représenter les vidéos, ils ouvrent aussi la voie à de nombreuses extensions. Par exemple, la description d'une même zone à travers plusieurs tubes de différentes tailles, qualités ou durées de vie, permet d'introduire les scalabilités en qualité, spatiale et temporelle. La scalabilité en mouvement est elle déjà accessible via raffinements successifs de la trajectoire des tubes. On peut également penser à étendre le concept des tubes pour décrire les changements entre différentes vues d'une même scène, particulièrement intéressant dans le cadre de la vidéo 3D. Enfin, le caractère pseudo-sémantique de la représentation est lui aussi intéressant en vue de la description et de l'indexation, qui peuvent être grandement facilitées par l'analyse de la séquence réalisée par les tubes.

Bibliography

[AAB+84] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, "Pyramid methods in image processing," *RCA Engineer*, vol. 29, no. 6, pp. 33–41, Nov/Dec 1984.

[Abh02] G. C. K. Abhayaratne, "Lossless and Nearly Lossless Digital Video Coding," Ph.D. dissertation, University of Bath, UK, 2002.

[Abh03] ——, "N-Points Discrete Cosine Transforms that Map Integers to Integers for Lossless Image and Video Coding," in *Proceedings of Picture Coding Symposium (PCS'03)*, Apr 2003, pp. 417–422, Saint-Malo, France.

[AFBD06] S. Amir, E. Flécher, M. Babel, and O. Déforges, "LAR vidéo: Codage sans perte à scalabilité sémantique," in *Proceedings of (CORESA'06)*, Nov 2006.

[AFJ90] M. H. Ahmad Fadzil and D. T. J., "A hierarchical motion estimator for interframe coding," in *Proceedings of IEEE Colloquium on Applications of Motion Compensation*, no. 128, Oct 1990.

[AKOK92] C. Auyeung, J. J. Kosmach, M. T. Orchard, and T. Kalafatis, "Overlapped block motion compensation," *Proceedings of SPIE, Visual Communications and Image Processing*, vol. 1818, pp. 561–572, 1992.

[AMB+04] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. V. der Schaar, J. Cornelis, and P. Schelkens, "In-band motion compensated temporal filtering," *Signal Processing: Image Communication*, vol. 19, no. 7, pp. 653–673, Aug 2004.

[And07] T. André, "Codage vidéo scalable par transformée en ondelettes et mesure de distortion entropique," Ph.D. dissertation, Université de Nice-Sofia Antipolis, Sep 2007.

[ANR74] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete Cosine Transform," *IEEE Transactions on Computers*, vol. 32, no. 1, pp. 90–93, Jan 1974.

[ARAM03] G. Al-Regib, Y. Altunbasak, and R. M. Mersereau, "Hierarchical Motion Estimation With Content-Based Meshes," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 10, pp. 1000–1005, Oct 2003.

[Arg60] J. H. Argyris, *Energy theorems and structural analysis.* Butterworth (London), 1960.

[Ash01] M. Ashikhmin, "Synthesizing Natural Texture," in *Proceedings of Symposium on Interactive 3D Graphics*, 2001.

[AT97a] Y. Altunbasak and A. M. Tekalp, "Scalable mesh-based interpolative coding of synthethic and natural image objects," in *Proceedings of SPIE*, vol. 3024, 1997, pp. 1004–1012.

[AT97b] Y. Altunbask and A. M. Tekalp, "Closed-form connectivity-Preserving solutions for motion compensation using 2-D meshes," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1255–1269, Sep 1997.

[BA83] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31, no. 4, pp. 532–540, Apr 1983.

[BA91] M. J. Black and P. Anandan, "Robust Dynamic Motion Estimation Over Time," in *Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR'91)*, Jun 1991, pp. 296–302, Maui, Hawai.

[BAHH92] J. R. Bergen, P. An, T. J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Proceedings of the 2nd European Conference on Computer Vision*, vol. 588, 1992, pp. 237–252.

[BD09] M. Babel and O. Déforges, "WG1N4870 - Response to call for AIC techniques and evaluation methods," ISO/ITU JPEG commitee, San Francisco, Tech. Rep., Jan 2009.

[BDH99] E. B. Bellers and G. De Haar, "Analysis of sub-pixel Motion Estimation," in *Proceedings of Visual communications and image processing (VCIP'99)*, vol. 3653, no. 2, Jan 1999, pp. 1452–1463, San Jose, CA, USA.

[BDR05] M. Babel, O. Déforges, and J. Ronsin, "Interleaved S+P pyramidal decomposition with refined prediction model," in *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, vol. 2, 2005, pp. 750–753.

[BDRB07] O. Brouard, F. Delannay, V. Ricordel, and D. Barba, "Fast long-term motion estimation for high definition video sequences based on spatio-temporal tubes and using the Nelder-Mead simplex algorithm," in *Picture Coding Symposium, PCS 2007*, Lisbonne, Portugal, Nov 2007. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00252140/en/

[BGLS06] J. F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal, *Numerical optimization, theoretical and numerical aspects*, 2nd ed. Springer-Verlag, 2006.

[BGM06] R. Balter, P. Gioia, and L. Morin, "Scalable and efficient coding using 3D modeling," *IEEE Transactions on Multimedia*, vol. 8, pp. 1147–1155, Dec 2006.

[BJM+88] M. F. Barnsley, A. Jacquin, F. Malassenet, L. Reuter, and A. D. Sloan, "Harnessing chaos for image synthesis," in *Proceedings of the 15th Annual Conference on Computer Graphics and Iterative Techniques (ACCGIT'88)*, June 1988, pp. 131–140.

[BL91] N. Baaziz and C. Labit, "Multigrid motion estimation on pyramidal representation for image sequence coding," INRIA Rennes, Tech. Rep. 1409, Apr 1991.

[BL02] M. Barlaud and C. Labit, *Compression et codage des images et des vidéos*. Hermès, 2002.

[Bra00] R. N. Bracewell, *The Fourier Transform and Its Applications*. MacGraw-Hill, Boston, 2000.

[Bru90] H. Brusewitz, "Motion compensation with triangles," in *Proceedings of 3rd International Conference on 64 kbit Coding of Moving Video*, Sep 1990, Rotterdam, Netherlands.

[Bru01] E. Bruno, "De l'estimation locale à l'estimation globale de mouvement dans les séquences d'images," Ph.D. dissertation, Université Joseph Fourier, Laboratoire des Images et des Signaux, Grenoble, Nov 2001.

[BS90] M. F. Barnsley and A. D. Sloan, "Methods and apparatus for image compression by iterated function system," U.S. Patent 4 941 193, Jul 10, 1990. [Online]. Available: http://www.google.com/patents/about?id=vC4aAAAAEBAJ&dq=4941193

[BS92] R. H. Bamberger and J. T. Smith, "A Filter Bank for the Directional Decomposition of Images: Theory and Design," *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 882–893, Apr 1992.

[BSCB00] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image Inpainting," in *Proceedings of SIGGRAPH'00*, July 2000, New Orleans, LA, USA.

[BTZ+99] P. V. Beek, A. Tekalp, N. Zhang, I. Celasum, and M. Xia, "Hierarchical 2D-mesh representation tracking and compression for object video," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 353–369, Mar 1999.

[BVSO02] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," UCLA CAM Report, Tech. Rep. 47, 2002.

[BW07] J. Ballé and M. Wien, "Extended texture prediction for H. 264/AVC intra coding," in *Proceedings of International Conference on Image Processing (ICIP'07)*, 2007.

[BZ07a] J. Bu and B. Zeng, "A Comparative Study of Compensation Techniques in Directional DCT's," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'07)*, May 2007, pp. 521–524, New-Orléans, LA, USA.

[BZ07b] ——, "Directional Discrete Cosine Transforms: A Theoretical Analysis," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'07)*, vol. 1, Apr 2007, pp. 1105–1108, Honolulu, HI, USA.

[Cam04] N. Cammas, "Codage vidéo scalable par maillages et ondelettes t+2D," Ph.D. dissertation, Université de Rennes I, Rennes, France, Nov 2004.

[Can98] E. J. Candès, "Ridgelets: theory and applications," Ph.D. dissertation, Department of Statistics, Stanford University, Palo Alto, CA, USA, 1998.

[Car88] S. Carlsson, "Sketch based coding of grey level images," *Signal Processing*, vol. 15, no. 1, pp. 57–83, 1988.

[Car89] S. Carlsonn, "Object detection using model based prediction and motion parallax," in *Proceedings of Stockholm Workshop on Computational Vision*, Aug 1989, Stockholm, Sweden.

[Cas85] P. Cassereau, "A new class of optlmal unitary transforms for image processing," Master's thesis, Departement of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, MA, USA, May 1985.

[CCCL95] M.-J. Chen, L.-G. Chen, T.-D. Chiueh, and Y.-P. Lee, "A new block matching criterion for motion estimation and its implementation," *IEEE Transactions on Image Processing*, vol. 5, no. 3, pp. 231–236, Jun 1995.

[CD99a] E. J. Candès and D. L. Donoho, "Ridgelets: a key to higher-dimensional intermittency ?" *Royal Society of London Philosophical Transactions*, vol. 357, no. 1760, pp. 2495–2509, Sep 1999.

[CD99b] ——, *A Surprisingly Effective Nonadaptive Representation for Objects with Edges*. Vanderbilt University Press, Nashville, TN, USA, 1999.

[CDF92] A. Cohen, I. Daubechies, and J. Feauveau, "Biorthogonal Bases of Compactly Supported Wavelets," *Communications on Pure and Applied Mathematics*, vol. 45, no. 5, pp. 485–560, Mai 1992.

[CDMS98] S. S. Chen, D. L. Donoho, Michael, and M. A. Saunders, "Atomic Decomposition by Basis Pursuit," *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1998.

[CGM04] V. Chappelier, C. Guillemot, and S. Marinkovic, "Image coding with iterated Contourlets and Wavelet transforms," in *Proceedings of IEEE International Conference on Image Processing (ICIP'04)*, Oct 2004, pp. 3157–3160, Singapore, Singapore.

[CH73] C. C. Cutler and N. J. Holmdel, "Conditional Replenishment video encoder with predictive updating," U.S. Patent 3 720 786, Mar, 1973. [Online]. Available: http://www.google.fr/patents?id=9sosAAAAEBAJ&zoom=4&dq=Conditional%20replenishment&pg=PP1#v=onepage&q&f=false

[CH07] D. M. Chandler and S. S. Hemami, "VSNR: A Wavelet-Based Visual Signal-to-Noise Ratio for Natural Images," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2284–2298, 2007.

[CHJ+06] B.-D. Choi, J.-W. Han, S.-W. Jung, J.-H. Nam, and S.-J. Ko, "Overlapped Block Motion Compensation based on irregular grid," in *Proceedings of International Conference on Image Processing (ICIP'06)*, 2006, pp. 1085–1088.

[CIS10] CISCO, "Cisco Visual Networking Index: Forecast and Methodology, 2009-2014," White Paper, Jun 2010.

[CJF73] P. Cummiskey, N. S. Jayant, and J. L. Flanagan, "Adaptive quantilation in differential PCM coding of speech," *Bell System Technical Journal*, vol. 52, pp. 1105–1118, Sep 1973.

[CJJ03] W. I. Choi, B.-W. Jeon, and J.-C. Jeong, "Fast motion estimation with modified diamond search for variable motion block sizes," in *Proceedings of International Conference on Image Processing (ICIP'03)*, vol. 2, 2003, pp. 371–374.

[CKS96] W. C. Chung, F. Kossentini, and M. J. T. Smith, "An efficient motion estimation technique based on a rate-distortion criterion," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP'96)*, vol. 4, May 1996, pp. 1926–1929.

[Clo60] R. Clough, "The Finite Element in Plane Stress Analysis," in *Proceedings of the 2nd ASCE Conference on Electronic Computations*, 1960, Pittsburgh, PA, USA.

[CLW69] J. Cooley, P. Lewis, and P. Welch, "The finite Fourier transform," *IEEE Transactions on Audio and Electroacoustics*, vol. 17, no. 2, pp. 77–85, 1969.

[CON00] Y. J. Chen, S. Oruintura, and T. Nguyen, "Video compression using integer DCT," in *Proceedings of IEEE International Conference on Image Processing (ICIP'00)*, vol. 2, 2000, pp. 844–847, Vancouver, Canada.

[COT+02] Y. J. Chen, S. Oraintara, T. D. Tran, K. Amaratunga, and T. Nguyen, "Multiplierless Approximation of Transforms with Adder Constraint," *IEEE Signal Processing Letters*, vol. 9, no. 11, pp. 344–347, Nov 2002.

[CP03] N. Cammas and S. Pateux, "Fine grain scalable video coding using 3D wavelets and active meshes," in *SPIE Visual Communications and Image Processing*, Jan 2003, Santa Clara, CA, USA.

[CPML10] T. Colleu, S. Pateux, L. Morin, and C. Labit, "A polygon soup representation for multiview coding," *Journal of Visual Communication and Image Representation*, vol. 21, no. 5-6, pp. 561–576, Jul-Aug 2010.

[CPW+09] J. Chen, S. Paris, J. Wang, W. Matusik, M. Cohen, and F. Durand, "The Video Mesh: A Data Structure for Image-based Video Editing," Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, Tech. Rep. MIT-CSAIL-TR-2009-062, Dec 2009.

[CR83] C. Cafforio and F. Rocca, "The differential method for motion estimation," *Image Sequence Processing and Dynamic Scene Analysis*, vol. 12, pp. 102–124, 1983.

[CRGP94] E. Chan, A. A. Rodriguez, R. Ghandi, and S. Panchanathan, "Experiments on block-matching techniques for video coding," *Multimedia Systems*, vol. 2, no. 5, pp. 228–241, Dec 1994.

[CS01] T. Chan and J. Shen, "Non-texture Inpainting by Curvature-Driven Diffusions," *Journal of Visual Communication and Image Representation*, vol. 12, no. 12, pp. 436–449, Dec 2001.

[CS02] ——, "Mathematical models for local non-texture inpaintings," *SIAM Journal of Applied Mathematics*, vol. 63, no. 3, pp. 1019–1043, 2002.

[CSF77] W. H. Chen, , H. C. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Transactions on Communications*, vol. 25, no. 9, pp. 1004–1009, Sep 1977.

[CSJ88] P. M. Cassereau, D. H. Staelin, and G. D. Jager, "Encoding of images based on a lapped orthogonal transform," *IEEE transactions on communications*, vol. 37, no. 2, pp. 189–193, 1988.

[Cut52] C. C. Cutler, "Differential Quantization of Communication Signals," U.S. Patent 2 605 361, July 29, 1952.

[CW98] M. C. Chen and A. N. Willson, "Rate-distortion optimal motion estimation algorithms for motion-compensated transform video coding," *IEEE Transactions on Circuits and Systems for Video Technology (CSVT)*, vol. 8, no. 2, pp. 147–158, Apr 1998.

[CW99] S.-J. Choi and J. W. Woods, "Motion-compensated 3-D subband coding of video," *IEEE Transactions on Image Processing*, vol. 3, no. 2, pp. 155–167, Feb 1999.

[CW04] P. Chen and J. W. Woods, "Bidirectional MC-EZBC with lifting implementation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 10, pp. 1183–1194, 2004.

[CXL01] L. Z. Cheng, H. Xu, and Y. Luo, "Integer Discrete Cosine Transform and its Fast Algorithm," *IEEE Electronic Letters*, vol. 37, no. 1, pp. 64–65, Jan 2001.

[CYC90] M. Chan, Y. Yu, and A. Constantinides, "Variable size block matching motion compensation with applications to video coding," *IEE Proceedings*, vol. 137, no. 4, pp. 205–212, Aug 1990.

[CZL02] L. Z. Cheng, G. J. Zhong, and J. S. Luo, "New family of Lapped Biorthogonal Transform via Lifting Steps," *IEEE Proceedings on Vison, Image and Signal Processing*, vol. 149, no. 2, pp. 91–96, Apr 2002.

[Dau88] I. Daubechies, "Orthomormal bases of compactly supported wavelets," *Communications on Pure and Applied Mathematics*, vol. 41, pp. 909–996, 1988.

[DBBR07] O. Déforges, M. Babel, L. Bédat, and J. Ronsin, "Color LAR Codec: A Color Image Representation and Compression Scheme Based on Local Resolution Adjustment and Self-Extracting Region Representation," *IEEE Transactions on Circuits an Systems for Video Technology*, vol. 17, no. 8, pp. 974–987, Aug 2007.

[DCM87] E. De Castro and C. Morandi, "Registration of translated and rotated images using finite Fourier transforms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 5, pp. 700–703, 1987.

[DCOY03] I. Drori, D. Cohen-Or, and H. Yeshurun, "Fragment-based image completion," in *Proceedings of International Conference on Computer Graphics and Interactive Techniques*, 2003, pp. 303–312, San Diego, CA, USA.

[DH98] A. Deever and S. Hemami, "Dense motion field reduction for motion estimation," in *Proceedings of Asilomar Conference on Signals, Systems and Computers (ACSSC'98)*, 1998, Pacific Grove, CA, USA.

[DH04] A. Dumitras and B. G. Haskell, "An encoder-decoder texture replacement method with application to content-based movie coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 6, pp. 825–840, Jun 2004.

[DHGF10] A. Drémeau, C. Herzet, C. Guillemot, and J.-J. Fuchs, "Sparse Optimization with Directional DCT Bases for Image Compression," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'10)*, 2010, pp. 1290–1293, Dallas, TX, USA.

[DMA97] G. M. Davis, S. Mallat, and M. Avellaneda, "Adaptive Greedy Approximations," *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[Don99] D. L. Donoho, "Wedgelets: Nearly Minimax Estimation of Edges," *The Annals of Statistics*, vol. 27, no. 3, pp. 859–897, Jun 1999.

[Don00] ——, "Orthonormal ridgelets and linear singularities," *SIAM Journal on Matehmatical Analysis*, vol. 31, no. 5, pp. 1062–1099, Apr 2000.

[DR99] O. Déforges and J. Ronsin, "Locally Adaptive Resolution method for progressive still image coding," in *Proceedings of the 5th International Symposium on Signal Processing and its Applications (ISSPA'99)*, Aug 1999, pp. 825–829, Brisbane, Australia.

[DT03] W. Dai and T. D. Tran, "Regularity-Constrained Pre and Post-Filtering for Block DCT-Based Systems," *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2568–2581, Oct 2003.

[Dud96] M. Dudon, "Modélisation du mouvement par treillis actifs et méthodes d'estimation associéees. Application au codage de séquences d'images," Ph.D. dissertation, Université de Rennes I, Rennes, France, Dec 1996.

[Duf94] F. Dufaux, "Multigrid Block Matching motion estimation for generic video coding," Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne, 1994.

[DV01] M. N. Do and M. Vetterli, "Pyramidal Directional Filter Banks and Curvelets," *Proceedings of IEEE International Conference on Image Processing (ICIP'01)*, vol. 3, pp. 158–161, Oct 2001.

[DV03] ——, "The finite ridgelet transform for image representation," *IEEE Transactions on image Processing*, vol. 12, pp. 16–28, Jan 2003.

[DV05] ——, "The Contourlet Transform: An Efficient Directional Multiresolution Image Representation," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2091–2106, Dec 2005.

[DVKG+00] N. Damera-Venkata, T. D. Kite, W. S. Geisler, B. L. Evans, and A. C. Bovik, "Image Quality Assessment Based on a Degradation Model," *IEEE Transactions on Image Processing*, vol. 9, no. 4, pp. 636–650, Apr 2000.

[DWL04] W. Ding, F. Wu, and S. Li, "Lifting-based Wavelet transform with directionally spatial prediction," in *Proceedings of Picture Coding Symposium*, Dec 2004, pp. 483–488, San Francisco, CA, USA.

[DWW+07] W. Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive Directional Lifting Based Wavelet Transform for Image Coding," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 416–427, Feb 2007.

[EDM+08] M. Eisemann, B. D. Decker, M. Magnor, P. Bekaert, E. de Aguiar, N. Ahmed, C. Theobalt, and A. Sellent, "Floating textures," in *(Proceedings of Eurographics (EG'08)*, vol. 27, no. 2, Apr 2008, pp. 409–418.

[EF05] A. Efros and W. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of SIGGRAPH'05*, 2005.

[EJN+06] K. Egiazarian, J.Astola, N.Ponomarenko, V.Lukin, F.Battisti, and M.Carli, "New full-reference quality metrics based on HVS," in *Proceedings of the Second International Workshop on Video Processing and Quality Metrics*, 2006, Scottsdale, AZ, USA.

[Eli75] P. Elias, "Universal codeword sets and representations of the integers," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 194–203, Mar 1975.

[Enk88] W. Enkelmann, "Investigation of Multigrid Algorithms for the Estimation of Optical Flow Fields in Image Sequences," *Computer Vision Graphics and Image Processing*, vol. 43, no. 2, pp. 150–177, Aug 1988.

[ESQD05] M. Elad, J. L. Starck, P. Querre, and D. L. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)," *Applied and Computational Harmonic Analysis*, vol. 19, no. 3, pp. 340–318, 2005.

[FBDC07] E. Flécher, M. Babel, O. Déforges, and V. Coat, "LAR Video: Hierarchical Representation for Low Bit-Rate Color Image Sequence Coding," in *Proceedings of Picture Coding Symposium (PCS'07)*, Nov 2007, Lisboa, Portugal.

[Fie93] D. J. Field, "Scale-invariance and self-similar "wavelet" transforms: Ananalysis of natural scenes and mammalian visual systems," in *Wavelets, fractals and Fourier transforms: New Developments and New Applications*. Clarendon Press, Oxford University, 1993, pp. 151–193.

[Fis92] Y. Fisher, "Fractal Image Compression," in *SIGGRAPH'92 Course notes*, 1992.

[FKE07] A. Foi, V. Katkovnik, and K. Egiazarian, "Pointwise Shape-Adaptive DCT for Hight Quality Denoising and Deblocking of Grayscale and Color Images," *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1395–1411, May 2007.

[FL88] O. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 2, no. 3, pp. 485–508, 1988.

[Fog91] S. V. Fogel, "The Estimation of Velocity Vector Fields from Time-Varying Image Sequences," *Computer Vision Graphics and Image Processing*, vol. 53, no. 3, pp. 253–287, May 1991.

[FS98] V. Fotopoulos and A. N. Skodras, "sMAE: An improved block matching criterion," in *Proceedings of the 5th International Conference on Electronics, Circuits and Systems (ICECS'98)*, vol. 3, Sep 1998, pp. 519–522, Lisboa, Portugal.

[Gal03] M. Galabov, "Fractal Image Compression," in *Proceedings of International Conference on Computer Systems and Technologies (CompSysTech'03)*, Jun 2003, pp. 320–326, Rousse, Bulgaria.

[Gam92] J.-P. Gambotto, "A region-based spatio-temporal segmentation algorithm," in *Proceedings of International Conference on Pattern Recognition (ICPR'92)*, vol. 3, no. 30, Aug 1992, pp. 189–192, The Hague, The Netherlands.

[GDZ00] X. Q. Gao, C. J. Duanmu, and C. R. Zou, "A multilevel successive elimination algorithm for block matching motion estimation," *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 201–504, Mar 2000.

[GGR95] I. Gorodnitsky, J. George, and B. Rao, "Neuromagnetic Source Imaging with FOCUSS: a Recursive Weighted Minimum Norm Algorithm," *Journal of Electroencephalography and Clinical Neurophysiology*, vol. 95, pp. 231–251, Oct 1995.

[Gha90] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Transactions on Communications*, vol. 38, no. 7, pp. 950–953, Jul 1990.

[Gil88] M. Gilge, "A high quality videophone coder using hierarchical motion estimation and structure coding of the prediction error," *Proceedings of the SPIE Visual Communications and Image Processing*, vol. 1001, no. 2, pp. 864–874, 1988.

[Gir87] B. Girod, "The efficiency of motion compensated prediction for hybrid video coding of video sequences," *IEEE Journal on Selected Areas in Communications*, vol. 5, pp. 1140–1154, Aug 1987.

[GM84] A. Grossmann and J. Morlet, "Decomposition of Hardy Functions into Square Integrable Wavelets of Constant Shape," *SIAM Journal on Mathematics Analysis*, vol. 15, no. 4, pp. 723–726, 1984.

[GM01] F. Galpin and L. Morin, "Computed 3D models for very low bitrate video coding," in *SPIE Proceedings of Conference on Visual Communications and Image Processing*, vol. 4310, Jan 2001, San Jose, CA, USA.

[GM02] ——, "Sliding adjustment for 3D video representation," *EURASIP Journal on Applied Signal Processing*, vol. 1, pp. 1088–1101, Jan 2002.

[GMR96] R. P. G.R. Martin and I. Rhee, "Variable size block matching estimation with minimal error," in *Proceedings of SPIE Conference on Digital Video Compression: Algorithms and Technologies*, vol. 2668, Feb 1996, pp. 324–333, San Jose, CA, USA.

[Gol66] S. W. Golomb, "Run-length encodings," *IEEE Transaction on Information Theory*, vol. 12, no. 3, pp. 399–401, 1966.

[Gra03] C. Grava, "Compensation de mouvements par réseaux neuronaux cellulaires. Application en imagerie médicale." Ph.D. dissertation, Institut National des Sciences Appliquées de Lyon, February 2003.

[Gué09] J. Guédon, *The Mojette transform: theory and applications*. University of Nantes, France: Hardback, Feb 2009.

[GWL08] Y. Guo, Y.-K. Wang, and H. Li, "Priority-based template matching intra prediction," in *Proceedings of International Conference on Multimedia & Expo (ICME'08)*, Apr 2008, pp. 1117–1120.

[HB93] F. Heitz and P. Bouthémy, "Multimodal estimation of discontinuous of the optical flow using markov random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 12, pp. 1217–1232, Dec 1993.

[HBH+89] P. J. Hurt, J. R. Bergen, R. Hingorani, R. Kolczinski, W. A. Lee, A. Leung, J. Lubin, and H. Shvaytser, "Object tracking with a moving camera, an application of dynamic motion analysis," in *Proceedings of IEEE Workshop on Visual Motion*, Mar 1989, pp. 2–12, Irvine, CA, USA.

[HBL+02] A. P. Hekstra, J. G. Beerends, D. Ledermann, F. E. de Caluwe, S. Kohler, R. H. Koenen, S. Rihs, M. Ehrsam, and D. Schlauss, "PVQM - A perceptual video quality measure," *Signal Processing: Image Communication*, vol. 17, no. 10, pp. 781–798, 2002.

[HBS84] B. L. Hinman, J. G. Bernstein, and D. H. Staelin, "Short-space Fourier transform image processing," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP'84)*, vol. 4, no. 8, 1984, pp. 1–4, San Diego, CA, USA.

[Hee90] J. Heel, "Direct Estimation of Structure and Motion from Multiple Frames," MIT AI LAB, Cambridge, MA, USA, Tech. Rep. 1190, 1990.

[HFB04] J. Huart, G. Foret, and P. Bertolino, "Extraction d'objets en mouvement par pyramide locale," in *Proceedings of CORESA 2004*, May 2004.

[HH94] C.-L. Huang and C.-Y. Hsu, "A new motion compensation method for image sequence coding using hierarchical grid interpolation," *IEEE Transactions on circuits and systems for video technology*, vol. 1, pp. 44–52, Feb 1994.

[Hil84] E. C. Hildreth, *Measurement of Visual Motion*. MIT Press Cambridge, MA, USA, 1984.

[Hil86] ——, "Computing the velocity field along contours," in *Proceedings of the ACM SIGGRAPH/SIGART interdisciplinary workshop on Motion: representation and perception*. New York, NY, USA: Elsevier North-Holland, Inc., 1986, pp. 121–127, Toronto, Ontario, Canada.

[HJJ79] D. N. Hein and H. W. Jones Jr., "Conditional replenishment using motion prediction," in *Proceedings of the Seminar Applications of digital image processing III*. SPIE, Aug 1979, pp. 268–277, San Diego, CA, USA.

[HKFT00] M. Harada, T. Kimoto, T. Fujii, and M. Tanimoto, "Fast calculation of IFS parameters for fractal image coding," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 4067, May 2000, pp. 457–464.

[HM99] P. Hosur and K. Ma, "Motion vector field adaptive fast motion estimation," in *Proceedings of 2nd International Conference on Information, Communications and Signal Processing (ICICS'99)*, Dec 1999, Singapore, Singapore.

[HM01] Z. He and S. Mitra, "ρ-domain bit allocation and rate control for real time video coding," in *Proceedings of International Conference on Image Processing*, vol. 3, 2001, pp. 546–549.

[HM03] D. Y. Huang and R. Ma, "Integer Fast Modified Cosine Transform," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'03)*, vol. 2, Jul 2003, pp. 729–732, Baltimore, MD, USA.

[HMCP01] G. Heising, D. Marpe, H. Cycon, and A. Petukhov, "Wavelet-based very low bit-rate video coding using image warping and overlapped block motion compensation," *IEEE Proceedings - Vision, Image, and Signal Processing*, vol. 148, no. 2, pp. 93–101, 2001.

[HO07] Y.-S. Ho and K.-J. Oh, "Overview of Multi-view Video Coding," in *Proceedings of 14th International Workshop on Systems, Signals and Image Processing*, Jun 2007, pp. 5–12.

[HP01] S.-C. Han and C. I. Podilchuk, "Video compression with dense motion fields," *IEEE Transactions on Image Processing*, vol. 10, no. 11, pp. 1605–1612, Nov 2001.

[HS81] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[Huf52] D. A. Huffman, "A Method for the Construction of Minimum-Redundancy Codes," in *Proceedings of the I. R. E.*, Sep 1952, pp. 1098–1102.

[Hut81] J. E. Hutchinson, "Fractals and self similarity," *Indiana University Mathematical Journal*, vol. 30, pp. 713–747, 1981.

[HW65] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture in two non-striate visual areas (18 and 19) of the cat," *Journal of Neurophysiology*, vol. 28, pp. 229–289, 1965.

[HW88] B. K. P. Horn and E. J. Weldon, "Direct methods for recovering motion," *International Journal of Computer Vision*, vol. 2, no. 1, pp. 51–76, Jun 1988.

[HZ00] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[IM97] P. Ishwar and P. Moulin, "Switched Control Grid Interpolation for motion compensated video coding," in *Proceedings of International Conference on Image Processing (ICIP'97)*, vol. 3, 1997, pp. 650–653.

[IM00] ——, "On spatial adaptation of motion field smoothness in video coding," *IEEE Transactions on circuits and systems for video technology*, vol. 10, no. 6, pp. 980–989, Sep 2000.

[ISO93] ISO / IEC Std. 11 172-2 Video, *Coding of moving pictures and associated audio for digital storage media at up to about 1.5 MBit/s*, MPEG-1. 1993.

[ISO94a] ISO / IEC Std. 10 918, *Digital compression and coding of continuous-tone still images*, JPEG. 1994.

[ISO94b] ISO / IEC Std. 13 818-2 Video, *Generic coding of moving pictures and associated audio information*, MPEG-2. 1994.

[ISO00a] ISO / IEC Std. 14 496-2 Video, *Generic coding of audiovisual objects*, MPEG-4. 2000.

[ISO00b] ISO / IEC Std. 15 444, *JPEG2000 Image coding system*, JPEG 2000. 2000.

[ITN02] M. Ikehara, T. D. Tran, and T. Q. Nguyen, "A Family of Lapped Regular Transforms With Integer Coefficients," *IEEE Transactions on Signal Processing*, vol. 50, no. 4, pp. 834–841, Apr 2002.

[ITU88] ITU-CCITT Std., *Codecs for videoconferencing using primary digital group transmission*, recommendation H.120. Nov 1988.

[ITU90] ITU-T Std., *Video codec for audiovisual services at px 64kbit/s*, recommendation H.261. 1990.

[ITU94] ITU-T Std., *Generic coding of moving pictures and associated audio information*, recommendation H.262. 1994.

[ITU95] ITU-T Std., *Video coding for low bit-rate communication*, recommendation H.263. 1995.

[ITU04] ITU-T Std., *Video coding for low bit-rate communication - Annex X : profiles and level definitions*, recommendation H.263. 2004.

[ITU09] ITU-T Std., *JPEG XR Image coding system - Image coding specification*, recommendation T.832. Mar 2009.

[Jel68] F. Jelinek, *Probabilistic Information Theory*. McGraw Hill, New York, NY, USA, 1968.

[JJ81] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol. 29, no. 12, pp. 1799–1808, Dec 1981.

[Jon77] H. W. J. Jones, "A conditional replenishment Hadamard video compressor," in *Proceedings of the International Optical Computing Conference on Applications of digital image processing*, SPIE. Bellingham, Wash, Aug 1977, pp. 91–98.

[JRG92] F. Jaureguizar, J. I. Ronda, and N. Garcia, *Statistical analysis of the motion compensated hybrid transform encoding of HDTV signals*. L. Chiarlione Elsevier Science Publishers B.V., 1992.

[JS94] B. Jawerth and W. Sweldens, "An overview of Wavelet Based Multiresolution Analysis," *SIAM Review*, vol. 36, no. 3, pp. 377–412, 1994.

[JVT03] JVT (ITU-T & ISO-IEC) Std. 14 496-10, *Advanced video coding for generic audiovisual services*, recommendation H.264. May 2003.

[JVT05] JVT (ITU-T & ISO-IEC) Std. 14 496-10, *Advanced video coding for generic audiovisual services*, recommendation H.264, Amendment 1, Fidelity Range EXTensions. Mar 2005.

[JVT07a] JVT (ITU-T & ISO-IEC) Std. 14 496-10, *Advanced video coding for generic audiovisual services*, recommendation H.264, Amendment 2, New profiles for professional applications. Apr 2007.

[JVT07b] JVT (ITU-T & ISO-IEC) Std. 14 496-10, *Advanced video coding for generic audiovisual services*, recommendation H.264, Amendment 3, Scalable Video Coding (Annex G). Nov 2007.

[JVT09] JVT (ITU-T & ISO-IEC) Std. 14 496-10, *Advanced video coding for generic audiovisual services*, recommendation H.264, Amendment 4, Multiview Video Coding (Annex H). Mar 2009.

[Kal60] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[KaMK85] M. Kunt and A. I. anf Michel Kocher, "Second-Generation Image-Coding Techniques," *Proceeding of IEEE*, vol. 73, no. 4, pp. 549–574, Apr 1985.

[Kar47] K. Karhunen, "Über lineare Methoden in der Wahrscheinlichkeitsrechnung," *Ann. Acad. Sci. Fennicae. Ser. A. I. Math.-Phys.*, vol. 37, pp. 1–79, 1947.

[KDST05] N. Kroupis, M. Dasygenis, D. Soudris, and A. Thanailakis, "A Modified Spiral Search Algorithm and its Embedded Hardware Implementation," *International Journal of Information Technology*, vol. 2, no. 3, pp. 199–205, Dec 2005.

[Kha89] S. A. Khakoo, "Signature-based search algorithm," in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP'89)*, vol. 3, May 1989, pp. 1874–1877, Glasgow, Scotland, UK.

[KIH+81] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proceedings of National Telecommunications Conference*, ser. 3, vol. 4, no. 5, 1981, pp. 1–5.

[Kim94] J. Kim, "3-D Kalman Filter for video and motion estimation," Ph.D. dissertation, Rensselaer Polytechnic Institute, Troy, NY, USA, 1994.

[Kom97] J. Kominek, "Advances in fractal compression for multimedia applications," *Multimedia Systems*, vol. 5, no. 4, pp. 255–270, 1997.

[Kor67] C. M. Kortman, "Redundancy reduction - A practical method of data compression," in *Proceedings of the IEEE*, vol. 55, no. 3, Mar 1967, pp. 253–265.

[KS80] R. Kindermann and J. L. Snell, *Markov Random Fields and Their Applications*. American Mathematical Society, 1980.

[KS98] P. Kauff and K. Schuur, "Shape-Adaptive DCT with Block-based DC Separation and ΔDC Correction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 237–242, Jun 1998.

[KV88] G. Karlsson and M. Vetterli, "Three Dimensional Subband Coding of Video," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'88)*, 1988, pp. 1100–1103.

[KWT88] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes, active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, Jan 1988.

[KXP97] B.-J. Kim, Z. Xiong, and W. A. Pearlman, "An emmbeded Wavelet Video Coder Using Three-Dimensional Set Partitioning in Hierarchical Trees (SPIHT)," in *Proceedings of IEEE Data Compression Conference (DCC'97)*, 1997, pp. 221–260.

[KYKR99] S. D. Kim, J. Yi, H. M. Kim, and J. B. Ra, "A deblocking filter with two separate modes in block-based video coding," *IEEE Transactions on Circuits an Systems for Video Technology*, vol. 9, pp. 156–160, Feb 1999.

[LB10] C. Li and A. C. Bovik, "Content-weighted video quality assessment using a three-component image model," *Journal of Electronic Imaging*, vol. 19, pp. 011 003, 1–8, 2010.

[LCL+97] M.-C. Lee, W. G. Chen, C. L. Lin, C. Gu, T. Markok, S. I. Zabinsky, and R. Szeliski, "A layered video object coding system using sprite and affine motion model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 130–145, Feb 1997.

[Lec99] P. Lechat, "Représentation et codage de séquences vidéo par maillages 2D déformables," Ph.D. dissertation, Université de Rennes I, Rennes, France, Dec 1999.

[Lee95] J. Lee, "Optimal quadtree for variable block size motion estimation," in *Proceedings of the International Conference on Image Processing (ICIP'95)*, vol. 3, Oct 1995, pp. 480–483.

[LFWLZ03] L. Luo, F. Feng Wu, S. Li, and Z. Zhuang, "Advanced lifting-based motion threading technique for the 3D wavelet video coding," *SPIE Visual Communication on Image Processing*, vol. 5150, pp. 707–718, 2003.

[LG99] S. Lin and O. Guleryuz, "Dense Motion Fields for Digital Video Processing and Compression," in *Proceedings of International Conference on Image Processing (ICIP'99)*, vol. 3, 1999, pp. 737–741.

[LG08] B. Le Guen, "Adaptation du contenu spatio-temporel des images pour un codage par ondelettes," Ph.D. dissertation, Université de Rennes I, Rennes, France, Feb 2008.

[Li07] X. Li, "Video Processing Via Implicit and Mixture Motion Models," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 8, pp. 953–963, Aug 2007.

[LJ84] G. G. Langdon Jr., "An introduction to arithmetic coding," *IBM Journal of Research and Development*, vol. 28, no. 2, pp. 135–149, Mar 1984.

[LK81] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," in *Proceedings of Imaging Understanding Workshop*, 1981, pp. 121–130.

[LK97] Y.-K. Lai and C.-C. J. Kuo, "Image quality measurement using the Haar wavelet," in *Proceedings of SPIE: Wavelet Applications in Signal and Image Processing V*, vol. 3169, Aug 1997, pp. 127–138, San Diego, CA, USA.

[LLL+01] L. Luo, J. Li, S. Li, Z. Zhuang, and Y.-Q. Zhang, "Motion-compensated wavelet and its application in video coding," in *Proceedings of IEEE International Conference on Multimedia and Expo*, 2001, pp. 481–484, Tokyo, Japan.

[Loè78] M. Loève, *Probability theory Vol. II, Graduate Texts in Mathematics*, 4th ed. Springer-Verlag, 1978, vol. 46.

[LP01] Y.-L. Lee and H. W. Park, "Loop filtering and post-filtering for low-bitrates moving pictures coding," *Signal Processing: Image Communication*, vol. 16, pp. 871–890, 2001.

[LRS98] P. Lechat, M. Ropert, and H. Sanson, "Hierarchical mesh-based motion estimation using a differential approach and application to video coding," in *Proceedings of European SIgnal Processing COnference (EUSIPCO'98)*, vol. 62, Sep 1998, pp. 2077–2080, Rhodes, Greece.

[LS95] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Transactions on Image Processing*, vol. 4, no. 1, pp. 105–107, Jan 1995.

[LSW07a] D. Liu, X. Sun, and F. Wu, "Edge-Based Inpainting and Texture Synthesis for Image Compression," in *Proceedings of International Conference on Multimedia and Expo (ICME'07)*, July 2007, pp. 1443–1446.

[LSW+07b] D. Liu, X. Sun, F. Wu, S. Li, and Y.-Q. Zhang, "Image Compression With Edge-Based Inpainting," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 10, pp. 1273–1287, Oct 2007.

[LT00] J. Liang and T. D. Tran, "Approximating the DCT with the lifting scheme: systematic design and applications," in *Proceedings of 34th IEEE International Conference on Information, Communications and Signal Processing (ICICS'01)*, vol. 1, Nov 2000, pp. 192–196, Pacific Grove, CA, USA.

[LT01] ——, "Fast multipliers approximations of the DCT with the lifting scheme," *IEEE Transactions on Signal Processing*, vol. 49, no. 12, pp. 3032–3044, Dec 2001.

[LT08] Z. Li and A. M. Tourapis, "An Estimation-Theoretic Interpretation of Video Rate Distortion Optimization with Lagrangian Formulation," in *Proceedings of the Data Compression Conference (DCC'08)*, 2008.

[LTT01] J. Liang, C. Tu, and T. D. Tran, "Fast Lapped Transforms via Time Domain Pre and Post-Processing," in *Proceedings of IEEE International Conference on Information, Communications and Signal Processing (ICICS'01)*, Oct 2001, Singapore, Singapore.

[MA88] W. Martin and J. K. Aggarwal, *Motion understanding, robot and human vision*. Kluwer Academic Publishers, 1988.

[MADB10] M. Moinard, I. Amonou, P. Duhamel, and P. Brault, "A Set of Template Matching Predictors for Intra Video Coding," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Mar 2010, pp. 1525–1530.

[Mal89a] S. Mallat, "Multiresolution Signal Decomposition: The Wavelet Representation," *Transaction of the American mathematical Society*, vol. 315, pp. 69–87, Sep 1989.

[Mal89b] ——, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, no. 7, pp. 674–693, Jul 1989.

[Mal90] H. S. Malvar, "Lapped Transforms for Efficient Transform/Subband Coding," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 6, pp. 969–978, Jun 1990.

[Mal98] ——, "Biorthogonal and Nonuniform Lapped Transforms for Transform Coding with Reduced Blocking and Ringing Effect," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 1043–1053, Apr 1998.

[Mal00] S. Mallat, *Une exploration des signaux en ondelettes*, E. diffusion, Ed. Ecole Polytechnique, 2000.

[Mar97] R. P. . M. S. . G. Martin, "Variable size block matching motion compensation for object-based video coding," in *Proceedings of IEEE 6th International Conference on Image Processing & its Applications*, 1997, pp. 56–60, Dublin, Ireland.

[Mar00] G. Marquant, "Représentation par maillage adaptatif déformable pour la manipulation et la communication d'objets vidéo," Ph.D. dissertation, Université de Rennes I, Rennes, France, Dec 2000.

[Mas02] S. Masnou, "Disocclusion : a variational approach using level lines," *IEEE Transactions on Image Processing*, vol. 11, no. 2, pp. 68–76, 2002.

[MC97] F. G. Meyer and R. R. Coifman, "Brushlets: A Tool for Directional Image Analysis and Image Compression," *Applied and Computational Harmonic Analysis*, vol. 4, no. 2, pp. 147–187, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/B6WB3-45MH2TF-8/2/26f109da3fd9f249aa4f21ceb5482349

[Mey88] Y. Meyer, "Construction de bases orthonormées d'ondelettes," *Revista Matematica Iberoamericana*, vol. 4, no. 1, pp. 31–39, 1988.

[Mey90] ——, *Ondelettes et Opérateurs*. Hermann, Paris, 1990, vol. 1.

[MF06] S. Maria and J. J. Fuchs, "Application of the Global Matched Filter to STAP data, an efficient algorithmic approach," in *Procedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP'06)*, 2006.

[MFGT07] A. Martin, J.-J. Fuchs, C. Guillemot, and D. Thoreau, "Sparse Representation for Image Prediction," in *Proceedings of 15th European Signal Processing Conference (EUSIPCO07)*, Sep 2007, Poznan, Poland.

[MHW03] D. Marpe, S. H., and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, vol. 13, no. 7, pp. 620–636, 2003.

[MJZ06] V. Muñoz-Jimenez and A. Zergaïnoh, "Etude de maillage rectangulaire déformable pour l'estimation de mouvement dans des séquences vidéo," in *Journées d'études et d'échanges sur la COmpression et Représentation des Signaux Audiovisuels (CORESA'06)*, Nov 2006, pp. 318–323, Cane, France.

[MKW97] P. Moulin, R. Krishnamurthy, and J. W. Woods, "Multiscale modeling and estimation of motion fields for video coding," *IEEE Transactions on Image Processing*, vol. 6, pp. 1606–1620, Dec 1997.

[MS74] J. L. Mannos and D. J. Sakrison, "The Effects of a Visual Fidelity Criterion on the Encoding of Images," *IEEE Transactions on Information Theory*, vol. 20, no. 4, pp. 525–535, 1974.

[MS89] H. S. Malvar and D. H. Staelin, "The LOT: transform coding without blocking effects," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 4, pp. 553–559, Apr 1989.

[MV07] E. Malis and M. Vargas, "Deeper understanding of the homography decomposition for vision-based control," Insitut National de Recherche en Informatique et en Automatique, Sophia Antipolis, France, Tech. Rep. 6303, Sep 2007.

[MZ93] S. Mallat and Z. Zhang, "Matching Pursuits with time frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, p. 12, Dec 1993.

[Nag83] H.-H. Nagel, "Constraints for the estimation of displacement vector fields from image sequences," in *Proceedings of International Joint Conference on Artificial Intelligence*, vol. 2, no. 10, 1983, pp. 945–951.

[NH87] S. Negahdaripour and B. K. P. Horn, "Direct Passive Navigation," *IEEE Transactions on Pattern Analysis*, vol. 9, no. 1, pp. 168–176, Jan 1987.

[NH91] Y. Nakaya and H. Harashima, "An iterative motion estimation method using triangular patches for motion compensation," in *Proceedings of SPIE's Visual Communications and Image Processing*, vol. 1605, 1991, pp. 546–557.

[Nic92] H. Nicolas, "Hiérarchie des modèles de mouvement et méthodes d'estimation associées, application au codage des séquences d'images," Ph.D. dissertation, Université de Rennes I, Rennes, France, 1992.

[NKPS95] K. M. Nam, J.-S. Kim, R.-H. Park, and Y. S. Shim, "A fast hierarchical motion vector estimation algorithm using mean pyramid," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 4, pp. 344–351, Aug 1995.

[NNHSW06] P. Ndjiki-Nya, T. Hinz, C. Stüber, and T. Wiegand, "A Content-based Video Coding Approach for Rigid and Non-Rigid Textures," in *Proceedings of IEEE International Conference on Image Processing (ICIP'06)*, Oct 2006, pp. 3167–3172, Atlanta, GA, USA.

[NNHW07] P. Ndjiki-Nya, T. Hinz, and T. Wiegand, "Generic and Robust Video Coding with Texture Analysis and Synthesis," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'07)*, Jul 2007, pp. 1447–1450, Beijing, China.

[NNSHW05] P. Ndjiki-Nya, A. Smolic, T. Hinz, and T. Wiegand, "A Generic and Automatic Content-Based Approach for Improved H.264/MPEG4-AVC Video Coding," in *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, Sep 2005, Genova, Italy.

[NO92] S. Nogaki and M. Ohta, "An overlapped block motion compensation for high quality motion picture coding," in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 5, Apr 1992, pp. 427–440.

[NR84] A. N. Netravali and J. D. Robbins, "Motion compensated television coding: Part I," *Bell System Technical Journal*, vol. 58, pp. 631–670, 1984.

[NW96] T. Naveen and J. W. Woods, "Subband finite state scalar quantization," *IEEE Transactions on Image Processing*, vol. 5, no. 1, pp. 150–155, Jan 1996.

[NZ97] R. Neff and A. Zakhor, "Very low bit-rate video coding based on matching pursuit video coder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 158–171, Feb 1997.

[OAL01] A. T. Oscar, O. C. Au, and M. L. Liou, "Predictive Motion Vector Field Adaptive Search Technique (PMVFAST) - Enhancing Block Based Motion Estimation," *Visual Communications and Image Processing*, vol. 4130, pp. 883–892, 2001.

[OB94] J.-M. Odobez. and P. Bouthémy, "Robust multiresolution estimation of parametric motion models in complex image sequences," in *Proceedings of 7th European Conf. on Signal Processing, Eusipco'94*, 1994, Edinburgh, Scotland, UK.

[OB95] ——, "Robust multiresolution estimation of parametric motion models," *Journal of Visual Communication and Image Representation*, vol. 6, no. 4, pp. 348–365, Dec 1995.

[OBMC01] M. M. Oliveira, B. Bowen, R. McKenna, and Y.-S. Chang, "Fast Digital Image Inpainting," in *Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP'01)*. ACTA Press, Sep 2001, pp. 261–266, Marbella, Spain.

[Ohm93] J.-R. Ohm, "Advanced Packet-Video Coding Based on Layered VQ and SBC Techniques," *IEEE Transactions on Circuits an Systems for Video Technology*, vol. 3, no. 3, pp. 208–221, Jun 1993.

[Ohm94] J. R. Ohm, "Three-dimensional subband coding with motion compensation," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 559–571, Sep 1994.

[OPT08] OPTICOM, "PEVQ - Perceptual Evaluation of Video Quality," Mar 2008.

[OS94] M. T. Orchard and G. J. Sullivan, "Overlapped Block Motion Compensation: an estimation theoretic approach," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 693–699, Sep 1994.

[Pau06] G. Pau, "Ondelettes et décompositions spatio-temporelles avancées ; application au codage vidéo scalable," Ph.D. dissertation, ENST Paris, Juin 2006.

[PBC07] S. Péchard, D. Barba, and P. L. Callet, "Video Quality Model based on a spatiotemporal features extraction for H.264-coded HDTV sequences," in *Proceedings of PCS*. IEEE, Nov 2007, Lisboa, Portugal.

[Pea01] K. Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space," *Philosophical Magazine*, vol. 2, no. 6, pp. 559–572, 1901.

[Pet99] N. Peterfreund, "Robust tracking of position and velocity with Kalman snakes," *IEEE transaction on pattern analysis and machine intelligence*, vol. 21, no. 6, pp. 564–569, July 1999.

[Pey05] G. Peyré, "Géométrie multi-échelles pour les images et les textures," Ph.D. dissertation, Ecole Polytechnique, France, Dec 2005.

[PHS87] A. Puri, H.-M. Hang, and D. Schilling, "An efficient block-matching algorithm for motion-compensated coding," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'87)*, Apr 1987, pp. 1063–1066.

[PJB87] J. P. Princen, A. W. Johnson, and A. B. Bradley, "Subband/transform coding using filter bank designs based on time domain aliasing cancellation," in *IEEE Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 12. IEEE, 1987, pp. 2161–2164.

[PJF95] C. I. Podilchuk, N. S. Jayant, and N. Farvadin, "Three Dimensional Subband Coding of Video," *IEEE Transactions on Image Processing*, vol. 4, no. 2, pp. 125–139, Feb 1995.

[PL99] S. Pateux and C. Labit, "Rate-distortion optimized region-based video coder," in *Proceedings of ICIP'99*, Oct 1999, Kobe, Japan.

[PM96] L.-M. Po and W. C. Ma, "A novel four step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 313–319, Jun 1996.

[PM05a] E. L. Pennec and S. Mallat, "Sparse Geometric Image Representations With Bandelets," *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 423–438, Apr 2005.

[PM05b] G. Peyré and S. Mallat, "Discrete Bandelets with Geometric Orthogonal Filters," in *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, vol. 1, Sept 2005, pp. 65–68, Genova, Italy.

[PM05c] ——, "Surface Compression with Geometric Bandelets," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 601–608, Jul 2005.

[PMCM01] S. Pateux, G. Marquant, and D. Chavira-Martinez, "Object mosaicking via meshings and cracklines techniques. Application to low bitrate video coding." in *Proceedings of Picture Coding Symposium (PCS'01)*, Apr 2001, Seoul, Korea.

[PPB01] B. Pesquet-Popescu and V. Bottreau, "Three-dimensional lifting schemes for motion compensated video compression," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP'01)*, vol. 3, May 2001, pp. 1793–1796, Salt-Lake City, UT, USA.

[Pre06] M. Pressigout, "Approches hybrides pour le suivi temps-réel d'objets complexes dans les séquences vidéo," Ph.D. dissertation, Université de Rennes I, Rennes, France, Dec 2006.

[PS00] J. Portilla and E. P. Simoncelli, "A Parametric Texture Model based on Joint Statistics of Complex Wavelet Coefficients," *International Journal of Computer Vision*, vol. 40, no. 10, pp. 49–71, Oct 2000.

[PT03a] G. Plonka and M. Tasche, "Integer DCT-II by lifting steps," in *International Series of Numerical mathematics*. W. Haussmann, K. Jetter, M. Reimer, J. Stöckler, Birkäuser Basel, 2003, vol. 145, pp. 235–252.

[PT03b] ——, "Invertible Integer DCT Algorithms," *Applied and Computational Harmonic Analysis*, vol. 15, no. 1, pp. 70–88, Jul 2003.

[PW83] S. M. Pandit and S.-M. Wu, *Time Series and System Analysis with Applications*. John Wiley & Sons, 1983.

[PW04] M. Pinson and S. Wolf, "A New Standardized Method for Objectively Measuring Video Quality," *IEEE Transactions on Broadcasting*, vol. 50, no. 3, pp. 312–322, Sep 2004.

[PXC02] S. Prince, K. Xu, and A. Cheok, "Augmented Reality Camera Tracking with Homographies," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 39–45, Nov 2002.

[Péc08] S. Péchard, "Qualité d'usage en télévision haute définition : évaluations subjectives et métriques objectives," Ph.D. dissertation, Ecole Polytechnique de l'Université de Nantes, Nantes, France, Oct 2008.

[Rad17] J. Radon, "Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten," in *Berichte über die Verhandlungen der Sächsische Akademie der Wissenschaften*, vol. 69, 1917, pp. 262–277.

[Rao97] B. Rao, "Analysis and Extensions of the FOCUSS Algorithm," in *Proceedings of the IEEE Asilomar Conference on Circuits, Systems, and Computers (ACCSC'96)*, vol. 2, 1997, pp. 1218–1240.

[RCN97] J. Ribas-Corbera and D. Neuhoff, "On the optimal block size for block-based, motion compensated video coders," *SPIE Proceedings on Visual Communications and Image Processing*, vol. 3024, pp. 1132–1143, Feb 1997.

[Ric03] I. E. Richardson, *H.264 and MPEG-4 Video Compression*, Wiley, Ed. Wiley, 2003.

[Ris76] J. Rissanen, "Generalized Kraft Inequality and Arithmetic Coding," *IBM Journal of Research and Development*, vol. 20, no. 3, pp. 198–203, May 1976.

[RKDM06] W. Rajkumar, M. V. Kulkarni, M. L. Dhore, and S. N. Mali, "Fractal image compression performance synthesis through HV partitioning," in *Proceedings of International Conference on Advanced Computing and Communications (ADCOM'06)*, Dec 2006, pp. 636–637, Surathkal, India.

[RLJ79] J. J. Rissanen and G. G. Langdon Jr., "Arithmetic coding," *IBM Journal of Research and Development*, vol. 23, no. 2, pp. 149–162, Mar 1979.

[RM04] M. Rowan and F. D. Maire, "An efficient multiple object vision tracking system using bipartite graph matching," in *Proceedings of FIRA Robot World Congress*. BEXCO, Oct 2004, Busan, Korea.

[RM06] G. Raja and M. J. Mirza, "In-loop Deblocking Filter for H.264/AVC Video," in *Proceedings of the 5th WSEAS International Conference on Signal Processing, Robotics and Automation*, 2006, pp. 235–240, Madrid, Spain.

[Rob08] A. Robert, "Transformées orientées par bloc pour le codage vidéo hybride," Ph.D. dissertation, Ecole Nationale Supérieure de Télécommunications (ENST) de Paris, Paris, France, Feb 2008.

[RSB03] S. D. Rane, G. Sapiro, and M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," *IEEE Transactions on Image Processing*, vol. 12, no. 3, pp. 296–303, Mar 2003.

[RWB02] J. K. Romberg, M. B. Wakin, and R. G. Baraniuk, "Multiscale Wedgelet Image Analysis: Fast Decomposition and Modeling," in *Proceedings of IEEE International Conference on Image Processing (ICIP'02)*, vol. 3, Jun 2002, pp. 585–588, Rochester, NY, USA.

[RY90] K. Rao and P. Yip, *Discrete Cosine Transform : Algorithms, Advantages, Applications*. Academic Press Professional, Inc., 1990.

[RZ52] J. R. Ragazzini and L. A. Zadeh, "The analysis of sampled-data systems," *Trans. Am. Inst. Elec. Eng*, vol. 71, pp. 225–234, 1952.

[SA96] H. S. Sawhney and S. Ayer, "Compact representation of videos through dominant and multiple motion estimation," *IEEE transactions on PAMI*, vol. 18, no. 8, pp. 814–830, Aug 1996.

[Sab84] S. Sabri, "Movement compensated interframe prediction for NTSC color TV signals," *IEEE Transaction on Communications*, vol. 32, pp. 954–967, 1984.

[San95] H. Sanson, "Vers une méthodologie pour l'identification paramétrique robuste du mouvement de régions par optimisation non-linéaire," in *Proceedings of GRETSI Symposium on Signal and Image Processing*, Sep 1995, Juan-Les-Pins, France.

[SB91a] G. J. Sullivan and R. L. Baker, "Motion compensation for video compression using control grid interpolation," in *Proceedings of International Conference on Speech and Signal Processing*. IEEE, 1991, pp. 2713–2716.

[SB91b] ——, "Rate-distortion optimized motion compensation for video compression using fixed or variable sized blocks," *Global Telecommunication*, vol. 1, pp. 85–90, Dec 1991.

[SB94] ——, "Efficient Quadtree Coding of Images and Video," *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 327–331, May 1994.

[SB02] G. Simon and M.-O. Berger, "Pose Estimation for Planar Structures," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 46–53, Nov 2002.

[SB06] H. Sheikh and A. Bovik, "Image information and visual quality," *IEEE Transactions on Image Processing*, vol. 15, no. 2, pp. 430–444, 2006.

[SBdV05] H. R. Sheikh, A. C. Bovik, and G. de Veciana, "An information fidelity criterion for image quality assessment using natural scene statistics," *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 2117–2128, 2005.

[SFZ00] G. Simon, A. Fitzgibbon, and A. Zisserman, "Markerless Tracking Using Planar Structures in the Scene," in *Proceedings of IEEE/ACM International Symposium on Augmented Reality*, Oct 2000, pp. 120–128.

[SGPK94] P. Salembier, C. Gu, M. Pardas, and M. Kunt, "Very low bit-rate video coding using morphological segmentation and contour/texture motion compensation," in *Proceedings of International Conference on Pattern Recognition (ICPR'94)*, ser. Speech and Signal Processing, vol. 3, 1994, pp. 25–29, Jerusalem, Israel.

[Sha48] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, July, October 1948.

[Sha93] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, Dec 1993.

[SHW03] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Transactions on Circuits an Systems for Video Technology*, vol. 13, no. 7, pp. 657–673, Jul 2003.

[SM95] T. Sikora and B. Makai, "Shape-Adaptive DCT for Generic Coding og Video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 1, pp. 59–62, Feb 1995.

[SM99] P. Salembier and F. Marquès, "Region-based representation of image and video: segmentation tools for multimedia services," *IEEE Transactions on circuits and systems for video technology*, vol. 9, no. 8, pp. 1147–1169, Dec 1999.

[SMP+97] P. Salembier, F. Marqués, M. Pardàs, R. Morros, I. Corset, S. Jennin, L. Bouchard, and B. Marcotegui, "Segmentation-based Video Coding System Allowing the Manipulation of Objects," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 60–73, Feb 1997.

[SMW07] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," *IEEE Transactions on Circuits an Systems for Video Technology (TCSVT)*, vol. 17, no. 9, pp. 1103–1120, Sep 2007.

[SP96] A. Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 1103–1120, Jun 1996.

[SP98] S. Servetto and C. Podilchuk, "Stochastic modeling and entropy constrained estimation of motion from image sequences," in *Proceedings of IEEE International Conference on Image Processing(ICIP'98)*, vol. 3, 1998, pp. 591–595.

[SS96] K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Transactions on Circuits an Systems for Video Technology*, vol. 6, no. 5, pp. 513–518, Oct 1996.

[ST01] A. Secker and D. Taubman, "Motion Compensated Highly Scalable Video Compression Using an Adaptive 3D Wavelet Transform Based on Lifting," *Proceedings of the IEEE International Conference on Image Processing (ICIP'01)*, vol. 2, pp. 1029–1032, Oct 2001.

[Sti94] C. Stiller, "Object-oriented video coding employing dense motion fields," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'94)*, Apr 1994, pp. 273–276, Adelaide, Australia.

[STL04] G. J. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range EXTensions," in *Proceedings of SPIE Conference on Applications of Digital Image Processing*, Aug 2004.

[Str99] G. Strang, "The Discrete Cosine Transform," *SIAM Review*, vol. 41, no. 1, pp. 135–147, 1999.

[Sul93] G. J. Sullivan, "Multi-hypothesis motion compensation for low bit-rate video coding," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, Apr 1993, pp. 427–440.

[SW98] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, pp. 74–90, 1998.

[Swe96] W. Sweldens, "The Lifting Scheme : a Custom-Design Construction of Biorthogonal Wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, no. 2, pp. 186–200, 1996.

[Swe97a] ——, "The Lifting Scheme: a Construction of Second Generation Wavelets," *SIAM Journal on Mathematics Analysis*, vol. 29, no. 2, pp. 511–546, 1997.

[Swe97b] ——, "Wavelets and the Lifting Scheme: a 5 Minutes Tour," *Journal of Applied Mathematics and Mechanics (JAMM)*, vol. 76, no. 2, pp. 41–44, 1997.

[SWL06] X. Sun, F. Wu, and S. Li, "Compression with Vision Technologies," in *Proceedings of Picture Coding Symposium (PCS'06)*, 2006.

[TAL02] A. Tourapis, O. Au, and M. Liou, "Highly efficient predictive zonal algorithms for fast block-matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 10, pp. 934–947, Oct 2002.

[Tau99] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Transactions on Image Processing*, vol. 9, no. 7, pp. 1158–1170, Jul 1999.

[TBS06] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra Prediction by Template Matching," in *Proceedings of IEEE International Conference on Image Processing (ICIP'06)*, Oct 2006, pp. 1693–1696, Atlanta, GA, USA.

[TEST96] C. Toklu, A. T. Erdem, M. I. Sezan, and A. M. Tekalp, "Tracking motion and intensity variations using hierarchical 2D mesh modelling for synthetic object transfiguration," *Graphical Models and Image Processing*, vol. 58, no. 6, pp. 553–573, Nov 1996.

[TH94] P. C. Teo and D. J. Heeger, "Perceptual image distortion," *Proceedings of SPIE*, vol. 2179, pp. 127–141, 1994.

[Tho87] G. Thomas, "Television motion measurement for DATV and other applications," BBC Research Department, Tech. Rep. BBC RD 1987/11, 11 1987.

[TLT03] T. D. Tran, J. Liang, and C. Tu, "Lapped Transform via Time-Domain Pre and Post-Filtering," *IEEE Transactions on Signal Porcessing*, vol. 54, no. 6, pp. 1557–1571, Jun 2003.

[TM01] D. Taubman and M. Marcellin, *JPEG2000: image compression, fundamentals, standards and practice*, 1st ed. Springer, Nov 2001.

[Tod96] D. Todorovic, "A gem from the past: Pleikart Stumpf's (1911) anticipation of the aperture problem, Reichardt detectors, and perceived motion loss at equiluminance," *Perception*, vol. 25, no. 10, pp. 1235–1242, July 1996.

[Tou02] A. M. Tourapis, "Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation," in *Proceedings of SPIE*, vol. 4671, no. 1069, Jan 2002, pp. 1069–1079, San Jose, CA, USA.

[Tra99] T. D. Tran, "A fast multiplierless block transform for image and video compression," in *Proceedings of IEEE International Conference on Image Processing (ICIP'99)*, vol. 3, Oct 1999, pp. 822–826.

[Tra00] ——, "The BinDCT: fast multiplierless approximation of the DCT," *IEEE Signals Processing Letters*, vol. 7, no. 6, pp. 141–144, Jun 2000.

[TT01] T. D. Tran and C. Tu, "Lapped Transform Based Video Coding," in *Proceedings of SPIE Applications of Digital Image Processing XXIV*, vol. 4472, Aug 2001, pp. 319–333, San Diego, CA, USA.

[TvdS02] D. Turaga and M. van der Schaar, "Wavelet coding for video streaming using new unconstrained motion compensated temporal filtering," in *Proceedings of the International Workshop on Digital Communications: Advanced Methods for Multimedia Signal Processing*, Sep 2002, pp. 41–48, Capri, Italy.

[TvdSA+05] D. Turaga, M. van der Schaar, Y. Andreopoulos, A. Munteanu, and P. Schelkens, "Unconstrained motion compensated temporal filtering (UMCTF) for efficient and flexible interframe wavelet video coding," *Signal Processing: Image Communication*, vol. 20, no. 1, pp. 1–19, Jan 2005.

[TZ94] D. Taubman and A. Zakhor, "Multirate 3-D Subband Coding of Video," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 572–588, Sep 1994.

[Tzi92] G. Tziritas, "A new Pel-Recursive Kalman-bsed motion estimation method," in *Proceedings of EUSIPCO'92*, 1992, pp. 1341–1344.

[UPND07] F. Urban, R. Poullaouec, J. F. Nezan, and O. Déforges, "H.264 Fractional Motion Estimation Refinement : a Real-Time and Low Complexity Hardware Solution for HD Sequences," in *Proceedings of 15th European Signal Processing Conference (EUSIPCO'07)*, 2007, pp. 836–840, Poznan, Poland.

[VBLV07] V. Velisavljević, B. Beferull-Lozano, and M. Vetterli, "Space-Frequency Quantization for Iamge Comrpession with Directionlets:," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1761–1773, Jul 2007.

[VBLVD05] V. Velisavljević, B. Beferull-Lozano, M. Vetterli, and P. L. Dragotti, "Approximation Power of Directionlets:," in *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, vol. 1, Sep 2005, pp. 741–744, Genova, Italy.

[VBLVD06] ——, "Directionlets: Anisotropic Multi-Directional Representation with Separable Filtering," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1916–1933, Jul 2006.

[VBLVD07] ——, "Image Representation and Compression using Directionlets:," in *Proceedings of SPIE Photonics & Optics*, Aug 2007, San Diego, CA, USA.

[VGP02] J. Viéron, C. Guillemot, and S. Pateux, "Motion compensated 2D+t wavelet analysis for low rate FGS video compression," in *Proceedings of International Thyrrhenian Workshop on Digital Communication*, Sep 2002, Capri, Italy.

[VM05] M. Vargas and E. Malis, "Visual servoing based on an analytical homography decomposition," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference*, Dec 2005, pp. 5379–5384, Seville, Spain.

[WA94] J. Y. A. Wang and E. H. Adelson, "Representing moving images with layers," *IEEE Trans. Image Processing*, vol. 3, pp. 625–638, 1994.

[Wat98] A. B. Watson, "Towards a perceptual video quality metric," in *Proceedings of the IS& T/SPIE Conference on Human Vision and Electronic Imaging III*, vol. 3299, Jan 1998.

[WB02] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 9, no. 3, pp. 81–84, Mar 2002.

[WBSS04] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr 2004.

[Wen03] S. Wenger, "H.264/AVC over IP," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 645–656, Jul 2003.

[WHS01] C. Wei, P. Hao, and Q. Shi, "Integer DCT-based Image Coding," in *Proceedings of Picture Coding Symposium (PCS'01)*, Apr 2001, pp. 175–178, Seoul, Korea.

[Wib48] H. Wibraham, "On a certain periodic function," *Cambridge and Dublin Mathematics Journal*, vol. 3, pp. 198–201, 1848.

[Win99] S. Winkler, "Visual quality assessment using a contrast gain control model," in *Proceedings of IEEE Signal Processing Society Workshop on Multimedia Signal Processing*, Sep 1999, pp. 527–532.

[WJ01] H. Watanabe and K. Jinzenji, "Sprite coding in object-based video coding standard: MPEG-4," in *Proceedings of Multiconference on Systemics, Cybernetics and Informatics*, vol. 13, Jul 2001, pp. 420–425.

[WJC05] X. Wu, , D. J. Jackson, and H.-C. Chen, "Novel fractal image-encoding algorithm based on a full-binary-tree searchless iterated function system," *Optical Engineering*, vol. 44, no. 10, p. 13, Oct 2005.

[WJH07] M.-S. Wu, J.-H. Jeng, and J.-G. Hsieh, "Schema Genetic Algorithm For Fractal Image Compression," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 4, pp. 531–538, Jun 2007.

[WJP+93] A. A. Webster, C. T. Jones, M. H. Pinson, S. D. Voran, and S. Wolf, "An objective video quality assessment system based on human perception," *Proceedings of SPIE*, vol. 1913, pp. 15–26, 1993.

[WKcLC00] Y.-T. Wu, T. Kanade, C. chung Li, and J. Cohn, "Image registration using Wavelet-based motion model," *International Journal of Computer Vision*, vol. 38, no. 2, pp. 129–152, 2000.

[WLB04] Z. Wang, L. Lu, and A. Bovik, "Video quality assessment based on structural distortion measurement," *Signal Processing: Image Communication, special issue on objective video quality metrics*, vol. 19, no. 2, pp. 121–132, Feb 2004.

[WR84] D. R. Walker and K. Rao, "Improved pel-recursive motion compensation," *IEEE Transactions on communications*, vol. 32, pp. 1128–1134, 1984.

[WRCB02] M. B. Wakin, J. K. Romberg, H. Choi, and R. G. Baranuik, "Image Compression Using an Efficient Edge Cartoon + Texture Model," in *Proceedings of IEEE Data Compression Conference (DCC'02)*, Apr 2002, pp. 43–52, Snowbird, UT, USA.

[WS03] T. Wiegand and G. J. Sullivan, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, Jul 2003.

[WSB03] Z. Wang, E. Simoncelli, and A. Bovik, "Multi-scale structural similarity for image quality assessment," in *Proceedings of IEEE Asilomar Conference on Signals, Systems and Computers*, 2003.

[WSD81] Z. Wu, H. Sun, and L. S. Davis, "Determining velocities by propagation," Maryland University., College Park, MD, USA, Tech. Rep., Dec 1981.

[WSWX06] C. Wang, X. Sun, F. Wu, and H. Xiong, "Image compression with structure-aware inpainting," in *Proceedings of International Symposium CAS on Circuit And Systems (ISCAS'06)*, 2006, pp. 1816–1819.

[Wu97] X. Wu, "Lossless Compression of Continuous-Tone Images via Context-Selection, Quantization and Modelling," *IEEE Transactions on Image Porcessing*, vol. 6, no. 5, pp. 656–664, May 1997.

[WXCM99] A. Wang, Z. Xiong, P. A. Chou, and S. Mehrotra, "Three-dimensional wavelet coding of video with global motion compensation," in *Proceedings of Data Compression Conference (DCM'99)*, 1999, pp. 404–413.

[WZVS06] D. Wang, L. Zhang, A. Vincent, and F. Speranza, "Curvelet Wavelet Tranform for Image Coding," *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2413–2421, Aug 2006.

[XLLZ01] G. Xing, J. Li, S. Li, and Y.-Q. Zhang, "Arbitrarily shaped video-object coding by wavelet," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 10, pp. 1135–1139, 2001.

[XWX+04] R. Xiong, F. Wu, J. Xu, S. Li, and Y.-Q. Zhang, "Barbell lifting wavelet transform for highly scalable video coding," in *Proceedings of Picture Coding Symposium*, Dec 2004, San Francisco, CA, USA.

[XXLZ01] J.-Z. Xu, Z. Xiong, S. Li, and Y.-Q. Zhang, "Three dimensional embedded subband coding with optimized truncation (3D ESCOT)," *Applied and Computational Harmonic Analysis*, vol. 10, pp. 290–315, 2001.

[YW94] O. L. Y. Wang, "Active mesh-A feature seeking and tracking image sequence representation scheme," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 610–624, Sep 1994.

[ZCBK01] Y. Zeng, L. Cheng, G. Bi, and A. C. Kot, "Integer DCTs and Fast Algorithms," *IEEE Transactions on Signal Processing*, vol. 49, no. 11, pp. 2774–2782, Nov 2001.

[ZCC01] G. Zhong, L. Cheng, and H. Chen, "Integer Lapped Biorthogonal Transform," in *Proceedings of IEEE International Conference on Image Processing (ICIP'01)*, vol. 2, Oct 2001, pp. 471–474, Thessaloniki, Greece.

[ZF06] B. Zeng and J. Fu, "Directional Discrete Cosine Transforms for Image Coding," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'06)*, Jul 2006, pp. 721–724, Toronto, Canada.

[ZK96] T. Zahariadis and D. Kalivas, "A spiral search algorithm for fast estimation of block motion vectors," in *Proceedings of the 8th European Signal Processing Conference (EUPSICO'96)*, vol. 29, Dec 1996, pp. 1799–1808.

[ZKU+04] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *Proceedings of ACM SIGGRAPH International Conference on Computer Graphics and Interactive Techniques (ICCGIT'04)*, 2004, pp. 600–608.

[ZM97] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block matching motion estimation," in *Proceedings of International Conference on Information, Communications and Signal Processing (ICICS'97)*, Sep 1997, pp. 292–296.

[ZM00] ——, "A new diamond search algorithm for fast block matching," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 287–290, Feb 2000.

[ZSWL07a] C. Zhu, X. Sun, F. Wu, and H. Li, "Video coding with spatio-temporal texture synthesis," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'07)*, 2007, pp. 112–115.

[ZSWL07b] ——, "Video coding with spatio-temporal texture synthesis and edge-based inpainting," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME'07)*, Apr 2007, pp. 813–816.

[ZT67] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method for Solid and Structural Mechanics*. McGraw Hill, New York, 1967.

[ZZYX05] G. Zhai, W. Zhang, X. Yang, and Y. Xu, "Image quality assessment metrics based on multi-scale edge presentation," in *Proceedings of IEEE Workshop on Signal Processing Systems Design and Implementation*, 2005, pp. 331–336.

List of Figures

List of Tables

Résumé

En quelques années, le trafic vidéo a augmenté de manière spectaculaire sur de nombreux médias. D'ici 2014, on estime que la quasi-intégralité du trafic IP sera composée de données vidéo. De même, l'usage de la vidéo sur les téléphones mobiles aura subi une augmentation sans précédent. Or, on estime que les infrastructures réseau, malgré les progrès constants en matière de transmission, ne pourront pas supporter une telle charge. A ce titre, il est plus que jamais capital d'améliorer nos capacités à compresser les vidéos.

Depuis 30 ans, la recherche travaille à l'élaboration de techniques de décorrélation, notamment afin de réduire les redondances spatiales et temporelles des séquences d'images et les compresser. A ce jour, l'approche classique est basée sur le concept de *macroblocs*: le contenu spatial est divisé en un ensemble de blocs. Le long de l'axe temporel, les images sont traitées une à une, sans faire apparaître de continuité évidente. Bien que cette approche soit déjà très efficace (cf. standards H.264/AVC et futur HEVC), l'emploi d'approches en rupture reste toujours envisageable. celles-ci offrent, entre autre, la possibilité de décrire l'évolution temporelle du contenu de manière continue. Cependant, elles mettent souvent en œuvre des outils dont l'utilisation, en pratique, est délicate.

Ce travail de thèse propose une nouvelle représentation, qui combine les avantages de l'approche classique et ceux de certaines approches en rupture, puis en étudie la viabilité. On cherche à exhiber la persistance temporelle des textures, à travers une description continue le long de l'axe temporel. A l'instar de l'approche classique, la représentation proposée est basée sur des blocs. Au lieu de réinitialiser la description à chaque image, notre représentation suit l'évolution de blocs initialement repérés à un instant de référence. Ces blocs, ainsi que leur trajectoire spatio-temporelle, sont appelés *tubes de mouvement*.

Trois problématiques sont soulevées. Tout d'abord, les *tubes* doivent être capable de représenter continuités et discontinuités du mouvement, ainsi que de suivre les déplacements et les déformations de patchs de texture. Des mécanismes de régularisation sont également mis en place, et s'assurent que l'évolution des tubes se fait de manière cohérente. Ensuite, la représentation doit gérer les recouvrements et les découvrements de *tubes*, et donc la manière dont la texture doit être synthétisée. Enfin, la problématique de vie et de mort des *tubes* est probablement la plus délicate: comment détecter la disparition ou l'impossibilité de suivre un patch de texture ? Le cas échéant, le *tube* correspondant devra être arrêté, ceci afin de garantir une représentation aussi compacte que possible. Les résultats montreront que notre représentation est viable, et ses performances seront comparées à celles du standard H.264/AVC.

Mot-clés : tubes de mouvement, vidéo, compression, représentation, persistance temporelle, suivi temporel

Abstract

Within a few years only, the amount of video information transmitted across a large range of communication channels has been critically increasing. It is expected, by 2014, that IP traffic will consist, most exclusively, of video data. In mobiles, video traffic is expected to undergo an increase without precedent as well. Despite the ever-increasing throughput of modern transmission channels, these will not be able to sustain such an increase in payload. More than ever, it is essential to improve our ability to compact the video information.

Research, for the past 30 years, provided numerous decorrelation tools that reduce the amount of redundancies across both spatial and temporal dimensions in image sequences. To this day, the classical video compression paradigm locally splits the images into blocks of pixels (*macroblocks*), and processes the temporal axis on a frame by frame basis, without any obvious continuity. Despite very high compression performances (*e.g.* H.264/AVC and forthcoming HEVC standards), one may still advocate the use of alternative approaches. Disruptive solutions have also been proposed, and notably offer the ability to continuously process the temporal axis. However, they often rely on complex tools (*e.g.* Wavelets, control grids) whose use is rather delicate in practice.

This thesis investigates the viability of an alternative representation that embeds features of both classical and disruptive approaches. Its goal is to exhibit the temporal persistence of the textural information, through a time-continuous description. However, it still relies on blocks, mostly responsible for the popularity of the classical approach. Instead of re-initializing the description at each frame, it is proposed to track the evolution of initial blocks taken from a reference image. A block, and its trajectory across time and space, is called a *motion tube*. An image sequence is then interpreted as a set of *motion tubes*.

Three major problems have been considered within this thesis. At first, *motion tubes* need to track both continuous and discontinuous displacements and deformations of individual patches of textures. Above all, it is critical for them to evolve as consistently as possible, which will require dedicated regularization mechanisms. Then, a second problem lies in the texture itself and the way it is used to synthesize images: how to handle non-registered and multi-registered areas. Finally, it is essential for a *motion tube* to be terminated whenever the corresponding patch of texture disappears or cannot be properly tracked any longer, which raises the problem of quality and efficiency assessment. This has a critical influence on the compactness of the representation. Results will eventually show that *tubes* can effectively be used to represent image sequences, and compare their performances with those of H.264/AVC standard.

Keywords: motion tubes, video, compression, representation, temporal persistence, tracking

INSA RENNES

N° ordre: D11-06

Institut National des Sciences Appliquées de Rennes

20, Avenue des Buttes de Coësmes . CS 14315 . F-35043 Rennes Cedex

Tel : 02 23 23 82 00 – Fax : 02 23 23 83 96