# Image structures: From augmented reality to image stylization

Jiazhou Chen

N° : ????

# THESIS

presented at

# BORDEAUX OF UNIVERSITY 1

DOCTORAL COLLEGE OF MATHEMATICS AND INFORMATION

by **Jiazhou Chen**

To obtain the degree of

## DOCTOR

SPECIALITY : COMPUTER SCIENCE

---

## Image structures:
### From augmented reality to image stylization

---

**Defend at :** July 12th 2012

**After review of :**

| | |
|---|---|
| Enhua Wu . . . . . | Professor |
| Joëlle Thollot . . . | Professor |

**The jury of defense includes :**

| | | |
|---|---|---|
| Enhua Wu . . . . . | Professor . | Reviewer |
| Joëlle Thollot . . . | Professor . | Reviewer |
| Adrien Bousseau | Researcher | Examiner |
| Pascal Guitton . . | Professor . | Examiner |
| Xavier Granier . | Researcher | Advisor |
| Qunsheng Peng . | Professor . | Advisor |
| Pascal Barla . . . . | Researcher | Co-Advisor |

– 2012 –

# Abstract

In this thesis we consider in general image structures and more specifically, image gradients and contours. They have been proven useful in recent years for various computer graphics applications, such as Augmented Reality (AR), image and video stylization. The goal of analyzing image structures is to describe a high level understanding of image contents and to provide a powerful support to improve the quality of applications, such as visual legibility, accuracy, spatial and temporal coherence.

We first demonstrate the important role of image structures in Focus+Context compositing. For Focus+Context rendering in AR, a technique dedicated to the visualization of hidden scenes in video streams, the use of screen segmentation and feature lines significantly emphasizes the depth cue of occluded scenes, and reveals the correct occluding order. We further extend Focus+Context rendering to importance-driven image synthesis, where image gradient and saliency map are used to composite multiple rendering styles in a coherent manner.

In the second part, we thus introduce a new approach to estimate a continuous gradient field without oversmoothing original details contained in an image. For this purpose, we develop a new and higher-order local approximation method for discrete non-oriented gradient fields based on a moving least square (MLS) formalism. We show that our isotropic linear approximation outperforms classical structure tensor: image details are better preserved and instabilities are significantly reduced. We demonstrate how our non-oriented MLS gradient field benefits to various image stylization approaches.

Finally, we demonstrate that the use of a feature profile analysis for image line extraction via fitting techniques permits to distinguish sharp and smooth features. Profile parameters are then mapped to stylistic parameters such as brush orientation, size or opacity to give rise to a wide range of line-based styles.

**Key-Words :** Image structure, Augmented reality, image and video stylization, gradient field, contours and lines.

# Résumé

Dans cette thèse, nous nous intéressons aux structures d'une image en général, et plus particulièrement aux gradients et aux contours. Ces dernières se sont révélées très importantes ces dernières années pour de nombreuses applications en infographie, telles que la réalité augmentée et la stylisation d'images et de vidéos. Le but de toute analyse des structures d'une image est de décrire à un haut-niveau la compréhension que l'on peut avoir de son contenu et de fournir les bases nécessaires à l'amélioration de la qualité des applications citées au-dessus, notamment la lisibilité, la précision, la cohérence spatiale et temporelle.

Dans une premier temps, nous démontrons le rôle important que ces structures jouent pour des applications de type composition "Focus+Context". Une telle approche est utilisée en réalité augmentée pour permettre la visualisation de parties d'une scènes qui sont normalement derrières ce que l'on peut observer dans un flux vidéo. L'utilisation d'une segmentation et de lignes caractéristiques permettent de mettre en avant et/ou de révéler les relations d'ordre entre les différents objets de la scène. Pour la synthèse d'images guidée par une fonction d'importance, de multiples styles de rendu sont combinés de manière cohérente grâce à l'utilisation d'une carte de gradients et une de saillance.

Dans un deuxième temps, nous introduisons une nouvelle techniques qui permet de reconstruire de manière continue un champ de gradient, et ceci sans trop lisser les détails originaux contenus dans l'image. Pour cela, nous développons une nouvelle méthode d'approximation locale et de plus haut-degré pour des champs de gradients discrets et non-orientés. Cette méthode est basée sur le formalisme "moving least square" (MLS). Nous démontrons que notre approximation isotrope et linéaire est de meilleure qualité que le classique tenseur de structure : les détails sont mieux préservés et les instabilités sont réduites de manière significative. Nous démontrons aussi que notre nouveau champ de gradients apporte des améliorations à de nombreuses techniques de stylisation.

Finalement, nous démontrons que l'utilisation d'une technique d'analyse de profil caractéristique par approximation polynomiale permet de distinguer les variations douces des zones dures. Les paramètres du profil sont utilisés comme des paramètres de stylisation tels que l'orientation des coups de pinceau, leur taille et leur opacité. Cela permet la création d'une large variété de styles de ligne.

**Mots-clefs :** Structures d'une image, réalité augmentée , stylisation d'images et de vidéos, champ de gradient, contours et lignes

# Acknowledgements

There is an old Chinese saying: “One day as your advisor, full life as your father!”. I wish to thank my advisors Prof. Qunsheng Peng, Xavier Granier and Pascal Barla. Prof. Peng created enough free space for me to grow, and nurtured me through my five years Master-PhD study like his own child. I can not imagine how I start and finish my PhD study in France without Xavier’s help, especially his invaluable discussion, meticulous writing correction, scholarship application and French culture introduction. And I can neither not imagine how vacant and wordless I would be if Pascal was not in the lab. His ideas, like his tangerines, will never be eaten up. I would like also to thank Prof. Pascal Guitton who made great efforts to build a great Sino-France collaboration, and found a funding for my joint-PhD program in Bordeaux.

There is another old Chinese saying from Confucius: “As three men are walking together, one of them is bound to be good enough to act as my advisor!”. Many other people contributed to this thesis. I would like to thank people from IPARLA team in Bordeaux, they are the nicest people I have met: Gaël Guennebaud, Christophe Schlick, Romain Vergne, David Vanderhaeghe for discussion and collaboration, and also Martin Hachet, Patrick Reuter, Romain Pacanowski, Nicolas Mellado, Yann Savoye, Simon Boye, Heqi Lu, Jérôme Baril for their help in the lab.

I also wish to thank former and current researchers and friends in the Peng group of the state key lab. of CAD&CG in Zhejiang University. Prof. Xueying Qin, Prof. Zhangye Wang have given me very helpful instructions to launch my passion on computer graphics research. Thanks to my collaborator Fan Zhong, Bin Pan, Yanli Liu, Naiyang Lin, Leiying Chen, Yujun Chen, I was able to finish several wonderful projects during this short PhD study. I want to express my appreciate to the helps from other colleagues: Yijiang Zhang, Yong Zhao, Xin Zhang, Guanyu Xing, Hanqi Fan, Jia Lu, Xia Yuan, Qingwei Wang.

Finally, I thank my parents, my wife and my family, for their selfless and endless love, comprehensiveness, support and care.

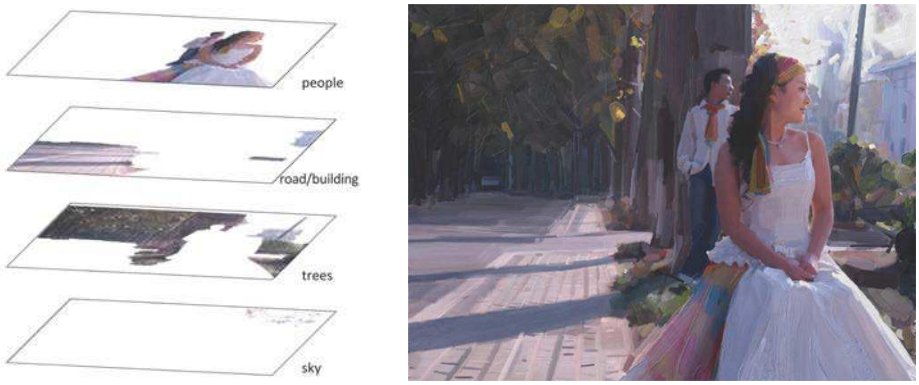
# Contents

# Chapter 1

# Introduction

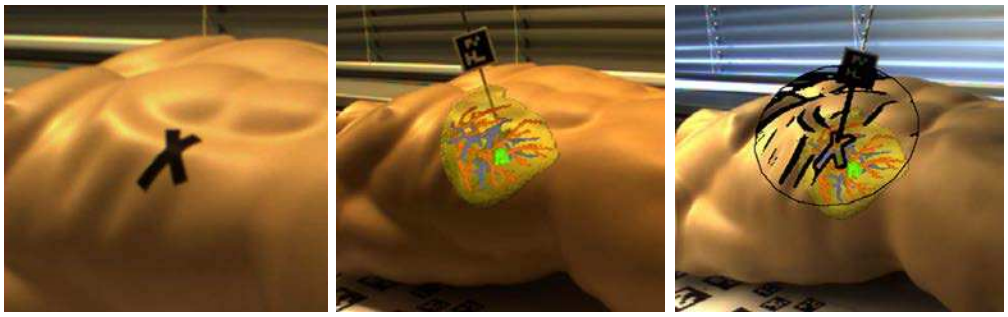
During the recent years, acquired images have been integrated into a large variety of computer graphics applications, thanks to the availability of affordable image-capture devices, such as digital cameras, cell-phones. For many of these applications, extended knowledge of the information contained in images is required, but absent in current 2D color array representation. For instance, smoothing is always employed as a pre-process to reduce noises of the image, but over-smoothing important features such as edges is difficult to prevent without the a-priori knowledge of their locations [SSD09]. Knowing the structures of an image, such as occlusion relationship, greatly improves the quality of image stylization (as shown in Figure 1.1). Unfortunately, for videos and mixed-reality applications, such a manual extraction of image structures will be hardly suitable for interactive applications. In this thesis, we thus focus on automatic techniques.

Image structure may be used in a larger range of applications than for image processing only. As an example, Focus+Context rendering is an Augmented Reality (AR) technique that aims to visualize hidden scenes in video streams: the vir-




Occlusion relation Painterly rendering

**Figure 1.1:** ***Painterly rendering based on image parsing [ZZXZ09].*** *The input image is segmented into four regions according to their occlusion relationships (left). Structural information improves the quality of painterly rendering. Boundaries are used to ensure the sharpness on the object contours. Occluding order is used to scale the brush size: small brushes for near scenes, large brushes for distant scenes.*



Input image | Direct integration | Integration w/ image structures

**Figure 1.2:** ***Visualization of hidden scenes [KMS07].*** *Direct integration, by adding the human organs on the top of the video stream (left), makes the medicinal operative landmark disappear (middle). By enhancing with lines, extracted from the video stream, the new AR system (right) provides visual cues to register the virtual object with the real environment. Therefore, doctors are able to insert the virtual iatrical instrument into the expected position accurately.*

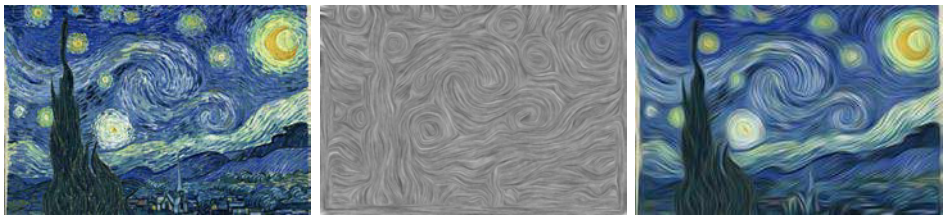
tual objects are the focus, while the real environment is the context. The structural information from the context could be used to improve the visualization quality. Figure 1.2 illustrates that extracted lines from video streams provide visual cues to register the virtual organ inside the human body.

The first part of this thesis is dedicated to the exploration of what are important image structures through one specific application: Context+Focus compositing in AR. We first present a new approach for the visualization of virtual underground pipelines in video streams based on cutaways [BF08] that improves Focus+Context rendering [KMS07] in AR. The contribution comes from the fact that direct compositing will cause an illusion that underground pipelines are floating on the ground. Thanks to our approach, we are able to provide depth cues to legibly reveal the correct occluding order.

Inspired from the Focus+Context rendering, we further extend it to importance-driven image synthesis using multiple rendering styles. Different layers are rendered separately with different styles, and the final result is obtained by compositing them driven by an importance map of the input image. Thanks to the gradient field that is, a field of 2D vectors that point in the direction of greatest rate of increasing of the input image luminance, we can thus stylize the input image in a spatial coherent manner.

The first part shows the importance of the gradient field. In the second part, we develop a non-oriented gradient field estimation approach, and demonstrate its advantages for image stylization. For instance, Figure 1.3 exhibits an improved image abstraction result using a gradient field estimated from Van Gogh's painting.

We thus introduce a new approach for defining a continuous non-oriented gradient field from an image in Chapter 5, a fundamental stage for a variety of computer graphics applications, such as curve reconstruction and image stylization.



Input image Tangent field Flow-based abstraction

**Figure 1.3:** ***Image abstraction using gradient field [KLC09].*** *The floating feeling in Van Gogh's painting is exaggerated by a flow-based bilateral filter. Note that here we employ a line integral convolution algorithm [CL93] to visualize the tangent field, that is orthogonal to the gradient field.*

Our approach is built on a moving least square formalism where the key is the computation of higher-order local approximations of non-oriented input gradients. In particular, we show that our novel isotropic linear approximation outperforms its lower-order alternative: image structures are much better preserved, and instabilities are significantly reduced.

We demonstrate the validity of our MLS gradient field on structure preservation by experimenting it on various stylization algorithms in Chapter 6, such as continuous glass pattern [PP09], flow-based image abstraction [KLC09], enhanced shock filter [KK11]. We also extend to image and video a feature profile analysis approach, that identifies feature lines via fitting techniques. Our approach extracts not only the location of features, but also their profile attributes, which permits to distinguish between sharp and smooth feature lines. Profile parameters are then mapped to stylistic parameters such as brush orientation, size or opacity.

In conclusion, we not only review all the contributions, but also detail interesting research directions that deserve further endeavor, including exploring advanced image structures, such as scales and multiple directions, structures for videos and 3D volume data, and an interactive image structure analysis tool.

CHAPTER 2

# Image structures

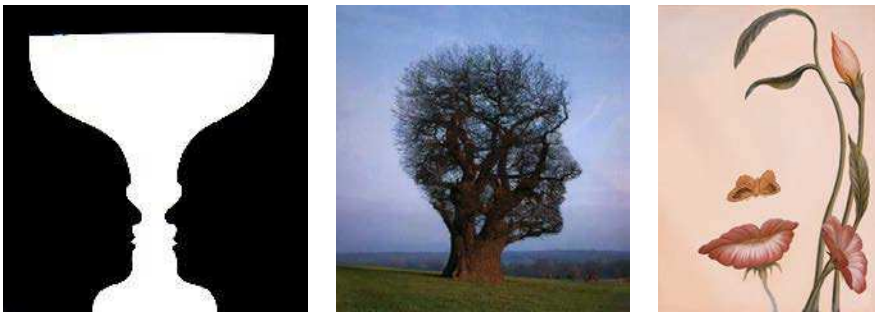
In this chapter, we present a general overview of the use of image structures and the previous work on their extraction. We particularly focus on gradient field and object contours, and emphasize their important roles in Augmented Reality and image stylization applications.

## 2.1 The use of image structures

Classically, an image is a 2D array that stores color data at each position. This array representation provides great conveniences for storage and directly matches screens and video projectors, but it does not contain any information about the image contents. When humans look at an image, they do not only see a pattern of color and texture, but the world behind it [HEK07]. Unfortunately, computer intelligence is far from fully recovering the world from a single image. But, researches in image structure analysis during recent decades show that essential image structures can be regarded as higher level understanding of image contents than a 2D color array, and provide richer structural information for further image processing in various applications.

**Contours** play an important role in human vision. Early clinical experiments [Zek93] show that damages in brain areas which are in charge of contour perception will make people completely lose the ability to recognize objects. Optical illusions in Figure 2.1 show that humans are naturally accomplished in discovering object contours, even if they are ambiguous in positive and negative spaces (left), and contrary to the real contents in the images (middle and right).

Due to the importance of object contours, contour-based image representation provides an ideal alternative of 2D array, which opens a door for creating and manipulating an image much more easily. Diffusion curves are such a powerful vector graphic tool that represents an image as a set of 2D Bézier curves with colors on both sides. Figure 2.2 illustrates an example of diffusion curves [OBW+08]. Though the reconstructed result (b) is smoother and more stylized, it looks very



© Edgar Rubin © CoolOpticalIllusions © Octavio Ocampo

**Figure 2.1:** ***Optical illusions*** *show that object contours play an important role in object recognition for human vision. Rubin's vase (left) shows a vase in positive space, while the negative space around the vase forms the contours of two faces in profile. Though the photo in the middle shows a tree, it is no doubt that we can immediately recognize the head contour. Similarly in the painting "Boca del Flor" (right), we see a woman face at the first glance, even faster than the flowers.*

(a) Input image (b) Drawing result (c) Diffusion curves

**Figure 2.2:** ***Diffusion curves [OBW+08].*** *Contour-based image representation (b) shows a result of a tracing using active contours and color sampling drawn by © Adrien Bousseau, and (c) is its corresponding diffusion curves.*


(a) [SKSK09] (b) [CCT+09] (c) [JTC09]

**Figure 2.3:** ***Line sketching for modeling.*** *Different line sketching, three input examples in the top row, provides sufficient description of a 3D scaffold (a), image content online (b), or symmetric architectures (c).*

close to the original image (a). What is more important is that the reconstructed result is created by drawing several curves and assigning their colors on both sides, as shown in Figure 2.2-(c). The reconstruction is done by solving a Poisson differential equation. This means the new image can be zoomed in an infinite resolution, and be easily edited by modifying sparse curves.

Object contours are also an ideal medium to describe an object, since they usually correspond to the maxima of differential geometric properties, such as silhouettes [DFRS03], ridges and valleys [LMLH07]. Such shape contours are also intuitive and have thus been used in sketching techniques [SKSK09, CCT+09, JTC09]. For instance, Google © SketchUp provides simple but efficient sketch-based user interaction tools to build a 3D scene. Figure 2.3 exhibits three recent applications that allow the user to model a 3D object or to search an expected image by sketch-based object description. Schmidt et al. [SKSK09] have introduced a new analytic drawing tool that infers 3D curves from perspective line drawings, and further 3D scaffolds reconstructed from these 3D feature-curves (as shown in Figure 2.3-(a)). Figure 2.3-(b) shows that simple freehand line sketches can be used to automatically convert a set of simple strokes into a photo-realistic picture by searching online images that closely match these contours and by seamlessly compositing them [CCT+09]. A 3D model can also be constructed using several sketched lines, for instance a symmetric architecture from a single image as shown in Figure 2.3-(c) [JTC09].

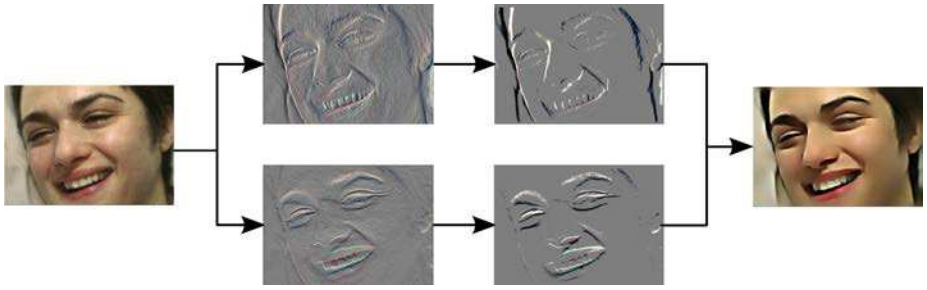
However, contours alone are not sufficient to describe small-scale image structures. For instance, diffusion curves preserve the contours of sharp objects and smooth interior regions, but fails to mimic texture details. Though an extended version [JCW11] defines texture details using Gabor noise in a resolution independent manner, but it can only model certain kinds of textures. To preserve image structures and texture details in various applications, other image structures such as gradients are needed to describe both the interior regions and contours of an object [LZPW04, JT08, MP08, BZCC10].

**Gradient** Gradient field is another important image structure. For instance, it has been used for image stitching. The use of gradient field allows to remove the visible seams in stitched images [LZPW04, ADA+04, AAC+06, JT08]. Even the images to stitch are captured at the same time, differences still remain due to either photometric inconsistencies caused by exposure times, or geometric misalignments caused by camera movements and moving objects in the images. Direct stitching will thus introduce seam artifacts, and simple color blending will blur out the compositing regions. It will be even more difficult if the images are not well registered [JT08]. Levin et al. [LZPW04] demonstrate that solving a cost function over image gradients overcome the above issue. Jia and Tang [JT08] further show that geometric misalignments can be removed by image structure deformation.

The use of image gradient also extends traditional image editing methods. A painting tool designed in the gradient domain [MP08] requires much less effort for

Source images Target image Pasted result

**Figure 2.4:** ***Poisson image editing [PGB03].*** *In order to integrate the source image into target image seamlessly, only spatial gradients of the source images are preserved, and the color of target image on the region boundaries is the Dirichlet boundary condition. Note how the tone of the source images adapts to the target image.*



Input image Original gradients Smoothed gradients Abstracted result

**Figure 2.5:** ***Basic pipeline of gradient domain manipulation [BZCC10].*** *Instead of abstracting the original image, gradient domain methods smooth the image gradients, the abstracted result is computed by a least square solver using the smoothed image gradients.*

users to accomplish global lighting and contrast adjustments, requiring only local image manipulations. Instead of assigning colors on both sides of contours as diffusion curves [OBW+08], gradient-domain painting directly allows users to draw, remove or copy contrasts on the boundaries. Image drag-and-drop in gradient domain [PGB03] recomputes the color of each pixel in the pasted region by solving a Poisson equation. Figure 2.4 illustrates this application, where image gradients from source images (a) and colors on the pasted boundaries in the target image (b) constitute Poisson constraints to compute the color in the pasted region (c). Image interpolation in gradient domain [MHM+09] moves gradients in the original images to produce an animation sequence that preserves the frequency content of the original images. Compared to traditional image morphing of pixel colors [BN92], interpolation in gradient domain provides a fully automatic solution.

Gradient information has been also used in image filters. Gradient-based image

Occluding scene Remote scene Focus+Context compositing

**Figure 2.6:** ***Visualization of hidden scenes [AST09].*** *In order to visualize remote scenes occluded by a wall, Focus+Context rendering techniques in AR overlay extracted lines to the hidden scenes, to reveal the correct occluding order.*

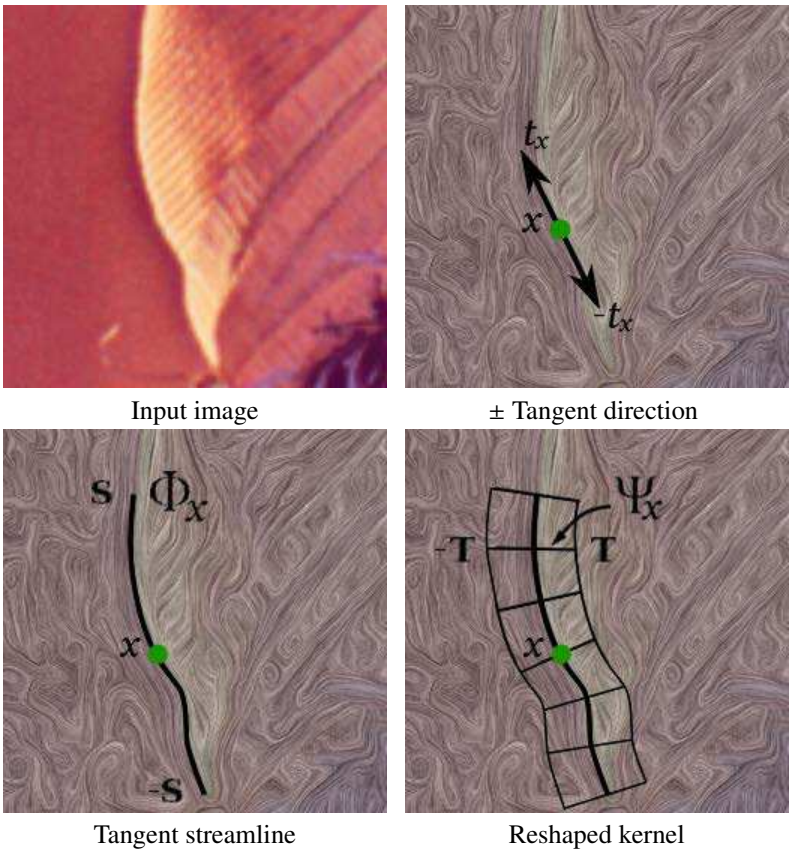
filters manipulate pixel difference (e.g. first order Gaussian derivatives) in addition to pixel values of an image [BZCC10]. Figure 2.5 illustrates the basic pipeline of a gradient domain filter. Instead of directly editing the original image, gradient domain methods manipulate the image directional gradient (along x- and y- frame), the final result is reconstructed by new image gradients using a least square solver. But it is limited to first order features, thus not straightforward to work with other image features, such as ridges and valleys.

As a conclusion of this section, all aforementioned applications show that contours and gradients, can provide a higher level of image understanding, thus improve the quality of these applications. We will give more details in the following section for their use in Augmented Reality and image stylization.

## 2.2 Focus+Context compositing

Augmented Reality (AR) is classically known as a seamless integration of virtual objects into real images or videos. Reconsidering AR from another viewpoint, the main goal of combination of virtual and real information is visual communication. For this purpose, stylized rendering has been proven to be efficient for revealing shapes and avoiding visual clutter, when compared to photo-realistic rendering [FBS05, WCL+10, CTM08, HLB05]. In practice, object contours are popularly used in stylized rendering for AR, since they enable quick recognition of objects with little distraction from relatively unimportant contexts. Contours of virtual objects are usually extracted in a view-dependent manner from a texture buffer that is either depth buffer [HLB05] or rendered luminance of the object [FBS05, WCL+10, CTM08]. And the uniformity of integration is emphasized by combining contours of virtual object and lines extracted from images or videos.

In this thesis, we are particularly interested in Focus+Context rendering of AR, which is known as visualization of hidden scenes in images or videos. In order to provide users a visual similarity of X-Ray vision to see-through the occluders, image structures, such as feature lines and curvatures, are usually preserved in the occluding regions [KMS07, AST09, KSW06]. Figure 1.2 in the previous chapter




Input image ± Tangent direction

Tangent streamline Reshaped kernel

**Figure 2.7:** ***An illustration of gradient-guided image kernel reshaping*** *on part of Lena image introduced by Kang et al. [KLC09]. We apply a blending of original image and LIC approach [CL93] to visualize the gradient field using a length 31 pixels. Given a gradient field, a streamline $\Phi_x$ is traced along the tangent direction on both sides $\pm t_x$, and the image kernel is reshaped by this streamline instead of isotropic kernel. Various image filters can benefit from this new kernel to preserve better image structures.*

shows one advantage of image structures demonstrated in work of Kalkofen et al. [KMS07], that is helping viewers to register between the virtual and real space by preserving image lines in the occluding region. Avery et al. [AST09] have presented a X-Ray vision system that allows users to see through a wall the remote scenes behind it, meanwhile lines extracted from the walls provides depth cues to make hidden objects appear to be behind the wall, as illustrated in Figure 2.6. Kalkofen et al. [KMS07] have also pointed out that image structures like texture details and edges can provide improved depth cues.

Image gradient can not directly benefit classical AR applications, but Wang et al. [WCL+10] have shown that it is particularly useful for stylized AR system. High-quality line drawing is obtained by using an adapted Flow-based anisotropic

Difference-of-Gaussian (FDoG) filter [KLC09, KD08]. Thus it improves both the spatial coherence of a single frame and the temporal coherence of augmented video streams.

## 2.3 Image stylization using image structures

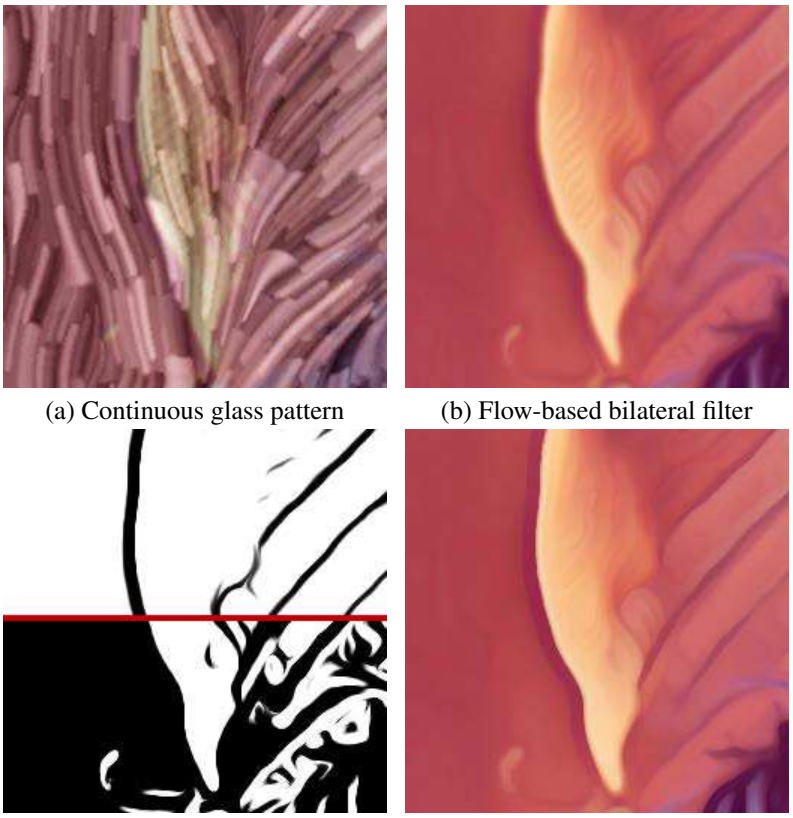
The use of gradient information for image stylization has been proven to increase accuracy. In particular, we are interested in gradient fields, defined as a directional field of 2D vectors that point in their gradient directions. Figure 2.7 visualizes such a gradient field. For stroke-based painterly rendering, gradient field guides the orientation of strokes [HE04, ZZXZ09], that mimics the classical way artists paint. For filtering-based image abstraction, the use of gradient field permits to orient filters along image features [KLC07, KLC09, KKD09, KK11] which guarantees spatial coherence. Figure 2.7 exhibits how a filter kernel is reshaped according to the local gradient field. For each position $\boldsymbol{x}$, a streamline is traced along the tangent direction with an assigned length S starting from both the positive and negative tangential direction $\pm \mathrm{t}_{\boldsymbol{x}}$ of $\boldsymbol{x}$ forms a tangent curve $\Phi_{\boldsymbol{x}}$. Reshaped kernel is then built by equidistant samples on the gradient curves $\Psi_{\boldsymbol{x}}$, which are streamlines trace along gradient field and start from the equidistant positions on $\Phi_{\boldsymbol{x}}$, or straight lines. Note that the tangent is the direction orthogonal to the gradient, thus the corresponding tangent field is automatically known if a gradient field is given.

There are many strategies to trace the streamlines. A straightforward approach is to use Euler integration method. Kyprianidis and Kang [KK11] have shown that second-order Runge-Kutta integration traces the streamlines at higher accuracy and does not blur edges. Since samples might be pixel centered, a bilinear interpolation is required to avoid aliasing. Fortunately, it is much less expensive with the help of parallel processing on GPU. Note that since the tracing occurs on both sides, we could only rely on the non-oriented direction, not the particular orientation, which means that we have to decide whether to go forward or backward depending upon the orientation of the previous step [KK11]. In practice, the orientation of current position $\boldsymbol{s}_i$ has to be temporarily flipped if it is not consistent with the previous orientation $\boldsymbol{s}_{i-1}$, when the dot product is negative that is $(\boldsymbol{s}_i, \boldsymbol{s}_{i-1}) < 0$.

**Gradient field visualization** Line Integral Convolution (LIC) [CL93] is a very classical gradient field visualization approach. It convolves an input noise texture $N$ using a low-pass filter along $\Phi_{\boldsymbol{x}}$ to exploit spatial correlation in a gradient field g:

$$\mathcal{L}_{\mathrm{g}}(\boldsymbol{x}) = \frac{1}{K} \int_{-S}^{S} N[\Phi_{\boldsymbol{x}}(s)]\, ds. \tag{2.1}$$

where $S$ is the half length of the 1D convolution kernel along the streamline, and $K = 2S + 1$ is a normalizing term. The input noise texture is generally a 2D white Gaussian noise texture. A discrete LIC on images is implemented by using bilinear interpolation. A LIC visualization blended with the original image is shown in



(a) Continuous glass pattern (b) Flow-based bilateral filter

(c) FDoG/XDoG (d) Enhanced shock filter

**Figure 2.8:** ***Various image stylization using gradient field*** *on part of Lena image. Isotropic filtering kernels are reshaped according to local gradient field, which allows to preserve better the original structures.*

Figure 2.7.

Recently, another technique has emerged. It is based on Glass Pattern [G+69], known as the superposition of two random point sets, one of which is obtained from the other by a small geometric transformation. Continuous Glass Pattern (CGP) [PP09], is defined as the maximum of point sets that are on the trace of continuous geometric transformation. Given a gradient field g and an input image $\mathcal{S}$, CGP for position $\boldsymbol{x}$ can be computed in a straightforward way by integrating along the gradient field and by taking the maximum of $\mathcal{S}$ over a streamline $\Phi_{\boldsymbol{x}}(t)$ (the same definition as LIC).

$$\mathcal{G}_{\mathrm{g}}(\boldsymbol{x}) = \max_{s\in[-S,S]} \{\mathcal{S}[\Phi_{\boldsymbol{x}}(s)]\}. \tag{2.2}$$

CGP convolutes a synthetic texture rather than a raw random noise texture, which permits to simulate brush strokes with different thickness. This integration is done numerically by using an Euler algorithm and the input image $\mathcal{S}$ is generated by

convolving a white Gaussian noise $N$ with a 2D Gaussian smoothing with a standard deviation $\sigma$. The stroke thickness is globally controlled by $\sigma$. Figure 2.8-(a) is a continuous glass pattern result with $\sigma = 2.0$ and $S = 25$.

**Gradient-based color abstraction** Convolution over a noise texture based on a gradient field extracted from the input image visualizes the local gradient of this input image. LIC convolutes a Gaussian noise texture, while CGP convolutes a synthetic texture. Kang et al. [KLC09] have shown that a separated bilateral filter [TM98] can be employed in the same way to improve the spatial coherence of color abstraction if taking the input image as the texture to convolute. The authors have proposed a flow-based bilateral (FBL) filter that separates 2D bilateral filter into two 1D bilateral filters, first along the tangent direction (Equation 2.3), and then along the gradient direction (Equation 2.4).

$$\mathcal{B}_{\mathrm{t}}(\boldsymbol{x}) = \int_{-S}^{S} I[\Phi_{\boldsymbol{x}}(s)]\, G_{\sigma_s}(s)\, G_{\sigma}(|I(\boldsymbol{x}) - I[\Phi_{\boldsymbol{x}}(s)]|)\, ds. \tag{2.3}$$

$$\mathcal{B}_{\mathrm{g}}(\boldsymbol{x}) = \int_{-T}^{T} I[\Psi_{\boldsymbol{x}}(t)]\, G_{\sigma_t}(t)\, G_{\sigma}(|I(\boldsymbol{x}) - I[\Psi_{\boldsymbol{x}}(t)]|)\, dt. \tag{2.4}$$

In practice, they are separated into two consecutive passes, thus this approximation considerably reduces the computation cost. Bilateral filter is known as a structure-preserving smoothing technique, but it still has color blending issues, especially with a large neighborhood size [CPD07, SSD09]. FBL is able to overcome this issue: a strong filter along tangent field can be applied (large $\sigma_t$) to smooth colors without oversmoothing structures, since tangential streamline will not go through features. And a weak filter along the gradient field is needed (small $\sigma_s$), because we know that gradients always go through the features if exist. Figure 2.8-(b) is a bilateral filter guided by a gradient field, with $S = 25$, $\sigma_t = 8.0$, $\sigma_s = 0.5$ and $\sigma = 0.2$.

**Coherent line drawing** In the same spirit of flow-based bilateral filter, flow-based Difference-of-Gaussian (FDoG) applies 1D DoG along gradient field (Equation 2.5) filter followed by a 1D tangential smoothing along tangent field (Equation 2.6), instead of 2D DoG [WOG06].

$$\mathcal{D}_{\mathrm{g}}(\boldsymbol{x}) = \int_{-T}^{T} I[\Psi_{\boldsymbol{x}}(t)]\, DoG_{\sigma_c}(t)\, dt. \tag{2.5}$$

$$\mathcal{D}_{\mathrm{t}}(\boldsymbol{x}) = \int_{-S}^{S} I[\Phi_{\boldsymbol{x}}(s)]\, G_{\sigma_m}(s)\, ds. \tag{2.6}$$

where $DoG_{\sigma_c}(t) = G_{\sigma_c}(t) - \rho\, G_{\sigma_s}(t)$, and $\sigma_s$ is set as $1.6\,\sigma_c$ to approximate closely to Laplacian-of-Gaussian. By setting $\rho$ less than 1, a black lines will be obtained if a step edge is nearby. $\sigma_c$ controls the spatial scale of the lines, and $\rho$ determines the sensitivity of the edge detector. For small values of $\rho$, less noise is detected, but

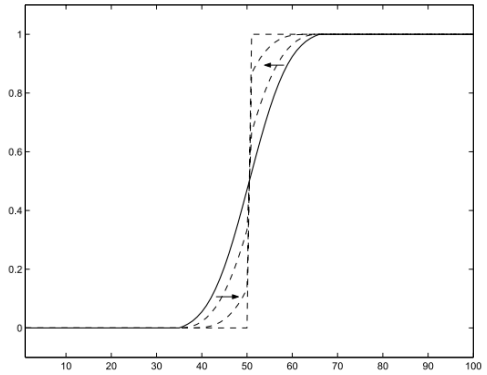



**Figure 2.9:** ***Iterative shock filter illustration on 1D example.*** *The step edges are sharpened by adding/reducing the gradient magnitude in convex/concave regions.*

real edges become less prominent. This flow-based separate filter not only speeds up the performance, but also improves the spatial coherence. In order to obtain different stylization effects, an adjustable soft ramp function is designed:

$$I'(x) = \begin{cases} 1, & \text{if } DoG(x) < \epsilon \\ 1 + \tanh(\varphi \cdot DoG(x)), & \text{otherwise} \end{cases} \tag{2.7}$$

where $\varphi$ controls the sharpness of the lines, and $\epsilon$ is a parameter to clamp the $DoG$ operator.

Winnemöller [Win11] has demonstrated that an extended version of flow-based DoG (XDoG) can get various kinds of stylization by modifying parameters $\rho$, $\varphi$ and $\epsilon$. For instance, it is able to convert the input image into a black-and-white image when $\rho < 1$, $\varphi \gg 1$, and $\epsilon < 0$. Figure 2.8-(c) shows a FDoG edge detection result (top part) and a XDoG black-and-white converting result (bottom part).

**Sharpening** Shock filters [OR90] create strong discontinuities at image edges, by a local dilation or erosion, depending on whether the pixel belongs to the influence zone of a maxima or a minima [Wei03]. Figure 2.9 illustrates how shock filter works iteratively on 1D. For each pixel $(x)$ of an image $I$, the classical shock filter evolution equation is given by:

$$h(\boldsymbol{x}) = -\operatorname{sign}(\triangle I)\ |\nabla I| \tag{2.8}$$

Gradient magnitude $|\nabla I|$ is added in convex regions ($\operatorname{sign}(\triangle I) < 0$) where a local maximum usually exists nearby, while gradient magnitude is subtracted in concave regions ($\operatorname{sign}(\triangle I) > 0$) where is a local minimum usually exists nearby. For an image, the gradients and curvatures are computed by 2D filters. The gradient magnitude is recomputed in each iteration, thus its addition or subtraction will sharpen the edges, but will not exceed the original luminance range.

Similar to FDoG, flow-based shock filter applies shock filter only along the gradient field, and Gaussian smoothing along the tangent field:

$$\mathcal{H}_{\mathrm{g}}(\boldsymbol{x}) = \int_{-T}^{T} I[\Psi_{\boldsymbol{x}}(t)]\, h(t)\, dt. \tag{2.9}$$

$$\mathcal{H}_{\mathrm{t}}(\boldsymbol{x}) = \int_{-S}^{S} I[\Phi_{\boldsymbol{x}}(s)]\, G_{\sigma_m}(s)\, ds. \tag{2.10}$$

In practice, the Laplacian and gradient operator in Equation 2.8 can be computed directionally. Thus, 1D shock filter is used to replace 2D version:

$$h(\boldsymbol{x}) = -\operatorname{sign}(\nabla_{\mathrm{g}}\, \nabla_{\mathrm{g}}\, I)\ \left|\nabla_{\mathrm{g}} I\right| \tag{2.11}$$

where $\nabla_{\mathrm{g}}\boldsymbol{.}$ is a 1D Gaussian derivative operator along gradient direction, its standard variation $\sigma$ controls the sharpening size. Kyprianidis and Kang [KK11] employ shock filter to obtain sharp transitions at edges. Figure 2.8-(d) shows a result of our implementation of their approach with one iteration and $\sigma = 2$. Kang and Lee [KL08] use shock filter to simplify shapes by combining mean-curvature flow.

## 2.4 Gradient field estimation

As illustrated in the previous section, gradient information is extraordinarily important in image stylization. But does the quality of the gradient affect the quality of image stylization? The answer is affirmative.

Many methods for computing image gradient have been proposed in recent decades, such as Sobel filter, Gaussian derivatives, steerable filter [FAoTMLVG91]. All of them can be used to compute directional gradient of an image $I$ in x- and y-frame, denoted as $\mathrm{g}_x$ and $\mathrm{g}_y$. Then a gradient field is defined as a vector field, that contains a normalized 2D vector $(\mathrm{g}_x, \mathrm{g}_y)$ for each position. This gradient field definition can be only computed for single channel images. A simply method to make use of this computation is to convert the color image into a gray image as a preprocess. But this conversion will immediately lose the color variation of the original image. To overcome this issue, color gradient is becoming popular recently [KD08, KKD09, KK11]. Frame derivatives are first computed separately for RGB channels:

$$I_x = (R_x, G_x, B_x), \quad I_y = (R_y, G_y, B_y) \tag{2.12}$$

Then a structure tensor is defined as a Jacobian matrix:

$$\begin{pmatrix} I_x \cdot I_x & I_x \cdot I_y \\ I_y \cdot I_x & I_y \cdot I_y \end{pmatrix} =: \begin{pmatrix} E & F \\ F & G \end{pmatrix} \tag{2.13}$$

This structure tensor is a symmetric matrix, the eigenvalues are real numbers and the eigenvectors are orthogonal. The major eigenvalue $\lambda_1$ is zero if and only if

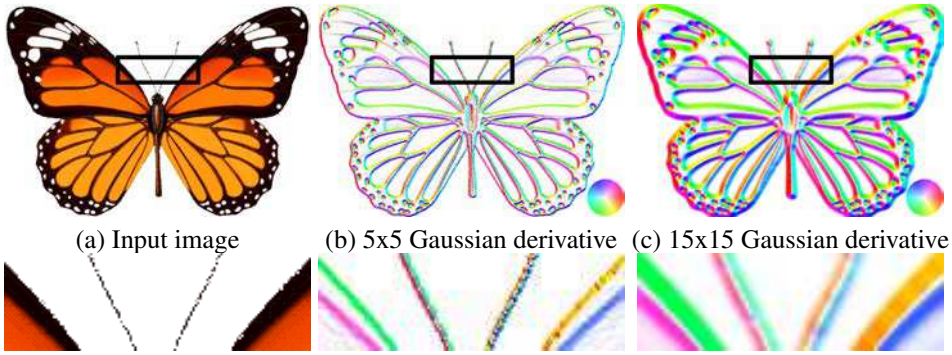



**Figure 2.10:** ***Gradient field Visualization using oriented mapping.*** *We map the gradient field to HSV color space, where hue represents angle of its direction while saturation represents gradient magnitude re-mapped to [0,1]. The mapping circle is shown in the right bottom corner. Visualization with oriented mapping in (b) and (c) shows that Gaussian derivative using either small or large neighborhood size has close-to-zero gradient magnitude in the center of thin features, and opposite directions on both sides. For instance, if we zoom in on the butterfly, we will see a white line in the middle of the antennas, and redgreen colors on the leftright of the antennas.*

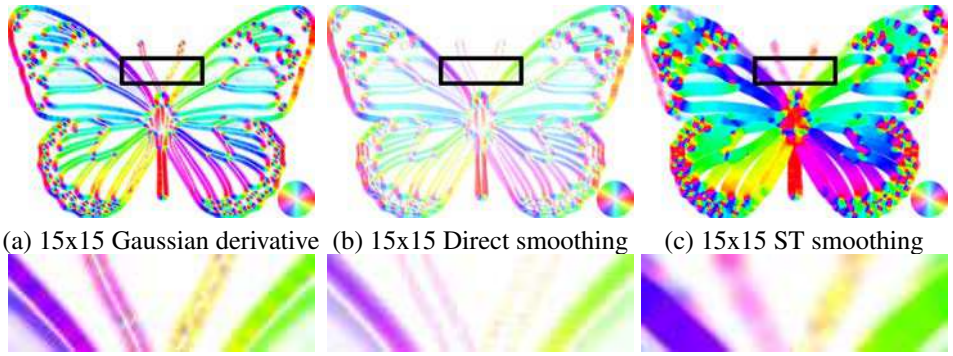



**Figure 2.11:** ***Gradient field Visualization using non-oriented mapping.*** *We map the gradient field to HSV color space, where hue represents* ***2 times*** *the angle of its direction while saturation represents magnitude re-mapped to [0,1]. The mapping circle is thus axial symmetrical. With our non-oriented mapping, gradients on both sides of the features appear the same color (a). Directly averaging opposite directions introduce pair cancellation issues (b). Structure tensor naturally encodes the non-oriented vector information (c), smoothing structure tensor result in filling in the direction gaps in the center of the features, but still quickly blurs out the original structures.*

both gradients are zero, and the minor eigenvalue $\lambda_2$ is zero when the input image is grayscale [KD08]. The eigenvectors are separately computed:

$$v_1 = \begin{pmatrix} F \\ \lambda_1 - E \end{pmatrix} \quad v_2 = \begin{pmatrix} \lambda_2 - G \\ F\lambda_1 - E \end{pmatrix} \tag{2.14}$$

and

$$\lambda_{1,2} = \frac{E + G \pm \sqrt{(E-G)^2 + 4F^2}}{2} \tag{2.15}$$

Color gradient encodes all the gradients from RGB channels, thus lose less color information than using only grayscale images. But their orientation is still discontinuous, even using large neighborhood size or large scale. For instance, a thin line feature is expected to yield to a unique tangent direction while its sides define opposite gradients and its center gives close-zero-magnitude gradient [BVL+06], as illustrated in Figure 2.10. Figure 2.10-(b) and (c) illustrate two gradient fields computed by Gaussian derivatives with small and large neighborhood size. Visualization using HSV color space mapping shows that derivatives computed with larger neighborhood size don't fix the above issues, and tend to have an oversmoothed field.

A sophisticated way to obtain smooth but reliable gradient field is to first compute the gradient with a small scale, and to smooth it considering its magnitude and directional discontinuities [KLC09]. But the opposite directions on both sides of the thin features introduces pair cancellation issues: directly averaging these opposite directions might produce close-to-zero directions. This can be observed around the antennas of butterfly in Figure 2.11-(b). Attempting to locally re-orient them as Kang et al. [KLC09] is bound to fail if the reference direction is not reliable when considering large neighborhoods. In order to overcome these difficulties, the most common approach consists in encoding the *non-oriented* vector information into a *structure tensor* [BG87]. Equation 2.13 is an example of structure tensor definition. This representation has been employed to interpolate very sparse non-oriented vector data for synthesis [ZHT07] or abstraction [KK11] purposes. Filtering structure tensors, smoothing the E, F and G in Equation 2.13, regularizes noise and merges gradients coming from multiple image channels [KD08]. Smoothing the angle of the gradients is equivalent structure tensor mathematically, and leads to similar result in practice [OH12]. In order to visualize a non-oriented gradient field and be similar to [OH12], we multiply the angle of the gradient by 2 before HSV color mapping. In the new visualization of Figure 2.11, opposite gradients ($\theta$ and $\theta \pm \pi$) have the same color, because $2\theta \equiv 2(\theta \pm \pi) \,(\mathrm{mod}\, 2\pi)$.

An explicit method to estimate a coherent gradient field without oversmoothing is to trust gradients in place where we are confident and use them to interpolate the whole image [HE04, OH12]. Considering the sparsity of image salient structures, Radial Basis Function (RBF) becomes a natural choice. Hays and Essa [HE04] and O'Donovan and Hertzmann [OH12] interpolate the whole gradient field using RBF based on sparse structures drawn by users. In order to provide more control

of the gradient field, O'Donovan and Hertzmann [OH12] further propose a mixture framework that allows to combine different fields, such as image gradient, region boundary field, constant field and the one constructed by RBF. Though O'Donovan and Hertzmann demonstrate that drawn strokes can be propagated through neighbor frames in video, specifying and adjusting structures in video frames is not an easy task for humans, which consumes a lot of time. An automatic method to find image structures can surely save many efforts. Kyprianidis and Kang [KK11] remove the gradients in low-contrast regions, and fill in by a process of relaxation. But, a unmanageable balance between removing gradients and preserving structures will always exist. Therefore, we believe that implicitly smoothing out unreliable gradients is better to preserve original structures than predicting structures explicitly.

## 2.5 Line extraction and rendering

We have introduced in Section 2.1 that feature lines extracted from images are used in various applications, including Augmented Reality and image stylization. The problem of identifying image feature lines has received a lot of attention in previous work. Lines in an image can be object silhouettes, occluding contours, folds of object surface, edges in object textures, or even edges caused by shadows. The boundaries of segmented regions produce closed lines [DS02, CM02], they usually refer to object silhouettes or occluding contours. But, the smoothness of the boundaries is difficult to guarantee. For instance, even if a hierarchical segmentation is already built, a curve smoothing process is still needed to have smooth contours [DS02]. Therefore, we mainly focus on direct feature-line extraction algorithms, rather than by segmentation.

Among all of the direct line extracting algorithms, Canny edge detection method is probably the most well-known one. It is defined as the zero-crossing of the second order directional derivatives of a smoothed image. Mathematically, it is equivalent to the maxima and minima of the first order derivative [LD08]. In practice, only the maxima crossings are of interest since these pixels represent the areas of the sharpest intensity changes in the image [MH80].

Ridges and valleys are second-order features, thus need to compute a curvature tensor $\mathbf{H}$ from which principal curvatures $k_{\max}$ and $k_{\min}$ and directions $\mathbf{t}_{\max}$ and $\mathbf{t}_{\min}$ are extracted. They have been well studied for line drawing of 3D objects [JDA07, OBS04]. For images, the same algorithm can be applied on the luminance after a suitable smoothing.

Inflections are third-order features, and thus require to compute a curvature-variation tensor $\mathbf{C}$ from $\mathbf{H}$, and extract a curvature gradient $\mathrm{v}_{\max}$ from $\mathbf{C}$. Luminance inflections are similar to Demarcating Curves for 3D surface [KST08], but are defined in image space. Higher-order features will surely need larger tensor matrix, and tends to be sensitive to the outliers, such as noises in the image and feature variation.

Our observation from aforementioned features with different orders is that in most previous methods, features are identified as maxima of a differential geometry invariant in a tangential direction. We call the loci of such maxima the feature *skeleton*. Formally, it is defined by

$$\mathcal{S} = \left\{ \mathbf{s} \in \mathbb{R}^2 \;\middle|\; \frac{\delta h(\mathbf{s})}{\delta \boldsymbol{\theta}(\mathbf{s})} = 0,\ \frac{\delta^2 h(\mathbf{s})}{\delta \boldsymbol{\theta}(\mathbf{s})^2} < 0 \right\}, \tag{2.16}$$

where $\mathcal{S} \subset \mathbb{R}^2$, $h : \mathbb{R}^2 \to \mathbb{R}$ is a $C^2$ height function, and $\boldsymbol{\theta} : \mathbb{R}^2 \to \mathbb{S}^2$ is a $C^1$ direction field. Both $h$ and $\boldsymbol{\theta}$ are easily instantiated to produce existing surface feature types. Canny edge is identified as maxima of luminance gradient magnitude in gradient direction. Ridges (resp. valleys) are obtained as maxima of $k_{\max}$ (resp. $-k_{\min}$) in the $\mathbf{t}_{\max}$ (resp. $\mathbf{t}_{\min}$) direction. And luminance inflection are obtained as maxima of $v_{\max}$ in its normalized corresponding direction.

To robustly locate features, Lee et al. [LMLH07] propose a filter-based technique that produces feature lines with controllable thickness. They find edges and ridges in shaded images of 3D objects or luminance of an image using a local 2D fitting approach. Although the method is limited to the depiction of luminance features with a simple line style, it shows that a fitting technique in image space is able to capture and render dynamic features in real-time. Our line drawing approach in Chapter 6 makes use of a fitting technique and may thus be seen as a generalization of Lee et al.'s approach that works with various images and surface features. Based on the generalized feature lines definition 2.16, we can analyze the profile attributes under different orders, and dynamically distinguish sharp and smooth feature lines.

## 2.6 Conclusion

In this chapter, we have first presented a general overview of the use of image structures in various applications, and then emphasized the importance of two image structures, gradients and contours in AR and image stylization. Finally, we have discussed the state of the art methods on estimating gradient field and extracting lines.

In the next part, we will thus first investigate further the importance of such image structures by introducing a new Focus+Context rendering of AR in Chapter 3, whose goal is to visualize hidden scenes in a video stream. For instance, a region segmentation reveals occluding orders, and lines extracted from contexts provides depth cues. We will also introduce an extended Focus+Context synthesis approach in Chapter 4, whose goal is to highlight the important regions using multiple rendering styles driven by an importance map. In this approach, we will exhibit that the importance map of an image can guide both image compositing and stylization, and the gradient field can improve the quality of various image stylization applications.

In this section, we have shown that gradient field suffers from several limitations. To fix these issues, in Part II, we will first present a new higher-order square

local approximation method for discrete non-oriented gradient fields based on a moving least square (MLS) formalism in Chapter 5. In Chapter 6, we will introduce a feature profile analysis approach for image line extraction via image-space fitting techniques. It permits to distinguish sharp and smooth features. Profile parameters are then mapped to stylistic parameters such as brush orientation, size or opacity to give rise to a wide range of line-based styles.

# Part I

# On Focus+Context compositing for Augmented Reality

CHAPTER 3

# Visualization of underground pipelines using context analysis

Visualization of underground pipelines is an X-Ray vision problem in Augmented Reality, whose task is to display hidden scenes in a real environment revealing a correct occluding order. For this purpose, we investigate in this chapter how the image and video structures contribute to provide depth cues. For instance, displaying in front moving objects of video emphasizes that the pipelines are underground, and the use of image feature lines preserves visual cues of the context in occluding region. The work of this chapter has been validated by the publication [CGLP10] and [CLLP12].

## 3.1 Introduction

Augmented Reality (AR) is classically described as the process to seamlessly and realistically integrate virtual objects into real images or videos. For this purpose, accurate camera calibration, accurate estimation of the geometry and reflective properties of each real components and accurate estimation of lighting conditions are required since any rendering error breaks the feeling of a uniform world. In some specific contexts and with dedicated devices, it is possible to obtain on-line solutions (e.g., [Ait10]) but the accuracy is still limited. Furthermore, non-photorealistic rendering (and more generally expressive rendering) is more suited when visual efficiency is required for conveying information [Hal04, FHT08] without the clutter that may result from a large amount of details due to realistic rendering.

In this chapter, we are interested in on-line visualization of objects that are hidden by real occluders issued from an image or a video (such as underground pipes, organs in a body [KBM+08], engines in a car [KMS09], ...): this is an important issue of AR [FAD02]. By definition, these invisible objects are always inside of or behind the occluding scene and they do not share the same lighting conditions with the real data. Moreover, such an augmentation, close to classical illustration,

should focus on conveying information such as depth, characteristic structures and features, and spatial relationships. In this chapter, we focus on the visualization of underground infrastructures, although the potential applications for this research are much broader, including technical illustrations, computer aided education and visualization of medical volume data.

Making "invisible" objects "visible" requires to (i) make sure that all their characteristic structures are getting clearly and legibly visible in the final image in order to be easily identified, and (ii) clearly and legibly provide some visual cues to illustrate that this object is behind the 3D scene that is, to show a correct spatial relationship. This problem is identified as the Focus and Context (F+C) rendering [KMS09]. In theory, the requirement (i) is trivial to achieve since this only requires to render the virtual object on top of the real image. Unfortunately, such an approach does not provide any cue about its relative positioning in the 3D scene, not fulfilling (ii), as illustrated in Figure 3.1. Simple transparency provides occlusion cues [LSG+03] but needs to be completed by other cues to not confuse the user [FAD02] and to understand the relative order. One possible metaphor in AR is X-Ray vision (e.g., [AST09]). For pure 3D models, Cutaway techniques [BF08] and volume rendering [KSW06] share the same goals. In most of these solutions, the transparency is efficiently replaced by compositing different layers [BRV+10] with different styles according to their content (main focus or context) and characteristic structures. In this chapter, we explore this approach for on-line video-based AR without the requirement of 3D reconstruction of the environment.

To achieve the expected results, we first introduce a depth-ordered frame segmentation for scenes with moving objects according to visualization purpose (Section 3.3). We then demonstrate in Section 3.4 that the introduction of masks which preserve the video structures creates some legible cues for the occluding order. Finally, we have experimented two different cutaways, and show that particular designed rendering of the virtual objects (both photorealistic and non-photorealistic rendering) and well-tuned contrast enhancement participate to the legibility of such cues. We demonstrate our on-line framework on several images and videos and discuss its current limitations and potential improvements for future work in Section 3.5.

## 3.2 Focus+Context based visualization in Augmented Reality

The visualization of hidden objects is a hot topic in AR since a long time. One of the main problems is to convey the difference between what is not visible and what would be visible [FAD02]. As an example, Bajura et al. [BFO92] have developed a technique for ultrasound medical imagery: the hidden organs appear in front of the occluder (the body skin). Enhanced visualization makes use of a cue similar to 3D cutaway but still displayed in front. Recently, Schall et al. [SMK+09] have used a similar rendering technique for handheld AR system for visualization of underground infrastructure. Most of these simple cutaway techniques use a sim-

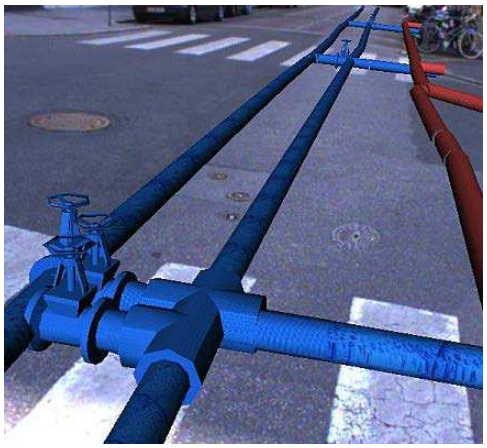


**Figure 3.1:** ***Direct overlay*** *does not provide viewers a correct occluding order, thus raises a depth illusion that the underground pipelines are floating on the ground.*

ple binary mask between the image and the virtual object, resulting in a loss of context cues. Some researchers have improved this technique to complex AR task. Mendez et al. [MKS06] have introduced an interaction tool that extends global information filtering [JBBL02] using Context Sensitive Magic Lenses, which works with highly complex AR scenes such as geo-data models. Such an approach efficiently allows the visualization of depth cues on 3D components, but failed in preserving the context issued from the videos.

In order to preserve the context, earliest work made use of simple wire frames to render hidden objects. Feiner et al. employed such a technique for laser printer maintenance [FMS93] and for architectural components [FWK+95]. With these solutions, the hidden object still appears in front of the occluders, inverting the depth order.

Another simple solution is to use transparency (e.g., [KTO04]). Despite being efficient to convey that there is an occlusion [LSG+03], it can be confusing when the number of layers increase [FAD02]. Moreover, it cannot convey alone the layers' order. Indeed, Furmanski et al. [FAD02] have shown that it is only one of the five depth-dependent perceptual cues: occlusion events, texture scaling, shading gradient and other cues such as shadows improve the depth perception. They also have introduced some guidelines for developing effective visualization of occluded information.

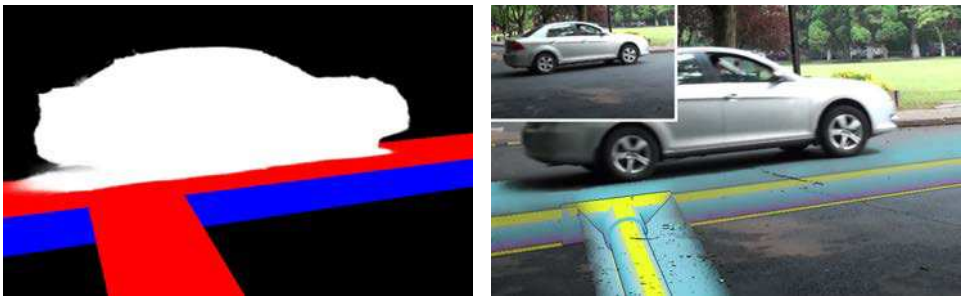
Among all the previous techniques, the most successful one is the X-Ray Vision. It has been defined as "the act of visualizing some target through at least one layer of occluding structure" [BH04]. Bane and Hollerer [BH04] have thus introduced the *Tunnel Tool* in which the invisible components are rendered when span in a frustum-cut (the tunnel) and the context (geometry of the occluders) is rendered using wires. Similar to cutaway [CH06], the lack of accuracy in the camera tracking can introduce incoherent motions between real and virtual objects [FAD02]

and thus lower down the perception of depth order since the wires seem to float on top of the occluding objects [AST09]. In their work, Kalkofen et al. [KMS09] have proposed an interactive context preserving Focus and Context (F+C) visualizations. Similarly, their approach requires a 3D reconstruction of the occluders and an accurate camera tracking. To overcome this problem, the authors propose to use simple edges from videos: but they are not as clearly defined as ones from 3D objects and they can be either too cluttered or not noticeable due to lack of contrast with hidden objects. In this chapter, we propose some improvements and others solutions to reduce this problem.

The best results in X-Ray Vision for AR are the one from Avery et al. [AST09]: they show how multiple view modes can provide depth cues and spatial awareness, and how edge overlay and tunnel cutout can convey the distance between the user and hidden objects. They use also edge detection to render the occluders, limited to brick walls in their experimentations. Despite being powerful, having too much edges might increase the difficulty for users to understand the 3D scenes [DCS10]. Thus, edge extraction should be controllable to avoid visual clutter.

Works in 3D visualization community also have proposed many ideas for suggesting depth-order. Their processing data, pure 3D or volumetric data, is different from AR community, but they shared the same main idea, that is revealing the depth cue of the hidden objects while providing the most important structures of both hidden objects and external surface (or real-world data in AR). In their "ClearView" system, Krueger et al. [KSW06] use the attributes of the surface, like curvature, to modulate the foreground transparency and thus preserve the context. Modulating the transparency has also been used by Bichlmeier et al. [BWHN07] taking into account curvature, view angle and distance falloff. The resulting partial occlusion of the focused object provides the user with an improved perception of depth. Viola et al. [VKG05] extend automatic F+C and cutaway techniques to volumetric data, introducing screen-door transparency. Recently, Bruckner et al. [BRV+10] have introduced a framework using rendered 3D layers instead of 2D layers, which can provide artists many extended compositing operators, such as selective transparency, occlusion overrides, and soft depth buffering. The GPU-based interface makes its visibility compositing and masking as easy as image alpha blending processing controlled by an alpha function. Unfortunately, their solution is tied to knowledge of the 3D environment and thus not directly usable in our on-line context since the reconstruction of the 3D environment is still hardly achievable. These approaches illustrate nevertheless the need of an efficient extraction and visualization of context structures to preserve depth cues.

In our approach, we introduce two different cutaway methods, and explore the use of recent trends in structure estimation from videos to create improved context visualization. We also ensure that during the final composition process all the cues from the context are legibly rendered. And, we focus on on-line techniques and do not rely on 3D knowledge of the real environment.



(a) (b)

**Figure 3.2:** ***Depth-ordered segmentation*** *(a) for the result using tunnel-based cutaways (b), the original image is on the top left corner. The red and blue regions are respectively the main and second focus region. The blue region corresponds to the cut geometry in front of the virtual object. Context cues are extracted from the video outside the white region that is the moving objects: they are in front of the virtual object.*

## 3.3 System overview

Our system targets the following scenario: the camera is fixed and the geometry of the real environment is mostly unknown. This last assumption is very important for out-door scenes that are constantly evolving on-line reconstruction of 3D environment is still not a mature technology. In such a scenario, we want to display hidden objects while preserving important contextual information issued from the video.

Inspired by cutaways popularly used in pure virtual reality applications and under the assumption that the geometry of the real environment can be approximated as a ground plane, we use two different configurations: 1) trapezoidal **tunnels** along the pipelines that better display the spatial distribution of the underground pipelines, and 2) vertical **sections** cutting the pipelines that are able to precisely reveal the depth of underground pipelines. For a sake of legibility, the description of our approach will be illustrated on the tunnel-based cutaway. Results from both configurations will be shown in Section 3.5. Instead of directly using the occluding cues introduced by cutaways [BH04], we identify four depth-ordered regions (a tunnel-based cutaway example is shown in Figure 3.2) that we detail in the following.

### 3.3.1 Focus

The focus of all the rendering is the virtual object, located behind the real scene. To improve depth perception, we add to the virtual object the geometry corresponding to the cutaway, either tunnel-based or section-based one. This is trivially done thanks to the fact that, contrary to the real scene, the object's geometry and position are more easily known. According to the visualization request, the virtual object

can be rendered in either photorealistic or non-photorealistic manner. Comparison and discussion of these two manners will be presented in Section 3.5.

Thanks to the integration of the cut geometry, we can divide the focus region covered by both the object and the cut in two. The first region (red in Figure 3.2) corresponds to the potentially visible parts of the object if a real cutaway hole was created to reveal it: we call it the **main focus**. Technically, it is easily identified using back-face culling on the combined object and cut geometry.

The second region (blue in Figure 3.2) corresponds to parts of the cut that occlude the object (the back-faced surfaces of the cut in front of the object): we call it the **second focus**. In order to suggest that this layer is in-between the main focus and the scene, we use a lower transparency in this region.

### 3.3.2 Context

Similar to the focus region, we subdivide the context region (the video frame) in two. The first contains the moving objects that are segmented out (white in Figure 3.2): the **moving context**. This segmentation may be done on-line [GWYY10]. As proof of concept, in this thesis we use an improved and off-line version of the on-line solution [ZQC+10] as preprocess. Since the moving objects are mostly in front of the scene and the virtual object, and in order to provide *structure-from-motion* [FAD02] cues, we render this layer as-it on the top of all the others: the occlusion events resulting from the movements reinforce depth-order understanding.

We call the frame region that does not contain the moving objects the **static context**. The realistic order of these four regions is: moving context, static context, main focus and second focus, from close to far. In the hidden scene's visualization purpose, we adjust this order as: moving context, main focus, second focus and static context. Thus, the virtual object is cut in static context region. We use this region to extract characteristic structures (such as edges, gradients, second order derivatives, ...) in order to re-introduce them as depth cues in the spirit of X-Ray Vision in front of the 3D object. The use of these image and video structures is detailed in the following section.

## 3.4 Image layers and compositing masks

Thanks to the depth-ordered segmentation, we define four different rendering processes to suggest the correct occluding order. We rely on simple tools to compute five layers and two masks (Figure 3.3 and following paragraphs). To efficiently preserve context cues from the video, these masks use structural information from both the video and the virtual object.

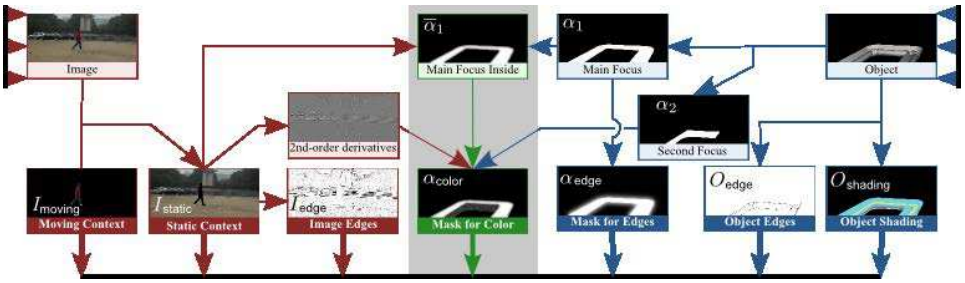


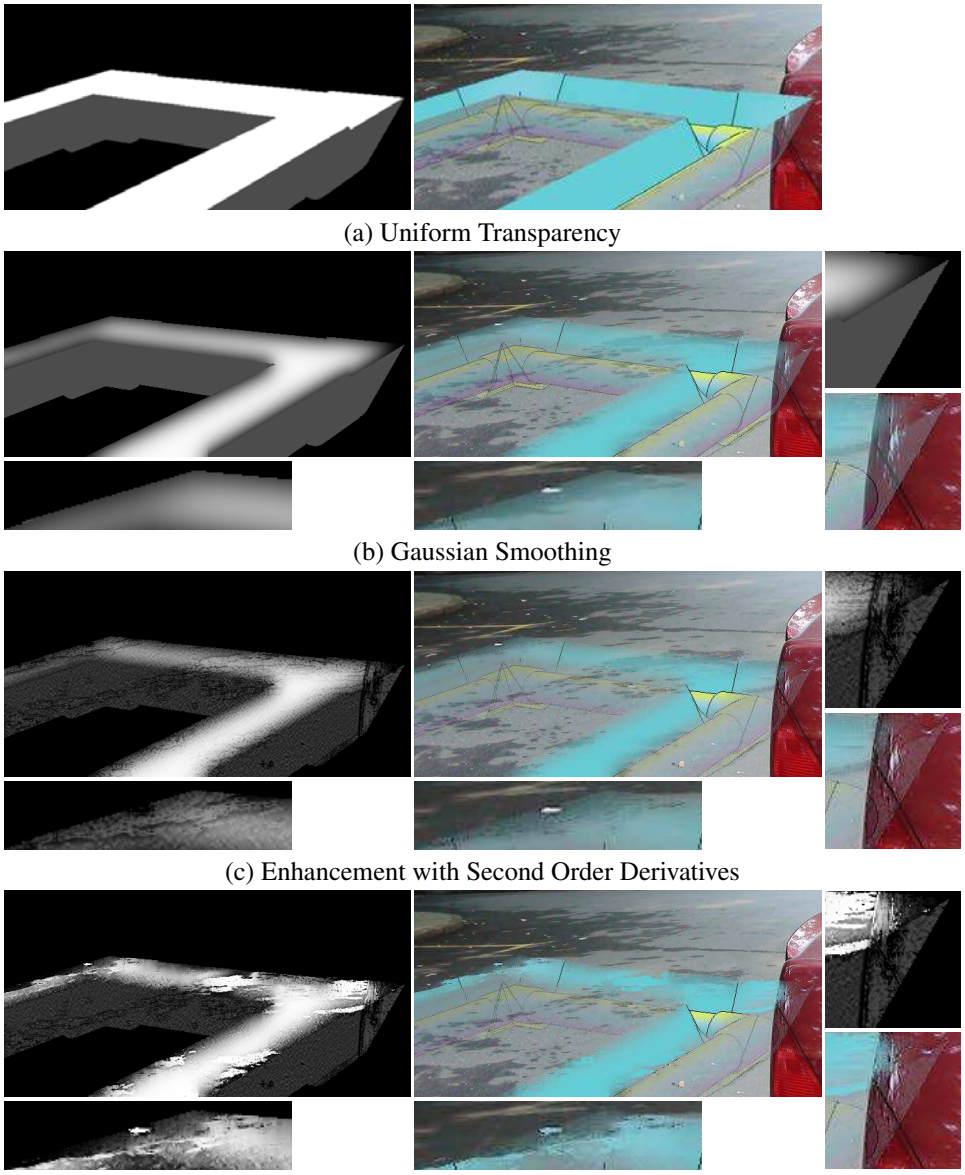

**Figure 3.3:** ***Our framework*** *extracts different layers and masks to composite. From the video (red paths), we extract three layers* $I_{\text{dynamic}}$*,* $I_{\text{static}}$ *and* $I_{\text{edge}}$*. From the virtual object (blue paths), we extract one mask* $\alpha_{\text{edge}}$ *and two layers* $O_{\text{edge}}$ *and* $O_{\text{shading}}$*. The last mask* $\alpha_{\text{color}}$ *is a hybrid one (green paths) that combines both video and object information to create the final cues.*

### 3.4.1 Layers and Masks

From the videos, three layers are issued. Their main goal is to provide some visual cues about the context. The two first layers contain different components of the video frame. The "static" part of the video $I_{\text{static}}$ is the main layer: this corresponds to the main context since it represents the real 3D scene and the main occluders of the virtual object. This layer is mostly rendered without any changes in the regions that are not covered by the virtual object and the cut geometry. The second layer $I_{\text{moving}}$ contains the moving objects of the video. Generally in front of the real scene and thus of the virtual object, this layer is rendered directly in front of all the other ones. Finally, the third one corresponds to image edges ($I_{\text{edge}}$, using a GPU implementation of flow-based operator of Kang et al. [KLC07]) as abstracted visualization of the static context.

From the virtual object and its corresponding cut geometry, two layers and one mask are issued. The two layers are the shaded object ($O_{\text{shading}}$) and its 3D edges ($O_{\text{edge}}$). Note that we do not take into account silhouette edges of the cut geometry (see Figure 3.5): they visually conflict with the smooth transition that we introduce later. To blend the image edges in the final image, we compute one mask from the virtual object: $\alpha_{\text{edge}}$. It identifies the main focus area ($\alpha_{\text{edge}} = 1$, as shown in Figure 3.6) with a smooth transition on its outside border: it is used to blend the edges from video.

Finally, the last mask and core contribution is issued both from the video and the virtual object $\alpha_{\text{color}}$ (shown in Figure 3.4). Similarly to the edges' one, it also identifies the main focus ($\alpha_{\text{color}} \neq 0$), but with the transition increase toward 1 on its inside border: it is used to blend the video frame with the virtual object.



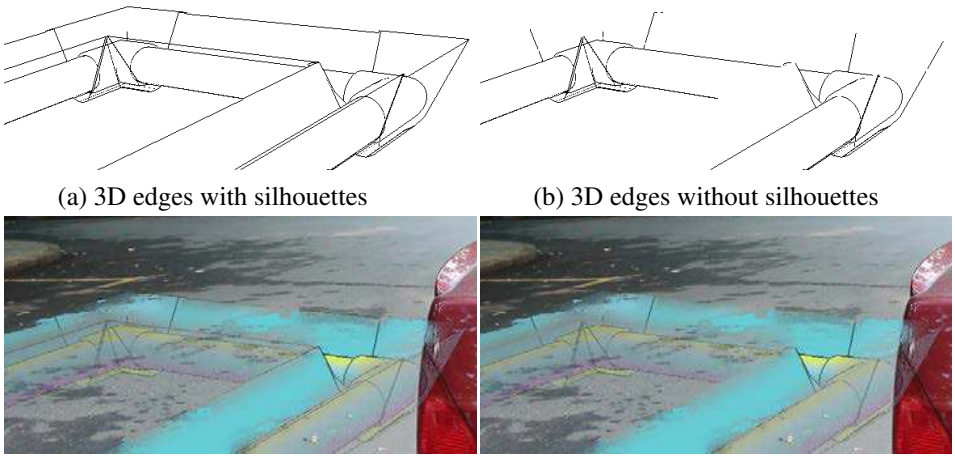
(a) Uniform Transparency

(b) Gaussian Smoothing

(c) Enhancement with Second Order Derivatives

(d) With Cross Bilateral Filtering

**Figure 3.4:** ***Mask for color blending.*** *From upper row to bottom one: uniform transparency, smoothing the edge inside, with modulation by image second order derivative, with using cross bilateral filtering. Note that modulation by curvature tends to preserve discontinuities and that cross bilateral filtering tends to preserve uniform regions (like the upper-left dark mark on the road).*



(a) 3D edges with silhouettes (b) 3D edges without silhouettes

(c) Result with silhouette edges (d) Result without silhouette edges

**Figure 3.5:** ***Computation of 3D edges from the virtual object:*** *the silhouettes of the cut may introduce unwanted discontinuities in the smooth transition created by the masks and connect un-wanted areas (such as the back of the car and the ground).*

### 3.4.2 Color Mask

Previous researches have shown that simple binary masks or simple transparency is not sufficient to visually convey the depth order in X-Ray Vision. Nevertheless, Krueger et al. [KSW06] have shown that modulating the transparency by the surface curvature helps in preserving the depth cues in their Clearview system. Viola et al. [VKG05], introducing the *screen door* effect, and Mendez and Schmalstieg [MS09], introducing *importance masks*, have shown that a smooth transition of transparency between the context and focus area also participate in preserving such cues.

Inspired by these approaches that rely on the knowledge of 3D geometry, we propose a solution for video-based AR that does not require any explicit 3D reconstruction of the real environment. For this purpose, we first modify the binary mask that identifies the main focus area, smoothing its mask $\alpha_1$ inside (to create a smooth transition from 0 to 1 where $\alpha_1 = 1$ as shown in the top middle image of Figure 3.4). We denote $\overline{\alpha}_1$ the inside smoothed mask. This results in a progressive removal of the context image on top of the virtual object. Furthermore, in order to reinforce the order cue, we also modulate the transparency using the curvature-like (i.e., second-order derivatives or gradients) information from the image to obtain similar effects like Viola et al. [VKG05] on 3D surfaces (see the third row of Fig-

ure 3.4):

$$\alpha_{\text{color}} = \overline{\alpha}_1^{1+\beta\left(|L_{xx}|+|L_{yy}|\right)}. \tag{3.1}$$

In this formula, $\beta > 0$ is a user-control parameter for the modulation strength, $L_{xx}$ and $L_{yy}$ are the second-order derivatives of the luminance $L$ of the static context $I_{\text{static}}$. With such a formula, image areas with strong image structures (high second derivatives) are kept on top of the virtual object. We also modulate the transparency in the second-focus area in order to preserve the same cues. However, to distinguish both areas, this transparency is initially set to a user-selected constant value $\gamma$ less than 1, which reveals the fact that the second focus is identified as back-faced surfaces. The final mask is thus

$$\alpha_{\text{color}} = \left(\overline{\alpha}_1 + \alpha_2\,\gamma\right)^{1+\beta\left(|L_{xx}|+|L_{yy}|\right)}. \tag{3.2}$$

Second order derivatives are only one of the potential image structures and convey ridge/valley-like information. Further improvements are obtained using cross bilateral filtering [ED04, PSA+04] with $I_{\text{static}}$ instead of a simple Gaussian filter for the estimation of $\overline{\alpha}_1$ (see the two last rows of Figure 3.4). While the transparency modulation preserves the image discontinuities, cross bilateral filtering preserves the uniform areas. Note that we use a small variance 0.05 for the Gaussian weights: such a choice has experimentally leaded into better visual results.

### 3.4.3 Edge Mask and Final Blending

Edges are classical and very strong image structures. We are using the *Edge Overlay* of Avery et al. [AST09]. Similarly to this work and in order to prevent a lost of connection with the context due to the brutal transition from image to edges, we use a smoothed mask. But contrary to the previous work and in order to distinguish the effect from the color transition, we use a Gaussian filtering outside of $\alpha_1$, only where $\alpha_1 = 0$ (see Figure 3.6).

Finally, a pixel color is $I_{\text{moving}}$ in areas containing moving objects, and is resulting from the use of the different masks and layers elsewhere:

$$\text{mix}\left(\boldsymbol{C}_{\text{edge2d}}, \text{mix}\left(\alpha_{\text{color}}\, O_{\text{shading}}\, O_{\text{edge}}, I_{\text{static}}, \alpha_{\text{color}}\right), \alpha_{\text{edge}}\left(1 - I_{\text{edge}}\right)\right). \tag{3.3}$$

$\text{mix}(x, y, \alpha) = \alpha\, x + (1 - \alpha)y$ is a linear blend between $x$ and $y$, and $\boldsymbol{C}_{\text{edge2d}}$ is the color of the edges extracted from the image, we use black color by default.

Unfortunately, it is still difficult to ensure with such alpha-blending that the image edges are clearly visible in the resulting image [KMS09]. Improving their legibility is important for a better preservation of the corresponding context information and thus the depth-order cues. For this purpose, we use bilateral filter operator [ED04] smoothing the input image before composition optionally and the unsharp masking operator [BA04] locally around the edges (few pixels around, as shown in Figure 3.7).

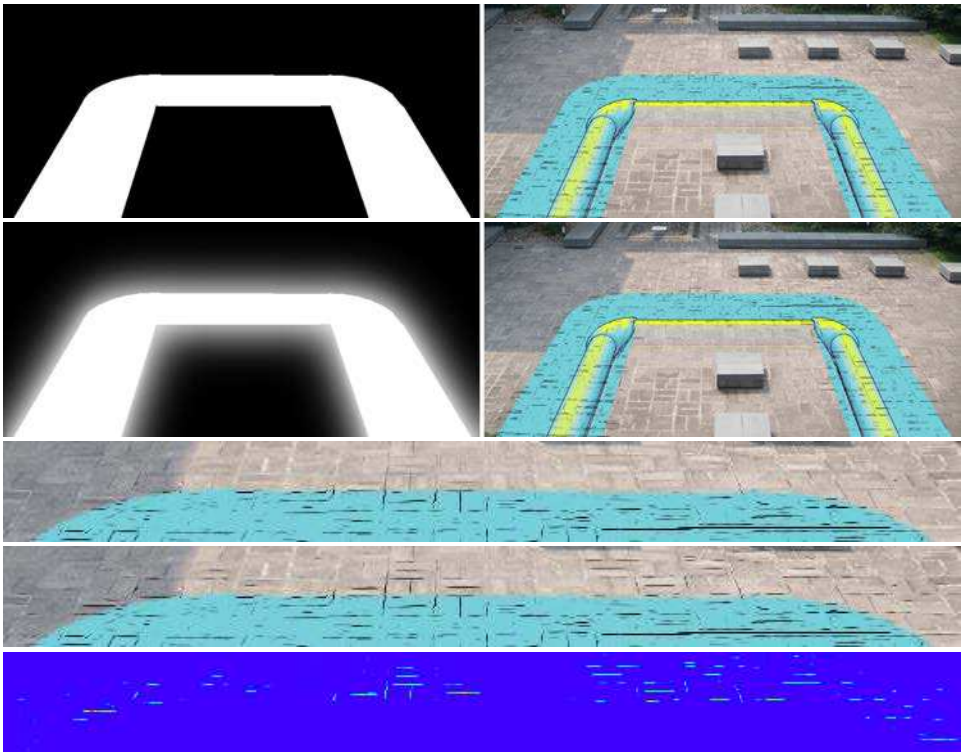


**Figure 3.6:** ***Edge Blending.*** *This figure shows both the mask without and with smoothing outside, and their results on the final compositing. The three last rows show a zoom on the upper part of two results and their Lab difference.*

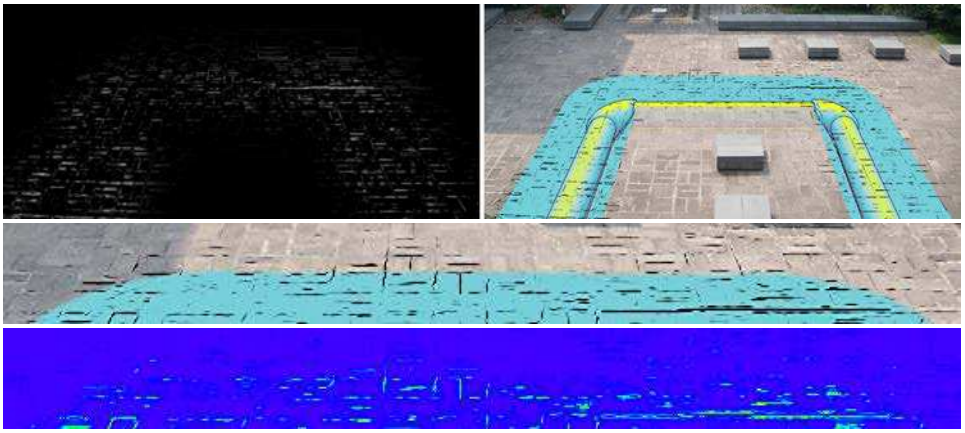


**Figure 3.7:** ***Local Unsharp Masking.*** *Unsharp masking is applied around the edges (in white areas in the left image) to enhance their legibility. The two last rows show a zoom on the upper part and a Lab difference comparing to the original image.*

## 3.5 Results and discussion

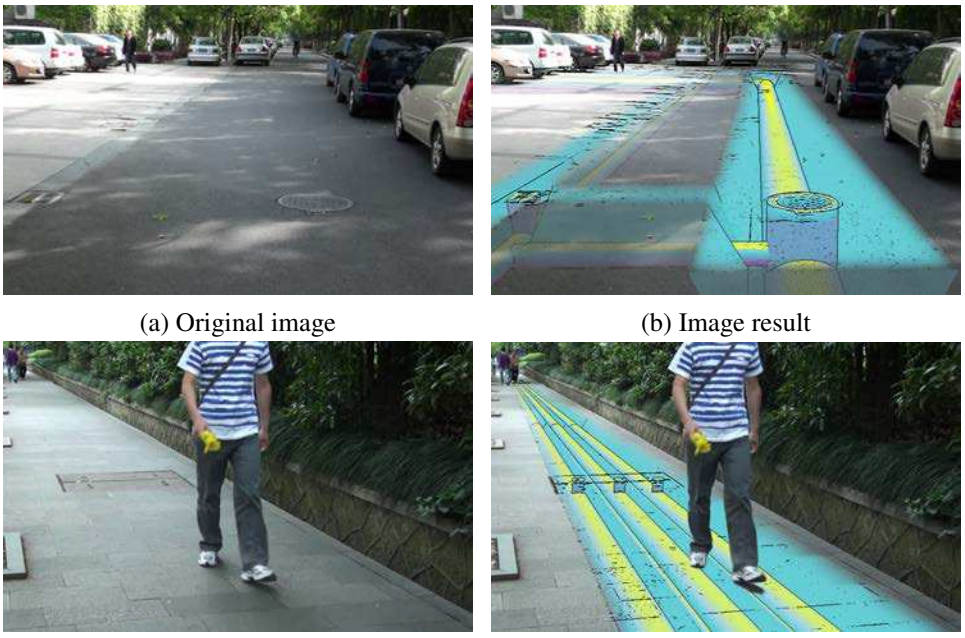
**Tunnel vs. section:** We have implemented two different cutaways. One is trapezoidal tunnels along pipelines like Figure 3.8, and the other is vertical sections cutting pipelines, like Figure 3.9. These two cutaways share the same framework above, except that the section-based one defines second focus as regions where pipelines are on the invisible side of the section, instead of back-faced surfaces of the cutaways. Tunnel-based cutaway generates tunnel everywhere pipelines exist, thus displays the spatial distribution of the pipelines clearly. But, it tends to fail where pipelines are dense or far from the camera. Compare to tunnel, section introduces much more accurate depth cues, and is still distinguishable for dense pipelines. And multiple sections can be used to satisfy complex visualization tasks, some examples in Figure 3.9-(b,d,f). The main limitation of section-based cutaway is that the depth cues are strong only near the section.

**PhotoRealism (PR) vs. Non-PhotoRealism (NPR):** The choice of rendering method depends on the purpose of the visualization, thus is related to the request of the user. It is worth to notice that hidden scenes don't share the same lighting condition with the real environment. To investigate the validity of our approach, we have experimented both photorealistic rendering (PR) and non-photorealistic rendering (NPR). For NPR, we use a simple X-toon shader [BTM06] to enhance the joint visualization of the shape and the depth with lines extracted from the depth-map. In Figure 3.8, bright yellow is used to ensure the clarity of the pipeline routing, and darker colors are applied to report the pipeline shape. For PR, we first attach cutaways textures with red height lines, and then add artificial directional lights to make underground pipelines cast shadows on the sections when they go through the section surface. Figure 3.9 shows that this PR design seems to provide accurate depth cues.

We have experimented different scenes, some with easily-extractable edges (Figures 3.6, 3.9 and 3.8-(d)) and some others with less strong patterns (Figures 3.4, 3.8-(a) and (c). We also have run our system on simple images (Figures 3.8-(c), 3.9-(a,c,e) and 3.6) and video from which we have segmented-out the moving objects (Figures 3.8-(b,c) and 3.9-(b,d,f)). The resulting videos and some other images can be seen in the companion videos.

One of the most noticeable results is the fact that we seem to reach the objective of a correct representation of depth order. Thanks to the color mask and the use of both bilateral filtering and second order derivative modulation, we better preserve the existing discontinuities in the original image (there can be some occlusion events) and uniform areas without the need of any 3D reconstruction. This is illustrated in Figure 3.4.

Beyond the occlusion order, the use of edges can help in understanding the positioning where the hidden object is. As examples, in Figures 3.9 and Figures 3.8, the edges show the real place of the pipe exits. This understanding is also reinforced in video by introducing the moving objects on the top, as show in Figures 3.9-(c) and (e) and Figures 3.8-(d).



(a) Original image (b) Image result

(c) One frame of original video (d) Video result

**Figure 3.8:** ***Visualization of underground pipelines using tunnel-based cut-aways*** *on an image (a) and videos with moving objects (c). The depth order is suggested by displaying structural features from images such as edges over the virtual objects, by introducing image-depend smooth transition for transparency, and by preserving the moving object in front (d). The frame rate ranging from 18 to 25 fps depends mostly on the size of filters for edge extraction and transparency smoothing.*

In our implementation, we use the robust flow-based technique [KLC07] to compute edges from $I_{\text{static}}$. This is also quite costly (between 9 and 14 ms), and a simple operator such as Canny [Can87] will reduce the performance impact. However, we believe that in future, the extracted flow might be used as a valuable image structure to improve the legibility of our current cues.

All the results are computed on a PC workstation with a NVIDIA GeForce 460 GTX, an Intel(R) Core(TM)2 Duo E6550 @ 2.33 GHz and 4 GB of memory. Due to its pixel processing nature, the achievable frame rate depends on the video/image resolution and on the size of the different filters. The most costly step is the cross bilateral smoothing used to create the two masks: it takes between 14 and 37 ms per frame with filter size quite large ranging from 90 to 105 pixels in Figure 3.8.

## Discussion

In our current implementation, we assume a fixed camera. In theory, our framework can be used for moving cameras, and since we do not require any 3D reconstruction

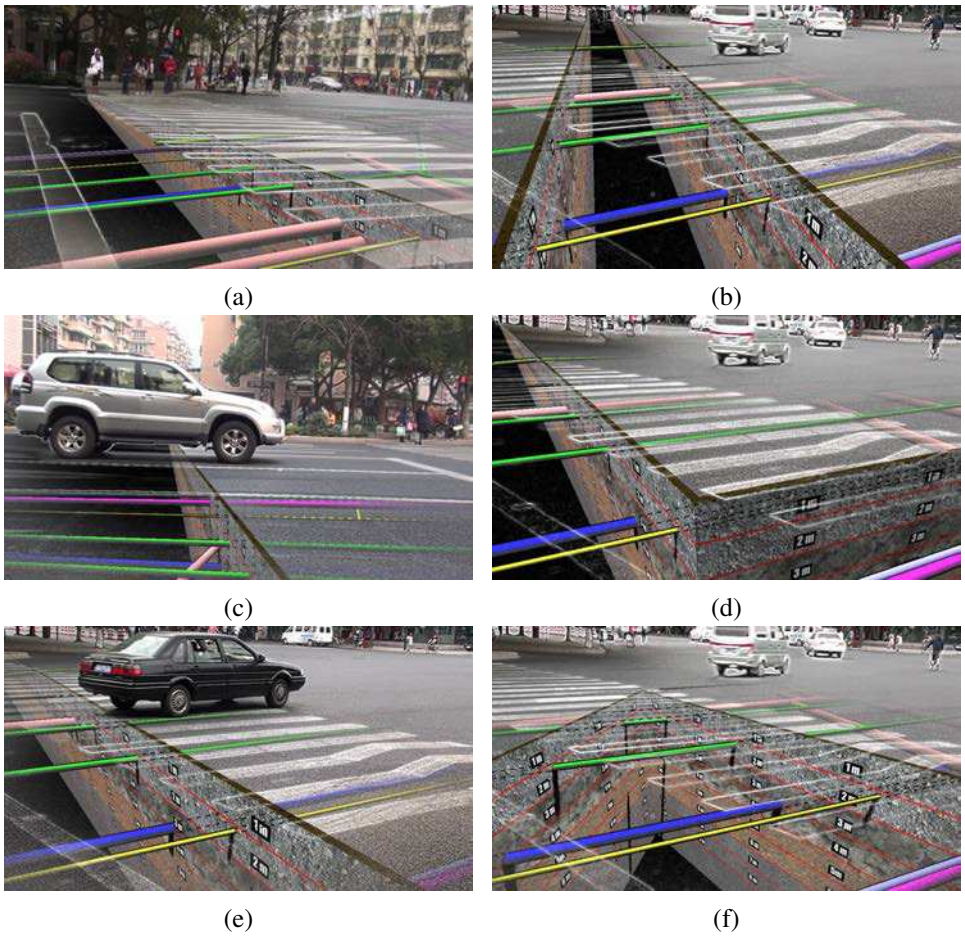


**Figure 3.9:** ***Visualization of underground pipelines using section-based cutaways*** *on images (a,c,e) and videos (b,d,f). Elaborative section textures and pipelines' shadow on the section provide very accurate depth cues. Image edges are white in order to be different from black object edges, they clearly reveal the occluding orders. Object edges occluded by the section are dotted to hint their invisibility. Segmentation of moving object in (c,e) further emphasizes correct occluding orders. The frame rate ranging from 22 to 45 fps depends mostly on the size of filters for edge extraction and transparency smoothing.*

of the real environment, the classical misalignment between reality and virtuality is less important than in systems that rely on such a reconstruction. However, the problem is still present and needs some improvements in camera tracking.

Our system does not need any 3D reconstruction of the environment and this leads to an easy to implement on-line solution (except the segmentation of dynamic objects) since there is no need in preprocessing to get the real 3D scenes. Unfortunately, this is also one of the limitations: better coherency and more accurate

geometry of the cut might be achieved with a better knowledge of the 3D environment. One direction is to move from simple video streams to stereo cameras in order to acquire on-line an approximation of the real scene, or to improve the extraction of 3D cues into the image [KCB+05]. As with any AR systems, such future work will result in improved solutions.

Finally, the current framework relies on a number of parameters (the different sizes of filters, the strength of the transparency modulation and of the unsharp masking, the transparency of the second focus area and all the parameters related to extraction of edges and the shading of the virtual object, etc). We believe that they are sufficiently comprehensible for most users to obtain quickly a convincing result. However, in order to reach the most visually efficient set of parameters, some user studies have to be done.

## 3.6 Conclusion

In this chapter, we have introduced a new framework for visualization of underground pipelines that improves X-Ray vision. By extracting image and video structures such as edges, second order gradients, and by creating some masks using both the virtual object and video frames, we manage to create visual cues that suggest the depth order while reveal the virtual object hidden behind the scene. Thanks to this approach, we achieve near real-time performances and good quality without the requirement of 3D reconstruction. We demonstrate the validity of our framework on two different cutaways, and believe that such a framework can be easily adapted to visualize other hidden objects and will take benefit from future work in on-line video processing.

Context structures play a key role in revealing depth cues of hidden scenes. A moving object segmentation provides a strong visual cue that the virtual pipelines are behind this object, thus under the ground. As an example, segmenting out the car in Figure 3.4 to use it as a layer in front of the 3D scene will definitively improve the result. Similarly, extracted lines preserve the object contours for the real environment with few spatial occupation. They provide visual cues to position registration between underground pipelines and real objects on the ground.

One limitation of our approach is the lack of coherence of image structures, including spatial coherence for images and temporal coherence for videos. In images with apparent features, such as zebra crossing lines in Figure 3.9, a simple line extraction method like Canny edges is efficient. An edge-preserving smoothing can be applied to remove image noises before any other processing. However, it is difficult to extract continuous lines in images with weak feature evidence, such as Figure 3.2-(b) and Figure 3.8-(b), although we have employed a flow-based line extraction technique. The same situation is observed for other structural features, like second-order derivatives.

Flickering sometimes is observed around extracted lines and segmented boundaries in video results of our approach. In theory, our frame-by-frame approach

naturally has this limitation, but it is more observable for video with low evidence of structures. Motion flow is known to be helpful to improve the temporal coherence [BNTS07, LLY09], a simultaneous use of gradient field and motion flow may result in a significant improvement of both spatial coherence and temporal coherence.

CHAPTER 4

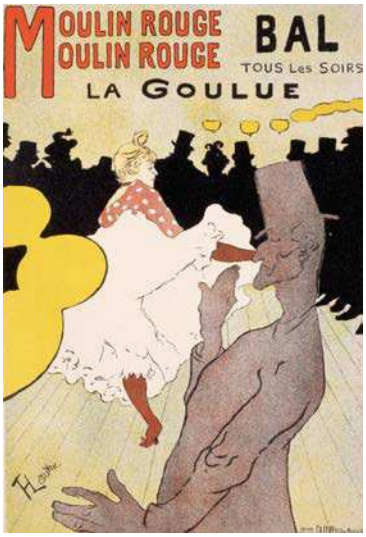
# Importance-driven image synthesis

In the previous chapter, we have shown that the use of image and video structures in compositing tasks for Augmented Reality applications improves the depth cue of hidden scenes. We introduce a non-uniform image synthesis approach that extends this idea in this chapter. The synthesis, driven by an importance map, is a composition combining multiple rendering styles. We demonstrate that image structures can directly and indirectly contribute to the quality of the importance map and stylized rendering. Besides Augmented Reality applications, we also illustrate the versatility of our approach for images, videos and pure 3D scenes. The work of this chapter has been validated by the publication [CCG+11] and [CCPP12].

## 4.1 Introduction

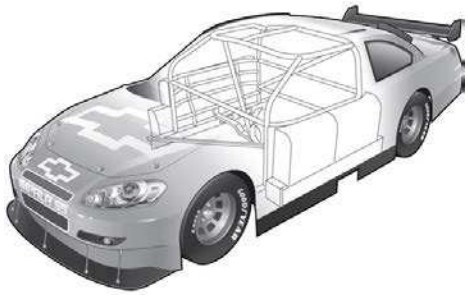
The use of multiple styles has been adopted in many artworks, mostly modern ones, to highlight the important meanings. However, it can only be conducted by skillful artists. Some examples are shown in Figure 4.1. DeCarlo and Santella [DS02] cite Toulouse-Lautrec's poster as an example to show that different regions can be abstracted at different levels. We also cite this poster in the left of Figure 4.1 to show that artists also employ significantly different styles for different regions. In the other two pieces of artwork in the right, different styles, such as watercolor painting, line drawing, luminance, are combined in different but reasonable manners to exhibit the creators' special design intents for different contents, which also presents the maximal of visual information.

In Computer Graphics, many rendering papers focusing on a single style have been published, while few works consider multiple styles. Focus+Context rendering methods in Augmented Reality render focus and context differently to reveal the hidden scenes, as detailed in the previous chapter. Take visualization of underground structures discussed in last chapter as example, only feature lines of the real environment are preserved in the occluding regions, while the virtual underground




©Henri de Toulouse-Lautrec

©Luis Ruiz



©Vic Kulihim Illustration

**Figure 4.1:** ***Three examples of artistic illustration with multiple styles.*** *The poster in the left, "Mouline Rouge-La Goulue", adopts detailed painting to exhibit the pose of the woman dancer and the material of her dress, uniform color plus line drawing for her partner, silhouette filled with black color for all the spectators in the background. In the right-top painting, the cathedral painted with watercolor stands out from surrounding building drawn by lines, and shows a clear overall arrangement of this city. In the right-bottom illustration, the line drawing in the middle of the car exhibits the interior structures while the luminance in the head and the tail shows the global appearance of the car.*

pipelines are fully rendered expressively with an auxiliary cutaway. Image abstraction techniques [DS02, SD02, WOG06, KLC09, ZMJ+09, KKD09] adopt edge-preserving smoothing that gives the freedom to omit or remove visual information under the guide of user interaction, eye tracking data or automatic saliency estimation. Level-of-Detail rendering of 3D scenes [LRC+03, NJLM06] is achieved by either pre-building the hierarchical structure of the mesh or adjusting the size of strokes adaptively. Unfortunately, they severely restrict the choices of styles. A pioneer work to extend this compatibility is called stylized focus [CDF+06], which directs the attention of viewers to areas of interests in architectural rendering of 3D models. User can select several elaborate focus models and rendering effects, but gaze-based importance definition can not satisfy complex compositing tasks.

In this chapter, we introduce a non-uniform image synthesis approach that integrates multiple rendering styles in a picture driven by an importance map. This map, either issued from saliency estimation or designed by a user, is introduced both in the creation of the multiple styles and in the final composition. Our approach independently renders the images, videos or 3D scenes with different styles

selected by users into texture buffers, which are finally composited in screen space. It shares a lot of similarities with the classical compositing methods, but brings three key advantages:

- It composites multiple rendering styles non-uniformly guided by an importance map instead of one or two styles;
- It can process different input data, such as images, videos, 3D scenes, or even mixed reality data;
- It is compatible with different importance maps, even user-created or user-edited ones.

The goal of this chapter is not to introduce a new rendering style, but an importance-driven approach using multiple existing styles. We introduce our general approach on an image example in Section 4.3, and then describe all the details of each step in our synthesis pipeline in Section 4.4 and 4.5. We also demonstrate the compatibility for various input data, styles, and applications in Section 4.6. The final Section 4.7 presents a conclusion and potential future extensions.

## 4.2 Previous work

Most of the previous work only combines two styles: on one side the lines and strokes and on the other side, color abstractions. DeCarlo and Santella [DS02, SD02] first proposed the idea of "image abstraction" that offers such a style with a spatially variant abstraction omits or removes redundant visual information. Orzan et al. [OBBT07] uses the Gaussian scale space theory to deal with the discontinuity problem in computing a perceptually meaningful hierarchy representation. Compare to general image stylization, image abstraction differs from its goal of efficient visual communication, rather than mimic artistic effect.

Inspired by these pioneer works, many extensions on image or video abstraction have been published [WOG06, KLC09, ZMJ+09, KKD09, KL08]. In general, they reduce the contrast in low-contrast regions while increase the contrast in high-contrast regions. Different from the hierarchical structure analysis used in [DS02, SD02], they employ various image filters, such as edge-preserving smoothing filter, Flow-based Difference-of-Gaussian (FDoG) filter, shock filter and Kuwahara filter, to achieve the goal of reducing or increasing the contrast. Recent techniques take the advantages of faster performance with parallel implementations and improved temporal coherence, but still severally restrict in a particular rendering effect.

All these techniques are based on an underlying importance map either extracted from eye-tracker data [SD02], or from information extracted from images. One more and more popular way to extract visual importance information is to compute a saliency map [IKN98, LSnZ+07, ZMJ+09, BZCC10]. These techniques detect low level features (such as intensity variance, orientation and color contrast, texture difference) and higher level features (such as horizon lines, human

faces, cars) using bottom-up computation, but they don't match actual eye movements [JEDT09]. Thus, Judd et al. [JEDT09] introduces the eye tracking data to learn a saliency model to predict where humans look. Zeng et al. [ZZXZ09] analyzes a top-down semantics of the image by image parsing, but requires much more user interaction to segment and label the image. Kang et al. [KCC06] defines an importance map based on outline, and controls the abstraction level by adaptively adjusting the stroke attributes.

Using saliency or any other perception-based importance map only allows to control the level of stylization according to what a common user will perceive in an image. A wider range of applications are possible if we want to use such a map to control the gaze. Presented in the last chapter, Focus+Context rendering is a well-known expressive rendering mechanism in Augmented Reality (AR) and Volume Rendering domain. Kalkofen et al. [KMS07] and Sandor et al. [SCDM10] render the focus (the hidden scenes in the X-Ray vision of AR) and the context (other visible scenes) with different methods, and composite them in the screen to reveal the correct occluding order. Viola et al. [VFSG06] and Krüger et al. [KSW06] composite interior structures and exterior surfaces by dynamically adjusting the transparency in volume rendering.

The gaze simulation in [CDF+06] is the closest related work of our approach. The authors introduce stylized focus to draw the viewer's gaze to the emphasized area through local variations in shading effects and line qualities. Though four focus models are proposed, some easy-use interaction tools are expected but missed unfortunately. And the stylized focus supports single-point based emphasized area only, which loses the compatibility with either eye tracking data or saliency map, and also the ability to fulfill complex stylization tasks.

In this chapter, we present an improved approach that allows the combination of a larger set of styles while allowing a direct user controls of the spatial variation of both styles and combination.

## 4.3 General approach

Our approach is indeed a generalization of image abstraction systems employed by many previous works [WOG06, ZMJ+09, JEDT09] and first described in DeCarlo and Santella's work [DS02]. This generalization is two-folds: 1) we can combine more than two styles that are user-selected; 2) users can modify easily the importance map. The former brings richer visual information and more stylization effects, the latter provides a controllable interface to alter the visual importance.

To introduce our approach, we are using three rendering styles, including original color (original color is considered as a stylization for simplification), color desaturation and line drawing (see the overview of our approach in Figure 4.2). The importance for each style is illustrated by different colors, such as red, green and blue color in Figure 4.2. We may use an image saliency estimation technique [JEDT09] to initialize the importance, to produce results similar to [DS02].

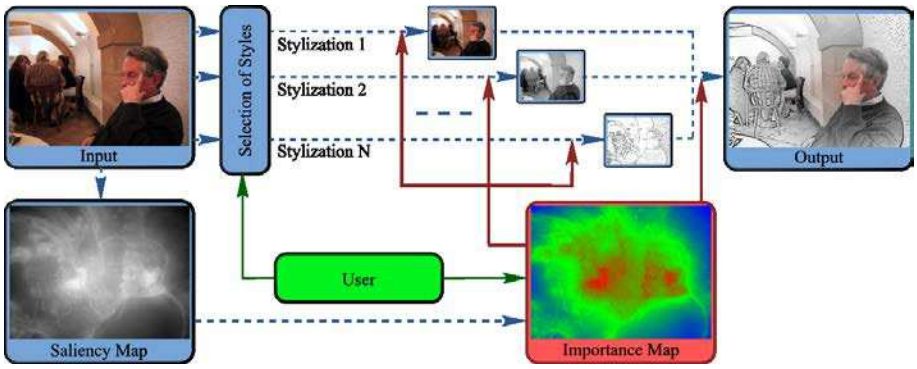



**Figure 4.2:** ***Overview of our system.*** *Three example styles (original color, desaturated color, line drawing) are presented by red, green and blue color respectively in the importance map. One of our main contributions is that we use the importance map to guide not only stylization, but also the synthesis, illustrated by thick red arrows. The user, presented by a green box, has to select the styles, and modify the importance map optionally.*

Or, we can let the user define it using several interaction tools.

Thanks to the importance map $\mathcal{I}$, we are now ready to get the final synthesis result. For each position $\boldsymbol{p} = (x, y)$ in the image, a non-uniform composition of $N$ stylized images $\boldsymbol{S}_i(\boldsymbol{p}, \mathcal{I})$, $i \in [1, ..., N]$, using weights $W_i(\boldsymbol{p}, \mathcal{I})$ is:

$$\boldsymbol{C}(\boldsymbol{p}) = \frac{1}{K} \sum_{i=1}^{N} W_i(\boldsymbol{p}, \mathcal{I}) \boldsymbol{S}_i(\boldsymbol{p}, \mathcal{I}), \tag{4.1}$$

where $K = \sum_{i=1}^{N} W_i(\boldsymbol{p}, \mathcal{I})$ is a normalizing term. We take the importance map as the parameters of both stylization and weighting functions. The introduction of importance improves the quality of synthesis, as presented in the following sections.

## 4.4 Importance map

Our importance map is still based on a gray-scale map with values between 0 and 1. But, in order to composite more than 2 styles, we subdivide the intervals in a sequence $E_i$, with $1 \geq E_0 > E_1 > ... > E_{N-1} \geq 0$. For each boundary of these intervals, a style is assigned. The combination of the gray-scaled image and the boundaries of interval subdivision is our importance map.

One goal of our approaches is to apply different styles in different regions with the guide of an importance map. This is the reason why the blending weight $W_i(\boldsymbol{p}, \mathcal{I})$ depends both on the position $\boldsymbol{p}$ and on the importance map $\mathcal{I}$. As we mentioned in Section 4.2, there are many ways to create an importance map. We divide them by two categories: 1) the main one is based on the core of this thesis, and is inspired by the previous chapter that is, by using extracted structures, such as features and saliency, 2) the other one is drawn by the user with our interaction

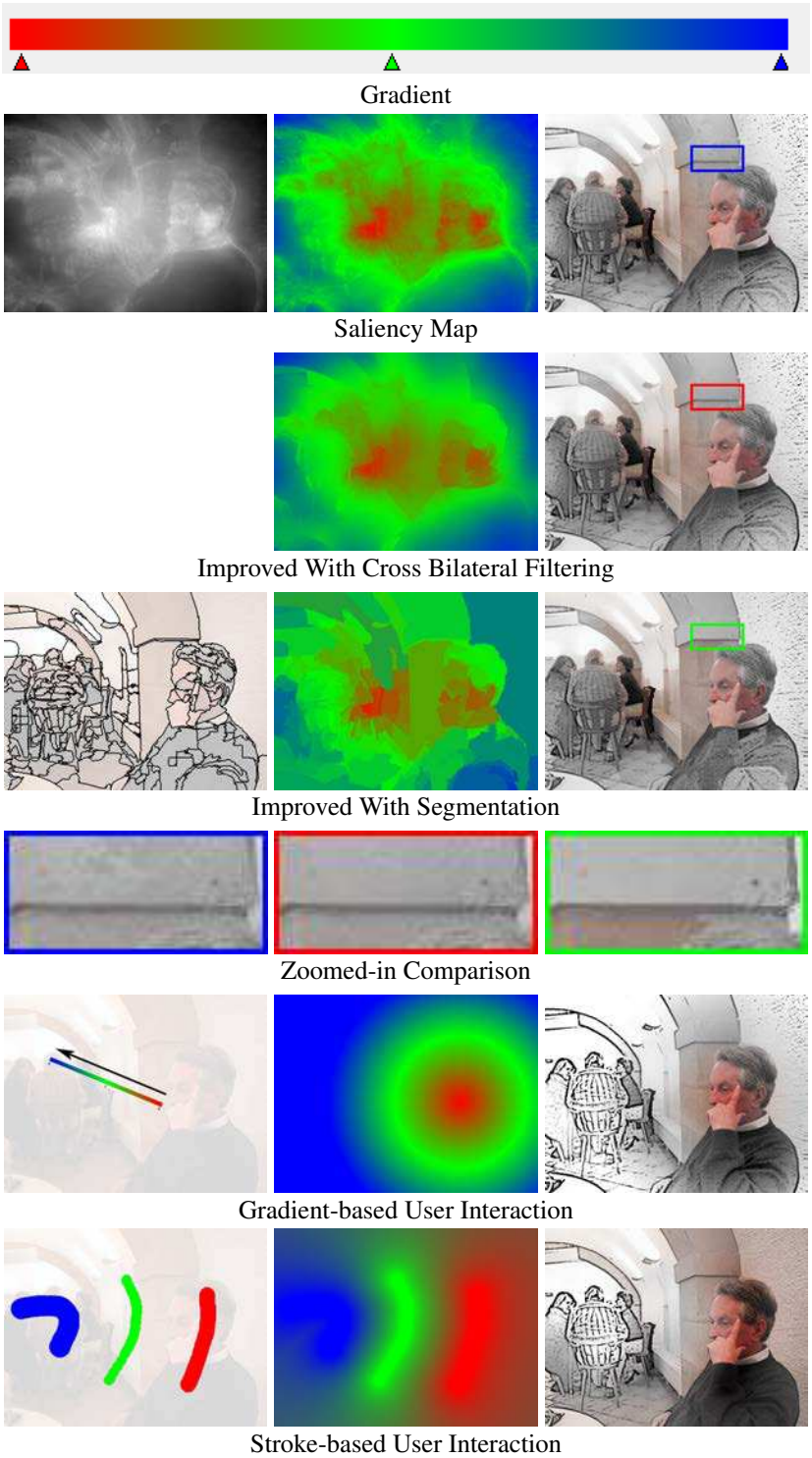



**Figure 4.3:** ***Five importance map examples computed by different user inputs.*** *We take three styles for example, and use red, green and blue color to present these styles, whose importance is 1, 0.5 and 0 respectively. Different user inputs are shown in the left column. In the middle column, we visualize the importance maps using one color for each style and composite them together using the weights described in equation 4.5, the composition results are shown in the right column.*

design. Our approach does not try to improve any of these techniques, but provides the compatibility to utilize any of them easily.

### 4.4.1 Derived from original image

Automatic construction of importance from image relies on image structures. Similar to human visual system, the importance map, describing the possibility of attracting user attention, has to be smooth everywhere. Sharp image features, such as lines or object boundaries, thus are not suitable to define an importance map. While saliency map is usually smooth, because it is computed by a combination of estimation in multiple scales or different levels. For instance, Itti et al [IKN98] employ linear filtering to compute color, intensity and orientation features in multiple scales, and Judd et al [JEDT09] use a machine learning approach to train a classifier directly from human eye tracking data, which considers low-, middle-, high-level features and center prior. In this chapter, we use source codes provided by Judd et al.

This map can be further improved using different image-processing tools. In our approach, we have experimented the following two. The first one, $\mathcal{I}^c$, is obtained by cross bilateral filtering (CBF) the importance map taking the color difference (i.e., gradient structure) in neighborhood pixels of the original image into account:

$$\mathcal{I}^c(\boldsymbol{p}) = \frac{1}{\mathcal{K}^c(\boldsymbol{p})} \sum_{\boldsymbol{q}\in\mathcal{S}} G_{\sigma_s}(|\boldsymbol{p}-\boldsymbol{q}|) G_{\sigma_r}(|\boldsymbol{C}(\boldsymbol{p})-\boldsymbol{C}(\boldsymbol{q})|) \mathcal{I}(\boldsymbol{q}) \tag{4.2}$$

where $\mathcal{K}^c(\boldsymbol{p}) = \sum_{\boldsymbol{q}\in\mathcal{S}} G_{\sigma_s}(|\boldsymbol{p}-\boldsymbol{q}|)\, G_{\sigma_r}(|\boldsymbol{C}(\boldsymbol{p})-\boldsymbol{C}(\boldsymbol{q})|)$ is a normalizing term, $G_\sigma(\boldsymbol{p}) = e^{-\boldsymbol{p}^2/2\sigma^2}$ is Gaussian smoothing, $\mathcal{S}$ is a set of neighboring pixels and $\boldsymbol{C}(\boldsymbol{p})$ presents the color of pixel $\boldsymbol{p}$ in original image. A CBF-based improved importance map and its relative result are shown in the third row of Figure 4.3. Compared with the original result (zoomed-in image in blue frame), $\mathcal{I}^c$ (zoomed-in image in red frame) removes small spots while preserves the strong edges.

The second one, $\mathcal{I}^s$, is defined by averaging importance values in the same segmented region, which is produced by Mean-Shift image segmentation [CM02]:

$$\mathcal{I}^s(\boldsymbol{p}) = \frac{1}{\mathcal{K}^s(\boldsymbol{p})} \sum_{L(\boldsymbol{q})=L(\boldsymbol{p})} \mathcal{I}(\boldsymbol{q}) \tag{4.3}$$

where L is the segmentation label for each pixel, and $\mathcal{K}^s(\boldsymbol{p})$ is the amount of pixels whose label equals to $L(\boldsymbol{p})$. An segmentation-based importance map and its relative result are shown in the forth row of Figure 4.3. Compared with the original result and CBF-based improved one, $\mathcal{I}^s$ (zoomed-in image in green frame) ensures the consistency in each segmented regions.

### 4.4.2 User interaction

Except selecting multiple styles, the user may decide themselves the expected importance level $E_i$ for each of them. Our system applies some styles to the most important regions, and some others to the less important regions. Fortunately, most of the users already have a general idea about this decision when they select the styles from our observation.

To ease this manipulation for users, we provide them with a classical gradient slider to adjust $E_i$ (see the top image in Figure 4.3). Initially, the interval subdivision is uniform. Then the user can adjust the intervals by dragging all the bars.

Thanks to this gradient slider, we now can also introduce a full gradient-based interaction tool, by which user can directly draws a line segment on the image to modify or recreate a new importance map. And we also provide three gradient modes: radial gradient, linear gradient and symmetrical gradient, see a radial gradient example in the sixth row in Figure 4.3.

To provide even more freedom to satisfy complex tasks, we also design a stroke-based interaction tool. After the selection of the styles, the user can directly draw strokes on the image, composed by a series of circles with different radii, to indicate different styles in different regions. The interpolation to get a whole importance map should: 1) smooth everywhere, 2) apply pure selected style for user drawn regions, this is a guarantee to not disobey the user assignation. Radial basis function is one of the well-known smooth interpolation functions, but its computation costs a lot, since it requires to solve a linear equation whose dimension is the amount of all the drawn pixels.

We propose a fast approximate solution based on the distance function:

$$\mathcal{I}(\boldsymbol{p}) = \frac{1}{K'} \sum_{i=0}^{N} E_i \prod_{j=1, j\neq i}^{N} D(\boldsymbol{p}, j) \tag{4.4}$$

where $K' = \sum_{i=0}^{N} \prod_{j=1, j\neq i}^{N} D(\boldsymbol{p}, j)$ is a normalizing term and $D(\boldsymbol{p}, j)$ is the distance between the current position $\boldsymbol{p}$ and the nearest circle related to the *jth* style. This distance to the circle can be easily computed by the distance to the center of circle subtracted by its radius, thus the proposed solution achieves fast performance. This interpolation meets all the required properties, see the bottom row in Figure 4.3.

## 4.5 Non-uniform synthesis

### 4.5.1 Multiple weights

Due to non-uniformity of the synthesis, each style needs its own weighting image, thus N weighting images are needed in total. But, our importance map initialized by saliency map is only a scalar image contains the importance level for each position, normally scaled to $[0, 1]$ and higher value indicates higher importance.

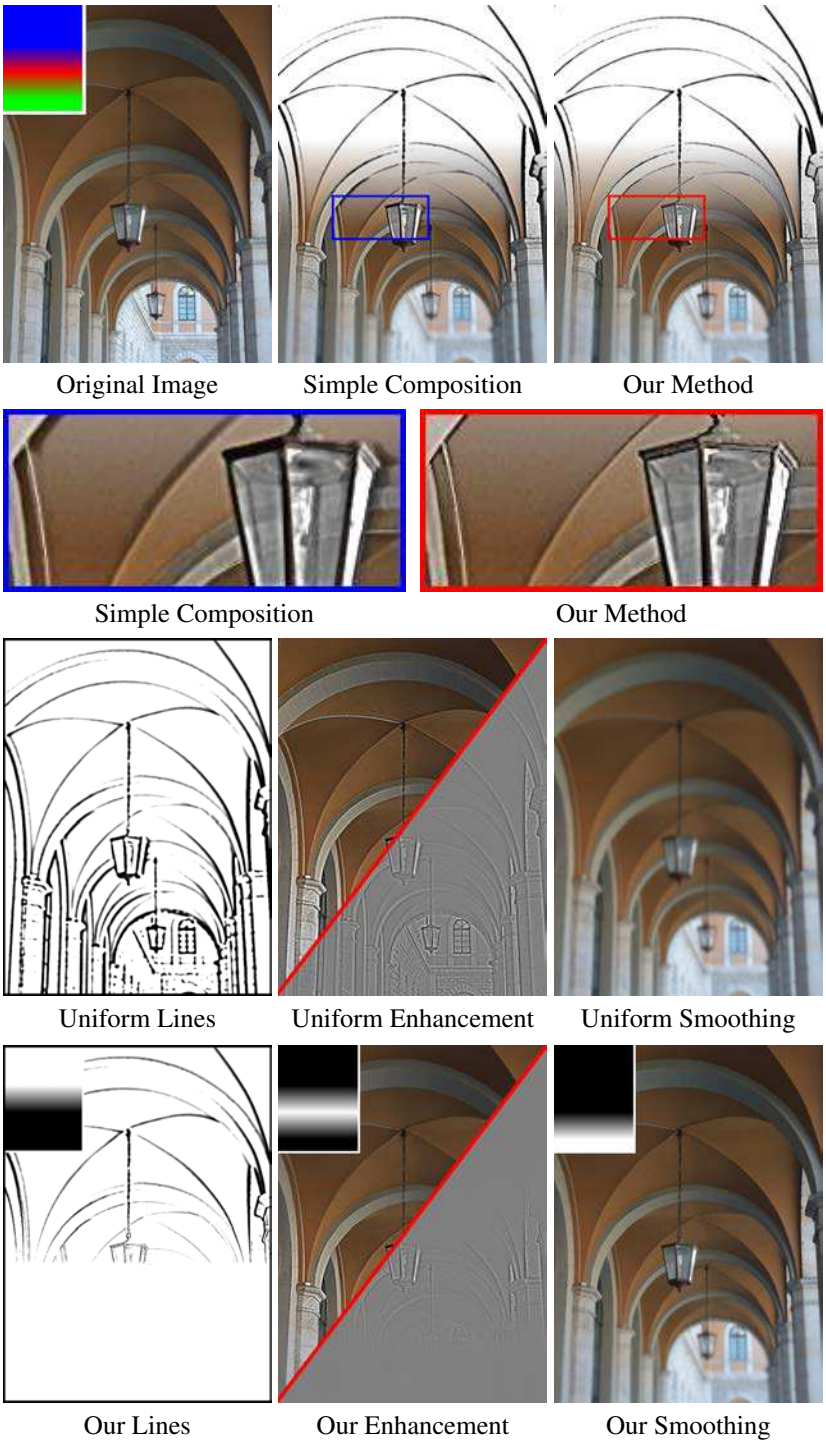



**Figure 4.4:** ***Compare with uniform stylization.*** *In this experiment, we apply line extraction, image enhancement and blurring driven by an importance map. The first row contains the original image (with an importance map), a simple composition result and a synthesis produced by our method. A zoomed-in comparison is shown below. To compare each component of these two compositions, we illustrate all the stylized images one-by-one in the last two rows.*

Therefore, we need to find a mapping function to define N weighting maps from an importance map and N expected importance values $E_i$ the user sets. In our system, we create the weighting maps using a piecewise linear interpolation:

$$W_i(\boldsymbol{p}, \mathcal{I}) = \begin{cases} M(i, i-1), & if\ \mathcal{I}(\boldsymbol{p}) \in [E_i, E_{i-1}] \\ M(i, i+1), & if\ \mathcal{I}(\boldsymbol{p}) \in [E_{i+1}, E_i) \\ 0, & \text{others} \end{cases} \tag{4.5}$$

$$M(i, j) = \frac{D_i(\boldsymbol{p}) \times E_j + D_j(\boldsymbol{p}) \times E_i}{D_i(\boldsymbol{p}) + D_j(\boldsymbol{p})} \tag{4.6}$$

where $D_i(\boldsymbol{p}) = |\mathcal{I}(\boldsymbol{p}) - E_i(\boldsymbol{p})|$ is a distance function between the importance value of current pixel and the expected importance level of the *ith* style. For simplifying the formula, we mark $E_{-1}$ as 1 and $E_N$ as 0. In the gradient tool, the user can see the interpolation result illustrated by different colors, as long as the all of $E_i$ are given. And then the importance map is illustrated in the same way before the stylization, as the middle column of Figure 4.3. Note that highest importance is presented by red, while least importance is presented by blue in this chapter, but the user can select any color for them.

### 4.5.2 Spatial varying stylization

Besides the composition, the importance map is also used to guide the stylization step. A uniform stylization without importance will introduces ghosting artifacts in the final compositing results, see Figure 4.4 for example. The goal of the composition in this example is to draw lines in near-by scene (the top part of the image), blur the distant scene (the bottom part) and enhance the color in the middle as focus. The original image and an gradient importance map is given in the left top image. A simple composition can be obtained by a composition of uniform stylized images. But, the color ghosting artifacts occurs in the regions where more than two styles exist, see the blue zoomed-in image.

Our method adapts the kernel size of filter according to the weights, which are computed by equation 4.5. In the regions with zero weight, the kernel size become zero, thus filter is invalid in these regions. And in the regions with high weights, the kernel size is maximal, a full filter is employed in these regions to achieve a high abstraction. Compared with simple color composition using uniform stylization, our method achieves higher quality. In Figure 4.4, the lines extracted by our method become thinner when the weights are getting smaller, while the lines in simple composition have uniform thickness which leads to ghosting artifacts. The same phenomenon can be observed for image enhancement and blurring.

We have implemented various stylization techniques to demonstrate the effectiveness of our approach:

- **Color desaturation** is an easy method to reduce the color cues. We reduce the two chromatic channels in the CIE Lab color space [Wys82]. Because

it is a pixel-wise conversion, its conversion does not need to be adapted according to the importance as the previous ones.

- **Enhancement** technique increases the sharpness in the region with high contrast. We employ Unsharp-masking [RSI+08] to enhance both images and 3D objects in high important regions.
- **Gaussian smoothing** is an ideal effect for distant scenes.
- **Edge-preserving smoothing** has the ability to smooth the low contrast regions while preserve the sharp edges. Bilateral filter [TM98] is a spatial variable technique which meets the requirement.
- **Line drawing** is the very common but effective method to outline the shape of the objects. We use Difference-of-Gaussian operator [WOG06] to detect lines for images and 3D objects.

All listed rendering techniques, except the color desaturation, are based on local filters. We thus adapt the kernel size of their filters, which use smaller filter kernel where the importance is low. Therefore, original style is rendered with different levels of abstraction according to its correspond weight map. And the combination of different styles thus achieves high compositing quality. The original color is sometimes needed to be preserved in some case, such as the important region in Figure 4.2, we consider it as a special style. We apply these styles directly on the images or each frame of videos. For 3D scenes, we use them either on luminance of their shading or normal map of the 3D meshes. Combination with three styles are employed in Figure 4.2, 4.3, 4.4, more results combined with more styles will be shown in next section.

## 4.6 Results and discussion

Our approach includes parameters $E_i$ (expected importance value for each style) for the composition and the others in each stylization processing, such as the maximal enhancement strength, maximal Gaussian smoothing kernel, spatial kernel and range kernel for bilateral filter, maximal spatial scale, sensitivity and sharpness of DoG edge extraction. The performance of our approach mainly depends on the stylization approaches we employed. Fortunately, our system prefers spatial variable stylization techniques, which are naturally parallel. Therefore, we have implemented our system in GPU architecture, that achieves a real time performance with above styles currently.

Figure 4.6 illustrates various stylized results generated by our system. Figure 4.6 (a) shows an example of architectural 3D models, which are rendered by photo-realistic manner and then stylized with lines, desaturated colors and enhancement from left to right. Figure 4.6 (b) mimics depth-of-field phenomenon approximately, by applying a linear gradient Gaussian blurring. An X-Ray vision

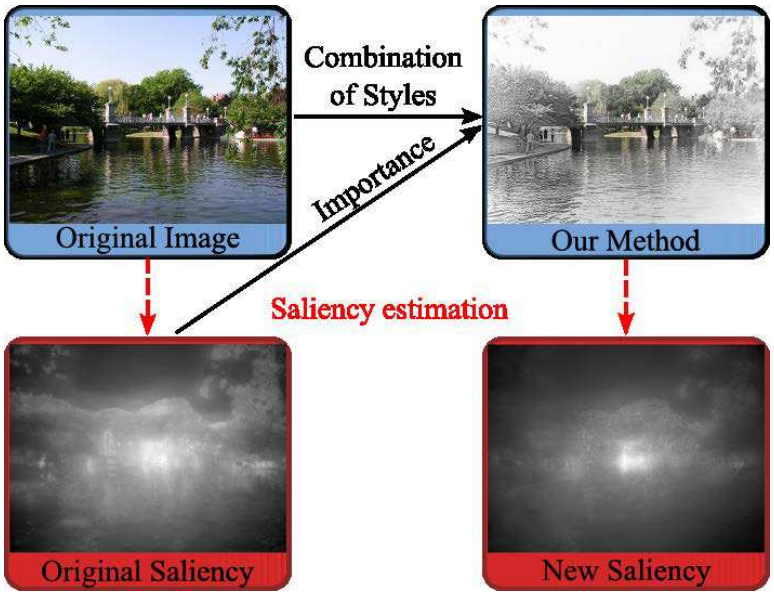



**Figure 4.5:** ***Overview of our evaluation on estimating the visual attention*** *before and after our stylization by the way of saliency Estimation.*

in AR example is given in Figure 4.6 (c), the virtual underground pipelines are rendered using X-toon shading [BTM06], the moving car is segmented out [ZQC+10] and presented in the front of the pipelines as the original color while the background are extracted as line drawing. Figure 4.6 (d) displays three frames from a video experiment, the dolphin is lighted by increasing the luminance, the color of everything on the boat behind the dolphin is desaturated and edge-preserving smoothed while lines are used for sea in the background. A example on large scale image, part of the Chinese ancient painting "Qingming Festival by the Riverside" painted by Song painter Zhang Zeduan, in Figure 4.6 (e), shows the ability of our approach for complex stylization tasks.

Since our importance map is initialized by saliency estimation, the attractive regions should receive higher saliency estimation if the style with a low abstracting level is applied to these regions. The evaluation illustrated in Figure 4.5 demonstrates this suppose. According to the saliency map of the original image, the original color is preserved in high-saliency regions, while desaturated in low-saliency regions. We then estimate the saliency of the processed result. Compared with the original saliency map, the new one gathers the saliency to the high-saliency regions, which shows the potential of our method in the future work to direct the viewer's attention.

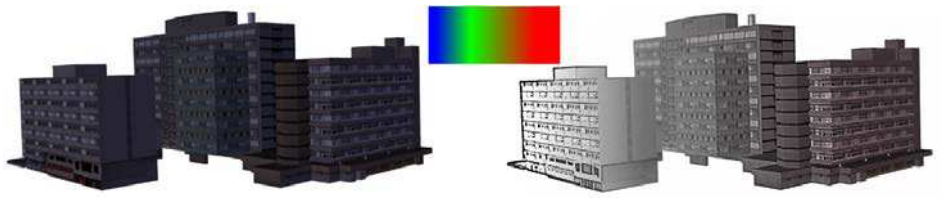
## 4.7 Summary and limitations

In this chapter, we have introduced a non-uniform image synthesis approach

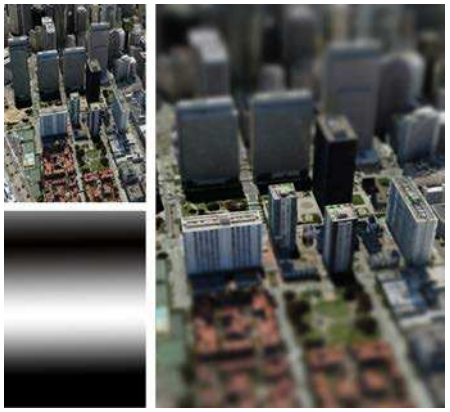
that combines multiple rendering styles driven by an importance map. This map is employed to guide both stylization and composition, which improves the quality and compatibilities with different input data and user interactions.

One limitation of our approach is that our system requires the user to select styles and adjust the expected importance values for each style. Though a default mode and user friendly gradient tools are provided, arbitrarily selection by fresh users may produce unexpected results or even reverse the visual attention. For directing attention, the user is supposed to select less-abstracted style to high-saliency region of the original image, and higher-abstracted style to low-saliency region. An automatic evaluation of the level of abstraction for each style would provide the use a very good suggestion.

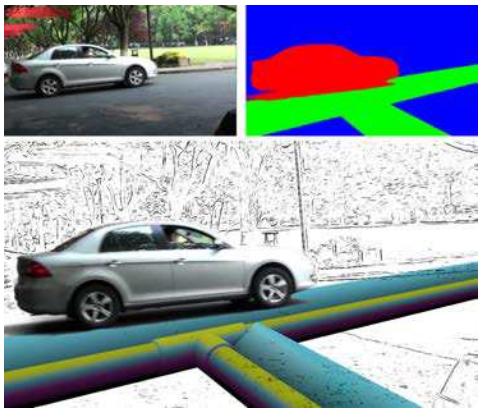
Another research direction deserves future endeavor is to investigate the image structures representing the contained scenes. In importance estimating stage, we have demonstrated that low-level local structure knowledge, such as local color similarities and boundaries of segmented objects, could participate to improve the validity of the importance map, see Figure 4.3. In stylization stage, though our method adapts the kernel size of filters, filtering-based rendering techniques, such as bilateral filter and DoG edge detection, might achieve higher quality stylized results with the help of the image structures. For instance, once again, the local gradient direction has been proved useful to guide the local filter or strokes in order to have a better spatial coherent result [KD08, KLC09]. We believe that current image analysis methods are either lacking of accuracy or too specific for particular applications. This justifies the development of a higher-order gradient field estimation with an exploitation of improvements for various applications in the next chapter of this thesis.



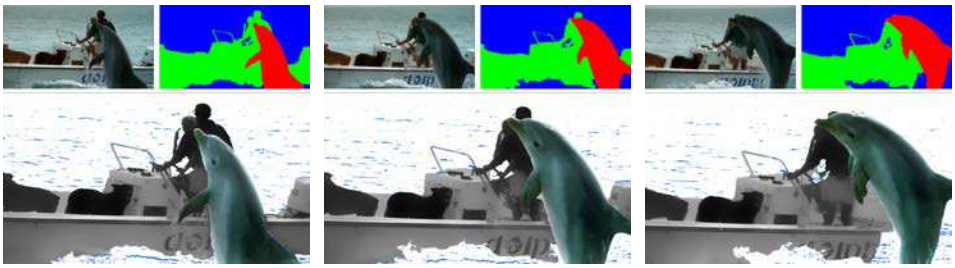
(a) Synthesis with four styles



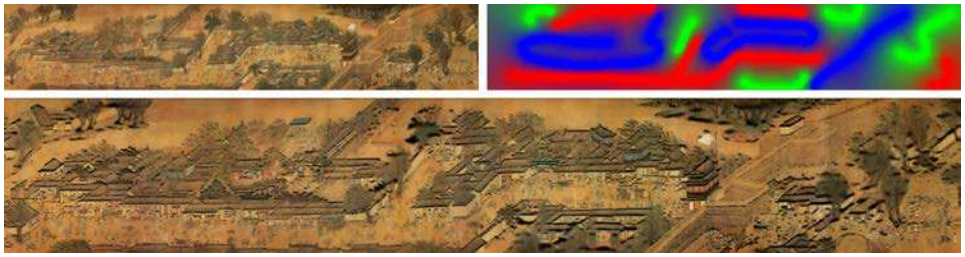
(b) Appr. depth-of-field



(c) X-Ray vision in Augmented Reality



(d) Three frames from our video example



(e) Complex stroke-based interaction

**Figure 4.6:** ***Various results.*** *Each result is shown with its scaled-down original image and the importance map. An example of architectural 3D models includes a photo-realistically rendered image, gradual and final result (a), a depth-of-field effect is approximated using symmetrical gradient importance (b), and (c) exhibits an example for X-Ray vision in AR. (d) shows three resulting frames in our video experiments, their importance maps produced by video segmentation are also given. We have also experimented complex stroke-based user interaction on large scale image (e): trees are all outlined with thick lines, and buildings are silhouetted by thin lines, and the edge-preserving smoothing is applied on the pedestrians.*

# Part II

# On improving structure analysis for image stylization

CHAPTER 5

# Non-oriented MLS gradient field

In the previous part, we have shown that the use of image structures, and particularly the gradient field, plays a key role in many applications, such as X-Ray vision in Augmented Reality, image synthesis by combining multiple rendering styles, and even directing user attentions. But, as pointed in the previous limitation sections, they are lack of either accuracy or spatial and temporal coherency. The same limitation exists for many image stylization methods guided by a gradient field [Win11, KLC09, KKD09], including the line extraction and rendering we employed [KLC07] in the previous part. In this chapter, we introduce a new approach for defining continuous gradient fields, a fundamental stage for a variety of computer graphics applications.

## 5.1 Introduction

The use of gradient information for image processing has been proven to increase accuracy. For image and video stylization, the use of a gradient field permits to orient filters along image features for a variety of applications: Kang et al. apply Difference-of-Gaussian filter along gradient fields to extract continuous line drawing [KLC07], and bilateral filter along gradient and tangent direction separately to abstract color [KLC09], many painterly rendering techniques orient strokes according to the gradient direction [Her98, HE04, PP09] to keep object boundaries clear. All of them result in stylized images that better preserve the original features.

As pointed out in Section 2.4, a large enough smoothing size is needed if there are a lot of noise or flat regions to be filled in; but computing or smoothing image gradient in a large neighborhood raises oversmoothing and pair cancellation issues. Pair cancellation issue can be well reduced by *structure tensor* [BG87]. Unfortunately, a structure tensor can only encode a locally *constant* gradient field, thus still tends to oversmooth original image structures. In the neighborhood of each pixel, only a principal direction is estimated as the eigenvector corresponding to the maximum eigenvalue in the structure tensor. This constitutes a major limitation when used to average information over curved neighborhoods. In this case, it quickly

fails to preserve the original structure and tends to smooth out important image details (Fig 2.11-(f)). This problem is amplified with increasing neighborhood sizes or sparsity. Medium-to-large filter sizes are not only required to regularize noise found in images, but also for simplification or abstraction purposes, especially for high dynamic range images [CPD07]. We also emphasize that in image processing, sparsity arises when resampling low resolution information, and in places where the input gradient has a close-to-zero magnitude, as shown in Figure 2.10.

It is interesting to note that the same issue raises in surface reconstruction applications. Normals (i.e., normalized gradient of a surface) have been widely used to regularize ill-posed optimization problems [HDD+92, SOS04, KBH06], or to improve the robustness of local implicit fitting techniques [AK04, AA04b, GG07], but consistently orienting normals from raw point clouds have been recognized considerably harder, as difficult as reconstructing the surface itself [HLZ+09, MDGD+10]. The limitation of structure tensor, a low order approximation, has already been pointed out in the previous *Algebraic Point Set Surfaces* (APSS) framework [GG07].

In this chapter, we introduce a new local approximation method for discrete *non-oriented* gradient fields that better preserves image structures. The key ingredient of our approach is an extension of the structure tensor to a *higher-order* basis. In particular, we show that an isotropic linear basis provides the best trade-off between accuracy and smoothness. Using a MLS formulation, we define a *continuous* non-oriented gradient field at arbitrary scale. Our main propose of this chapter is image features. However, similarly, we will also illustrate our approach on 2D surface reconstruction, as well as on 1D curve reconstruction. Surface and curve approximation from a raw point cloud is achieved by a MLS integration of the local gradient fields, leading to a continuous implicit reconstruction. Our reconstructed gradient fields and implicit curves of surface are analytically differentiable. Last but not least, our approach does not require any preprocessing and involves only local computations.

## 5.2 Non-oriented MLS gradient field

In this section we present our novel non-oriented gradient field and the subsequent manifold approximation technique which builds on the moving least squares (MLS) formalism. We first describe the general approach (Sec 5.2.1), then discuss the choices of appropriate regularization (Sec 5.2.2) and higher-order basis (Sec 5.2.3), and finally give some differential properties of our approach (Sec 5.2.4).

Before introducing our approach, we briefly reintroduce the formulism of MLS method. The MLS method starts with a weighted least squares formulation, and then moves point $\boldsymbol{x}$ over the entire parameter domain $\prod_m^d$, where a weighted least squares fitting is computed and evaluated for each point individually$f(\boldsymbol{x})$ [Nea04]. $\prod_m^d$ is the space of polynomials of total degree $m$ in $d$ spatial dimensions. It can be shown that the global function $f(\boldsymbol{x})$ at point $\boldsymbol{x}$ is defined as the value of the

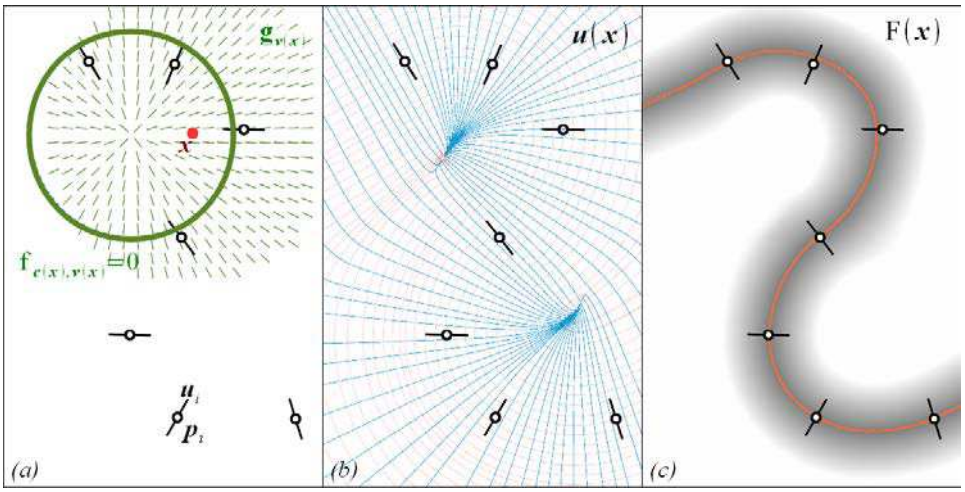



**Figure 5.1:** ***Overview of our MLS approach on sparse samples.*** *(a) In green, the local approximation* $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ *of the input vectors* $\boldsymbol{u}_i$ *around the evaluation point* $\boldsymbol{x}$*. (b) Streamlines of the global reconstruction of the continuous non-oriented vector field* $\boldsymbol{u}$ *(in blue), and of its respective 2D tangent field (in red). (c) The reconstructed continuous and unsigned scalar potential F, and its* 0*-isovalue. Note that this iso-contour* ***does not*** *correspond to the tangent field streamlines. This figure has been made using the isotropic linear basis.*

approximating function at this point $\mathrm{f}_{\boldsymbol{x}}(x)$:

$$\mathrm{f}(\boldsymbol{x}) = \mathrm{f}_{\boldsymbol{x}}(\boldsymbol{x}), \quad \underset{\mathrm{f}_{\boldsymbol{x}} \in \prod_m^d}{\arg\min} \sum_i w_i(|\boldsymbol{x} - \boldsymbol{x}_i|)\ |\mathrm{f}_{\boldsymbol{x}}(\boldsymbol{x}_i) - \mathrm{f}_i|^2 \tag{5.1}$$

where $\mathrm{f}_i$ is the given scalar values in the neighborhood of point $\boldsymbol{x}$, and $w_i$ is the weights depending on a spatial distance between central point $\boldsymbol{x}$ and its neighbor points $\boldsymbol{x}_i$.

### 5.2.1 General approach

Our algorithm takes as input a set of non-oriented and unit vectors $\boldsymbol{u}_i \in \mathbb{R}^d$ specified at sample positions $\boldsymbol{p}_i \in \mathbb{R}^d$, with $d$ the dimension of the ambient space (see Figure 5.1-(a) for an example in $\mathbb{R}^2$). The samples should be pixels on the image grid, here we use sparse samples for a sake of legibility. In this work, we further assume the directions $\boldsymbol{u}_i$ come from the normalized gradient of an unknown continuous scalar potential. Their orientation (i.e., signs) are either unknown or irrelevant as previously motivated in the introduction.

Given an arbitrary evaluation point $\boldsymbol{x} \in \mathbb{R}^d$, we assume the neighborhood gradients $\boldsymbol{u}_i$ may be well approximated by a low degree polynomial gradient field $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$

defined in matrix form as (Figure 5.1-(a)):

$$\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}(\mathbf{y}) = \mathbf{B}(\mathbf{y})^T \boldsymbol{v}(\boldsymbol{x}) \,, \tag{5.2}$$

where $\mathbf{B}$ is a polynomial basis matrix with $d$ columns, and $\boldsymbol{v}$ is the vector of unknown coefficients. For local gradient approximation, $\mathbf{B}$ must represent an integrable basis, and its particular choice will be discussed in section 5.2.3.

The key idea to determine the coefficients of $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ is to consider the sum of the squared dot products between the local gradient field and the prescribed input vectors $\boldsymbol{u}_i$, which is maximized when they are aligned, independently of their orientation:

$$\boldsymbol{v}(\boldsymbol{x}) = \arg\max_{\boldsymbol{v}} \sum_i w_i(\boldsymbol{x}) \left( \boldsymbol{u}_i^T (\mathbf{B}(\boldsymbol{p}_i - \boldsymbol{x})^T \boldsymbol{v}) \right)^2 \,. \tag{5.3}$$

Here, $w_i$ is a smooth weight function, decreasing with respect to the distance $\|\boldsymbol{p}_i - \boldsymbol{x}\|$, that plays the role of low-pass filter. In this chapter, we take the suggestion of Guennebaud and Gross [GG07], $w_i = \phi(\left|\boldsymbol{p}_i - \boldsymbol{x}\right| / S)$, where $S$ is the support size, and $\phi$ is a quartic kernel function:

$$\phi(x) = \begin{cases} (1 - x^2)^2 & \text{if } x < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

In matrix form, Eq (5.3) becomes:

$$\boldsymbol{v}(\boldsymbol{x}) = \arg\max_{\boldsymbol{v}} \boldsymbol{v}^T \mathbf{A}(\boldsymbol{x}) \boldsymbol{v} \,, \tag{5.5}$$

with the covariance matrix $\mathbf{A}$ defined as:

$$\mathbf{A}(\boldsymbol{x}) = \sum_i w_i(\boldsymbol{x}) \boldsymbol{u}_i^T \mathbf{B}(\boldsymbol{p}_i - \boldsymbol{x})^T \mathbf{B}(\boldsymbol{p}_i - \boldsymbol{x}) \boldsymbol{u}_i \,. \tag{5.6}$$

From these local approximations, we reconstruct a continuous, but non-oriented, vector field $\mathbf{u} : \mathbb{R}^d \to \mathbb{R}^d$ that globally approximates the discrete vectors $\boldsymbol{u}_i$ (Figure 5.1-(b)). Since we are using a centered basis, $\mathbf{u}$ is classically defined as $\mathbf{u}(\boldsymbol{x}) = \mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}(0)$. This MLS reconstruction of the gradient field can be used directly for image abstraction and stylization purposes.

Omitting the dependencies on $\boldsymbol{x}$ for brevity, when the input samples are supposed to lie on an unknown manifold, each local gradient approximation $\mathbf{g}_{\boldsymbol{v}}$ can be integrated to recover a local scalar potential $\mathrm{f}_{c,\boldsymbol{v}}(\mathbf{y}) = c + \mathrm{h}_{\boldsymbol{v}}(\mathbf{y})$ such that $\nabla \mathrm{f}_{c,\boldsymbol{v}} = \mathbf{g}_{\boldsymbol{v}}$. The scalar potential $\mathrm{h}_{\boldsymbol{v}}$ is directly obtained by integrating $\mathbf{g}_{\boldsymbol{v}}$, while the constant offset $c$ is recovered such that the 0-isosurface of $\mathrm{f}_{c,\boldsymbol{v}}$ best approximates the input sample positions $\boldsymbol{p}_i$ nearby $\boldsymbol{x}$ (Figure 5.1-(a)). To this end, we minimize the potentials $\mathrm{f}_{c,\boldsymbol{v}}(\boldsymbol{p}_i)$ in a moving least square sense:

$$c = -\frac{\sum_i w_i(\boldsymbol{x}) \mathrm{h}_{\boldsymbol{v}}(\boldsymbol{p}_i - \boldsymbol{x})}{\sum_i w_i(\boldsymbol{x})} \,. \tag{5.7}$$

As illustrated in Figure 5.1-(c), a continuous, but unsigned, scalar potential $F : \mathbb{R}^d \rightarrow \mathbb{R}$ whose 0-isosurface globally approximates the non-oriented input samples is finally defined as $F(\boldsymbol{x}) = f_{c(\boldsymbol{x}),\boldsymbol{v}(\boldsymbol{x})}(0)$.

Two questions remain open though. Indeed, the maximization of Eq (5.5) is an under-constrained problem which has to be regularized in order to avoid the trivial solution of a vector field with infinite magnitude. This question will be addressed in the next sub-section before answering the second question of the critical choice of a higher order polynomial basis for the local gradient fields.

### 5.2.2 Problem regularization

Before presenting our regularization constraint for a higher-order basis, we briefly study the case of a constant local approximation.

#### Regularization for constant approximations

The simplest approach is to assume that $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ is a constant multi-valued function, i.e., $\mathbf{B}$ is the identity matrix and $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}(\boldsymbol{y}) = \boldsymbol{v}(\boldsymbol{x})$. Since we assume unit input vectors, a natural regularization is then to constrain $\boldsymbol{v}(\boldsymbol{x})$ to be a unit vector. Under such a constraint, the solution $\boldsymbol{v}(\boldsymbol{x})$ of Eq (5.5) is directly found as the eigenvector $\boldsymbol{v}$ of the maximal eigenvalue $\lambda$ of the following eigenvalue problem:

$$\mathbf{A}(\boldsymbol{x})\boldsymbol{v} = \lambda \boldsymbol{v} \; . \tag{5.8}$$

This formulation corresponds to a continuous definition of the *structure tensor* defined from discrete and possibly scattered inputs. In the context of MLS surface reconstruction, it has already been suggested by Amenta and Kil [AK04], though to our knowledge no result has been shown yet.

#### Generic regularization

In order to overcome the limitations of constant approximations, we study the possibility to locally approximate the vector field by higher order polynomials. In this case, the previous regularization term ($\left|\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}\right|^2 = 1$) does not apply: if $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ is a multivalued polynomial of arbitrary degree, then its magnitude is not constant anymore. Nevertheless, one could be tempted to enforce this in a least squares sense by minimizing $\sum_i w_i(\boldsymbol{x})(\left|\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}(\boldsymbol{p}_i - \boldsymbol{x})\right| - 1)^2$. To our knowledge, there is no direct method to solve for Eq (5.5) with such additional least squares constraints. Moreover, this strategy would exhibit the undesirable effect to favor local solutions close to a constant gradient field because this is the only solution that can fully satisfy such a constraint in general. Naively constraining the Euclidean norm of the solution vector, i.e., $|\boldsymbol{v}(\boldsymbol{x})| = 1$, is not an option either since, for a general basis, such a normalization is clearly not invariant to similarity transformations. It would therefore introduce a huge bias in the optimization process [Pra87].

Our key observation here is that, **in average**, the norm of $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ should be equal to 1 nearby the considered samples. Formally, this can be expressed as a quadratic normalization constraint:

$$\boldsymbol{v}(\boldsymbol{x})^T \mathbf{Q} \boldsymbol{v}(\boldsymbol{x}) = 1 \,, \tag{5.9}$$

with $\mathbf{Q}$ a symmetric positive definite matrix:

$$\mathbf{Q}(\boldsymbol{x}) = \frac{\sum_i w_i(\boldsymbol{x}) \mathbf{B}(\boldsymbol{p}_i - \boldsymbol{x}) \mathbf{B}(\boldsymbol{p}_i - \boldsymbol{x})^T}{\sum_i w_i(\boldsymbol{x})} \,. \tag{5.10}$$

Equation 5.9 reduces the space of solution to the set of unitary vectors with respect to our *data-dependent* quadratic norm defined by the matrix $\mathbf{Q}$. Note that the choice of the target magnitude, here 1, is totally arbitrary and does not affect the directions of the fitted gradient. Strictly speaking, our norm $\mathbf{Q}$ is obviously not affine invariant. However, by construction it is both invariant to the choice of a basis frame, and to similarity transformations of the input data. The solution $\boldsymbol{v}(\boldsymbol{x})$ of Eq (5.5) is now directly found as the eigenvector $\boldsymbol{v}$ of the maximal eigenvalue $\lambda$ of the following *generalized* eigenvalue problem:

$$\mathbf{A}(\boldsymbol{x}) \boldsymbol{v} = \lambda \mathbf{Q}(\boldsymbol{x}) \boldsymbol{v} \,. \tag{5.11}$$

When $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ represents a constant vector field, then $\mathbf{Q}$ is the identity matrix, and our generalized approach gently boils down to the previous structure tensor. In the next section, we discuss the choice of the local approximation $\mathbf{g}_{\boldsymbol{x}}$.

### 5.2.3 Choice of the basis

The choice of basis for $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ is critical since it characterizes the quality of the approximation. In particular, we will consider constant $\mathbf{g}^0$, linear isotropic $\mathbf{g}^*$, and general linear $\mathbf{g}^1$ gradient fields whose expression and respective scalar potential are given below:

$$\begin{array}{rclcrcl} \mathbf{g}^0(\mathbf{y}) & = & \boldsymbol{r} & \rightarrow & \mathrm{f}^0(\mathbf{y}) & = & c + \boldsymbol{r}^T \mathbf{y} \\ \mathbf{g}^*(\mathbf{y}) & = & \boldsymbol{r} + l\mathbf{y} & \rightarrow & \mathrm{f}^*(\mathbf{y}) & = & c + \boldsymbol{r}^T \mathbf{y} + \frac{1}{2} l \mathbf{y}^T \mathbf{y} \\ \mathbf{g}^1(\mathbf{y}) & = & \boldsymbol{r} + \mathbf{L} \cdot \mathbf{y} & \rightarrow & \mathrm{f}^1(\mathbf{y}) & = & c + \boldsymbol{r}^T \mathbf{y} + \frac{1}{2} \mathbf{y}^T \mathbf{L} \mathbf{y} \end{array}$$

where $c$, $r$, $\frac{1}{2} l(\mathbf{L})$ are scalar potential coefficients of constant, linear and quadric term respectively, and $\mathbf{L}$ is a symmetric matrix to ensure that $\mathbf{g}^1$ is a gradient field. Once integrated, they represent a hyper-plane, a hyper-sphere, and a general quadric respectively. $\mathbf{g}^*$ and $\mathrm{f}^*$ are illustrated in Figure 5.1, compared to others in Figure 5.2 for sparse data, and in Figure 5.4 for an image. As already noticed in the introduction, the constant basis cannot approximate well highly-curved regions, yielding to over-smoothing of the features and even instabilities (Figure 5.2-(c), 5.4-(b)). On the other hand, the general linear basis of $\mathbf{g}^1$ already presents too many degrees of freedom (d-o-f) (5 in 2D, and 9 in 3D), and leads to over-fitting

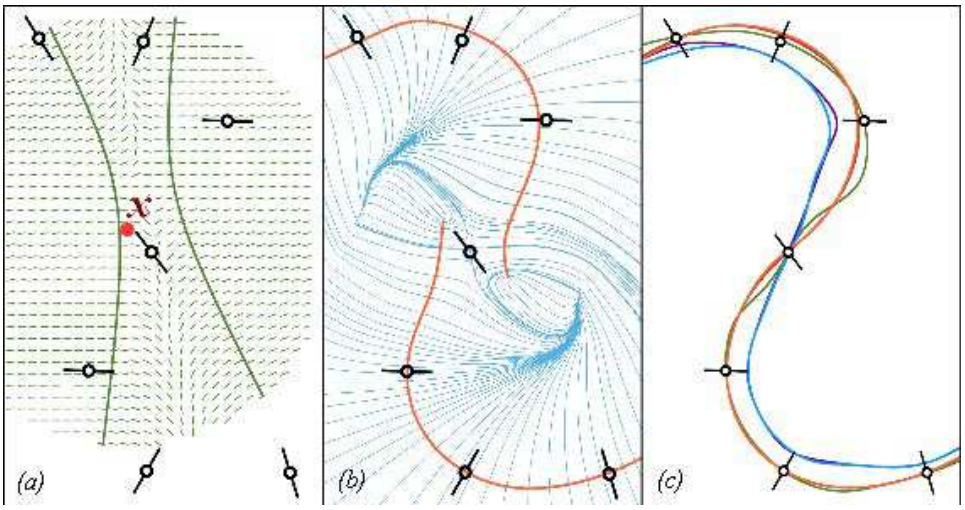



**Figure 5.2:** ***Illustration of the overfitting issue*** *of the fully integrable linear basis* $(\mathbf{g}^1, f^1)$*, with (a) the local gradient field and best fitted isoline for the given red point, and (b) the global MLS reconstruction of the gradient field and isocurve. (c) Comparison of various MLS variants: planar fit without normals (magenta) and with non-oriented normals (**our** $f^0$, blue), spherical fit without normals (green) and with non-oriented normals (**our** $f^*$, orange), and, for comparison purpose, spherical fit with consistently oriented normals (red). Note that these last two curves match almost exactly.*

issues in the form of the generation of oscillations and details which are not present in the input data (Figure 5.2-(a), 5.2-(b), 5.4-(c)).

The isotropic basis $\mathbf{g}^*$ clearly appears to be the best trade-off. As already noticed by Guennebaud and Gross [GG07], compared to a constant basis it only increases by one the number of d-o-f while offering a much richer space of solutions. Keeping the number of d-o-f as low as possible is highly desirable, not only for performance reasons, but also to keep high stability with small neighborhood. Enlarging the neighborhood increases both the computation costs and the risk to take into account different and/or inconsistent parts of the input data. Moreover, it should be noted that in the case of manifold reconstruction, our approach works best if the norm of the gradient along a given isovalue of the respective local scalar potential is constant. Since this is not the case of the full linear gradient, this basis also suffers from a slight bias in the fitting process.

Figure 5.2-(c) shows that the non-oriented normal information has a greater impact on spherical MLS than on planar MLS. Our intuition is that the non-oriented normal information is required to avoid the fitted spheres to *sink* in between the input samples, thus producing a high quality smooth surface that well preserves the features. Finally, it should be noted that in this example, our MLS reconstruction

with non-oriented normals works very closely to the APSS method which can fit spheres with oriented normals only.

### 5.2.4 Differential quantities

An important property of MLS is that the continuity of the reconstruction depends on the continuity of the weight functions $w_i$. Assuming twice differentiable weight functions $w_i$, our global gradient and scalar fields $\mathbf{u}$ and F can then be analytically differentiated to compute, for instance, exact surface normals and/or surface or flow curvature. In practice, differentiating $\mathbf{g}_{\boldsymbol{v}(\boldsymbol{x})}$ or $\mathrm{f}_{\boldsymbol{v}(\boldsymbol{x})}$ with respect to $\boldsymbol{x}$ involves the differentiation of a generalized eigenvalue problem which turns out to lead to solving an additional linear singular problem [Mag85].

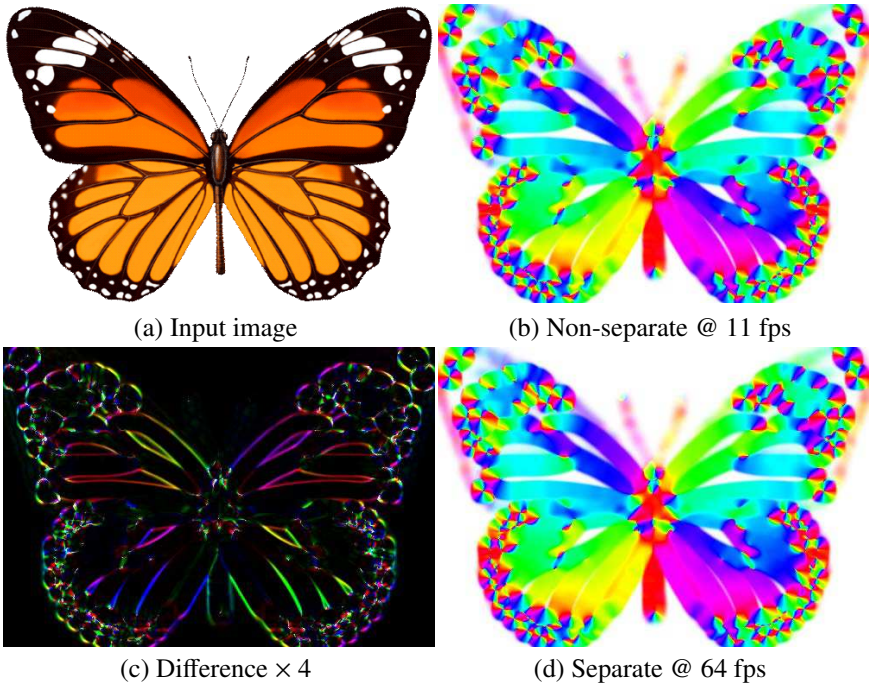
When speed matters, our local approximations may be used to approximate these differential properties. For instance, the gradient field $\mathbf{u}$ may be considered as an approximation of the normal field of the reconstructed surfaces. A higher-order isotropic linear basis allows to go a step further to approximate the surface mean curvature $\kappa_m$ as $\kappa_m \approx l/\sqrt{\boldsymbol{r}^2 - 2c\,l}$. In the same vein, for a 2D gradient field, the tangential curvature $\kappa_t$ of the respective tangential field can be approximated as $\kappa_t \approx 2l$.

## 5.3 Results and discussion

**Implementation details** In practice, we employed a $5 \times 5$ Gaussian derivative filter to compute the image gradient $\nabla I$ from which we get the unit vectors $\boldsymbol{u}_i$. In order to favor gradients with large magnitude while taking advantage of the regular image structures via separable weights, the weight functions $w_i$ of position $\boldsymbol{x} = (x, y)$ are slightly modified as:

$$w_i(x,y) = \left(\phi\left(\frac{|x - x_i|}{S}\right)\phi\left(\frac{|y - y_i|}{S}\right)\right)^{\frac{1}{2}} |\nabla I(x_i, y_i)|^2 \ . \tag{5.12}$$

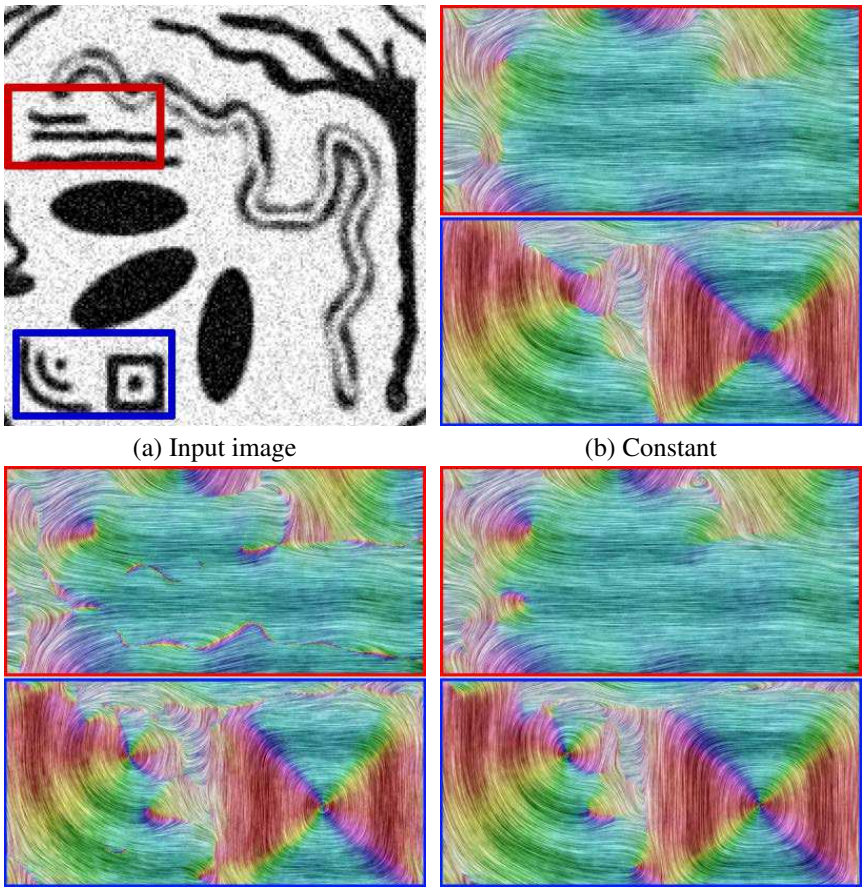
Figure 5.3 exhibits that only a minor difference occurs on the boundaries of the constructed gradient field. Multichannel images yield multiple gradient vectors per pixels which are all taken into account and approximated during the construction of the covariance matrix $\mathbf{A}$ (Eq (5.6)). We unfold each term of this matrix, compute all 1D sums related to x coordinate but irrespective of y coordinate in the first pass, and then store these sums into 3 buffer textures, finally construct matrix $\mathbf{A}$ by computing 1D sums in y coordinate of values in these textures. This separate computing strategy reduces the number of texture sampling and averaging from $O(N^2)$ to $O(2N)$, where $N = (2S + 1)^2$. It significantly speeds up the performance when a large neighborhood size is used. Our image processing system is entirely implemented on the GPU, and we obtained the results using an NVIDIA GeForce GTX 460 with 1GB of memory.



(a) Input image (b) Non-separate @ 11 fps

(c) Difference × 4 (d) Separate @ 64 fps

**Figure 5.3:** ***Difference between non-separate and separate computation*** *on a* $500 \times 347$ *image (a), using a* $23 \times 23$ *neighborhood size, is subtle. (c) shows that the difference on the features are nearly none. The major difference is observed on the boundaries of the reconstructed flow, due to different kernel shapes of non-zero weights (square for separate, round for nonseparate). Note that the performance superiority will be more distinctive if larger neighborhood size is employed.*



(a) Input image (b) Constant

(c) Full linear (d) Isotropic linear

**Figure 5.4:** ***Study of different approximations:*** *Noisy images (*$400 \times 400$*), such as in (a), require a large smoothing neighborhood (*$33 \times 33$ *here). The gradient field is visualized using line-integral convolution, and a HSV color code. The constant approximation (b) tends to oversmooth (cf. line extremities and dot centers). The full linear approximation (c) leads to over-fitted issues. Our isotropic linear solution provides the best trade-off (d).*

**Gradient field results** Thanks to the separability and the GPU implementation, our system easily achieves real-time performance. For instance, on a $512 \times 512$ image, our non-optimized implementation of the ILA costs 5.1 ms (resp. 11.5 ms) for a $11 \times 11$ (resp. $41 \times 41$) neighborhood. Compared to structure tensor, we observe a slowdown factor of about 1.5 to 2 because of the need to solve a slightly larger eigenvalue problem (3×3 instead of 2×2). However, our ILA implementation currently relies on a generic eigensolver, while we use a direct one for CA. This small overhead could probably be significantly reduced using a more GPU friendly direct $3 \times 3$ eigensolver.

Figure 5.4 compares gradient field estimated by constant, full linear and isotropic

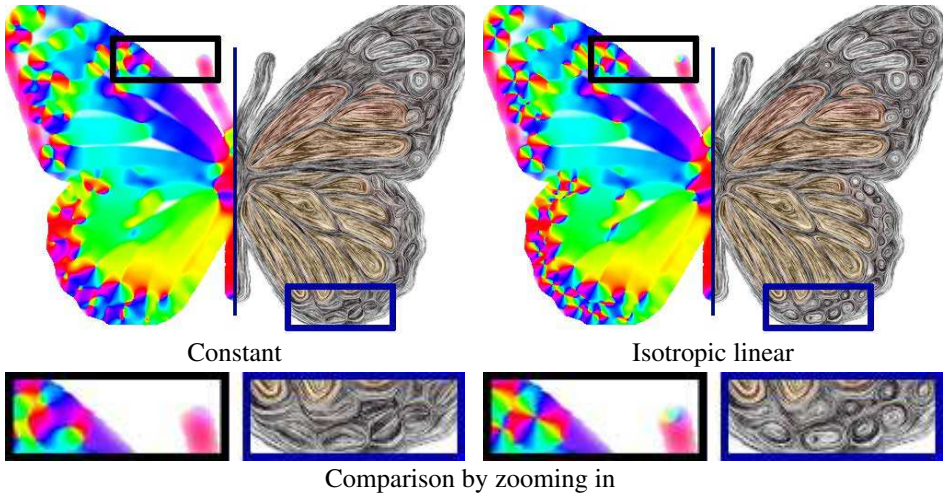
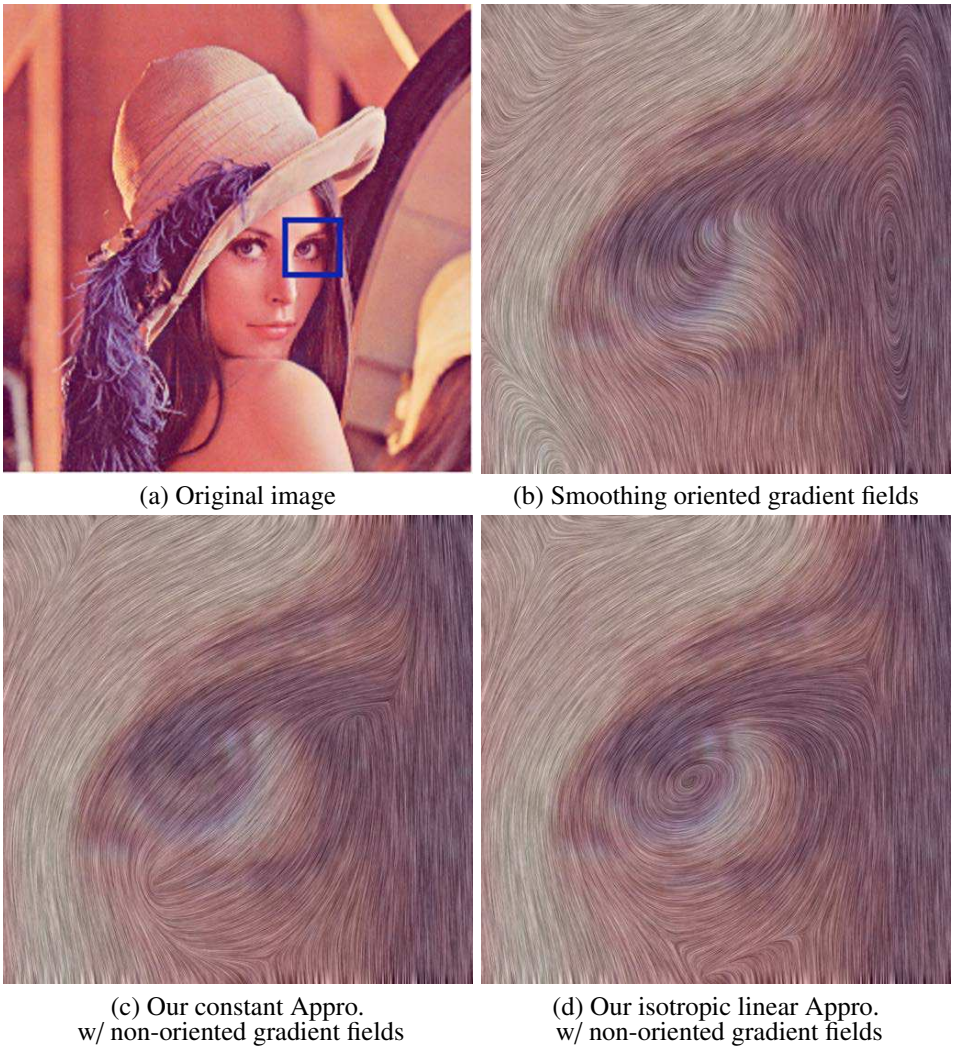



**Figure 5.5:** ***Comparison with constant approximation.*** *To fill in gradient holes of Figure 5.7-(a) (*$500 \times 347$*), a* $23 \times 23$ *neighborhood size is used. Comparison with either color visualization or LIC (line integral convolution). Note the general improvement in shape preservation using our isotropic linear approximation, such as line extremities on the antennas and dot stripes on the wings in the zoomed-in comparison.*

linear approximations on a well-known noise image. The constant approximation tends to oversmooth quickly. For instance, it extends lines on their extremities, because the line direction dominates this approximation. With the same limitation, the constant approximation also fails to preserve the accuracy of the center of the dot. The full linear approximation overcome the above shortcomings, but leads to over-fitting artifacts, i.e. the ones between slightly waved lines in red box of Figure 5.4-(c). Our isotropic linear solution provides the best-off. Figure 5.5 illustrates the same conclusion. Note that though constant approximation in Figure 5.5-(a) is similar to structure tensor result (Figure 2.11-(c)) in theory, it is still better to fill in gradient holes due to our new weighting strategy. Figure 5.6 shows pair cancellation issues for directly smoothing oriented gradient fields in (b), and shape losing problem of constant approximation in (c).

**Surface and curve reconstruction results** As we have introduced in Section 5.2.1, our approach can also be used to reconstruct scalar potential and thanks to the MLS compatibility for different dimension, we are able to demonstrate scalar potential reconstruction in 2D curve reconstruction and 3D surface reconstruction.

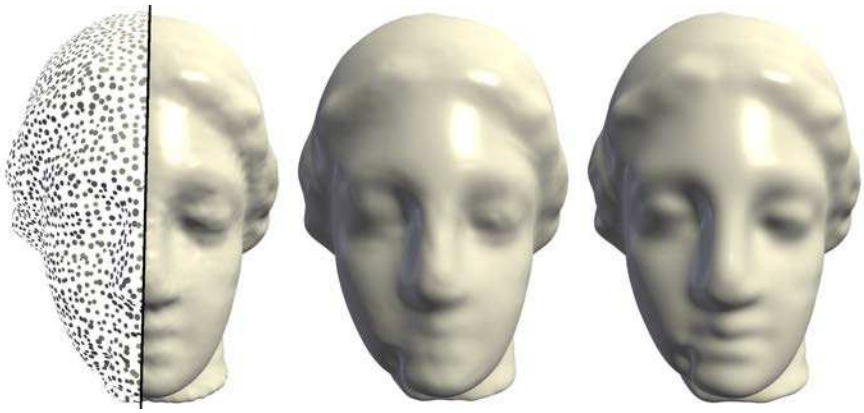
Our novel surface reconstruction approach essentially extends the APSS [GG07] method to support non-oriented input normals, while APSS is limited to consistently oriented inputs. As a fast preprocess, we employed the sphere fitting without normal technique of the APSS framework to estimate the input normal directions $\boldsymbol{u}_i$ since it better preserves the surface details than simpler covariance analysis. The



(a) Original image (b) Smoothing oriented gradient fields

(c) Our constant Appro. w/ non-oriented gradient fields (d) Our isotropic linear Appro. w/ non-oriented gradient fields

**Figure 5.6:** ***Comparison with smoothing oriented gradient fields.*** *LIC gradient field visualization for Lena image (*$512 \times 512$*), using* $25 \times 25$ *neighborhood size. Directly smoothing oriented directions raises pair cancellation issue (b), constant approximation on non-oriented gradient field fails to preserves shape features (c). Thanks to the continuous MLS reconstruction, we can recompute the gradient field at arbitrary scales, the LIC visualization remains clear even in 10 times zooming-in.*



(a) Input / APSS w/o normals (b) Constant w/ non-oriented normals (c) Isotropic Linear w/ non-oriented normals

**Figure 5.7:** ***Results on 3D point sets.*** *For 3D point set (a-left), not using a normal field leads to poor quality reconstructions when using spherical MLS approximations [GG07] (a-right). The use of non-oriented normals leads to easy-to implement improvements (b-c) without the burden of having to coherently orient them. Note that constant approximations (b) are outperformed by our novel isotropic linear approximations (c) that better preserves surface features, while exhibiting much smoother and stable surface reconstructions.*

reconstructed unsigned implicit functions have been meshed using the advancing front algorithm implemented in MeshLab. Our approach is also well suited for direct raytracing [AA04a], or for the GPU accelerated resampling framework of Guennebaud et al. [GGG08] to enable real-time visualization. All the results have been made using exact surface normals (Sec 5.2.4). Figure 5.7 illustrates the ability of our approach to reconstruct very sparse data with a high fidelity.

Figure 5.8 illustrates the natural stroke simplification ability of our approach. Compared with an input image, the 2D real-time reconstruction takes place in background while the gradient information directly comes from the stroke directions. We record all points the mouse passed under the pressed mode, and initialize their gradients by the normalized vector constructed from previous point to the current one. We store these position and gradient information into a texture. When the user is drawing, this texture is updating, and the whole gradient field on the screen is recomputed dynamically. Finally, a 0-isocurve is approximated by Equation 5.7. In any moment during the interaction, the user can decide to draw new curves without taking the previous strokes into account. In this case, only new strokes are used to reconstruct the gradient field, but the previous reconstructed curves will still appear in the final result.

Thanks to our real-time implementation and the sparsity of strokes, the user can get an immediate feedback, either the constructed gradient field or the simplified curves, as illustrated in Figure 5.8. One observation in the system testing our drawing software is that it is extraordinarily difficult for untrained users to draw an

expected smooth and continuous curve by hand, even using a digital pen. Since the reconstructed curves are recomputed dynamically, they can be easily modified by adding or deleting strokes, for instance elongation, straightening or bending, see the companion video. We would like also to emphasize how coherently orienting the gradients between the different strokes would be particularly challenging on these examples in the middle of Figure 5.8.

Differential properties, such as tangential curvature, can be easily estimated based on our isotropic linear approximation, and further be used in image stylization applications. As an example, the last figure in the bottom of Figure 5.8 shows cordate curves with varying thickness. It is rendered implicitly by convolution of footprints whose size is dependent on curvatures. The detail of this rendering technique will be presented in the next chapter.

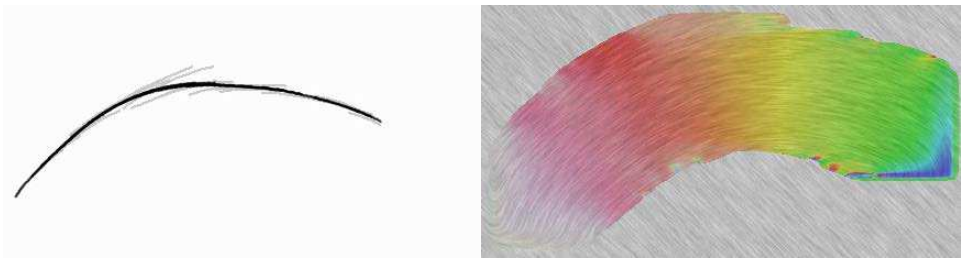
**Discussion** Both our constant and higher-order gradient fields are defined at any location, thus allowing for infinite zoom at a cost which is independent of the zoom level. On the other hand, the color information used in various stylization is only limited to the original resolution. Though we employed a bilinear interpolation, color aliasing is still observed when the image is zoomed in a lot. We believe sharper results could be obtained using more elaborated color interpolation techniques. Our gradient fields could certainly help here for better edge preservation.

In this chapter, we have focused on the approximation of *smooth* gradient fields mainly ignoring the junctions, occlusions and sharp edges that may arises in images and 3D point clouds respectively. This is however an orthogonal problem. Indeed, to deal with discontinuities, many extensions of MLS have already been proposed (e.g., [FCOS05, OGG09]), and many of them could be directly applied to our approach without major difficulties. In the same vein, we consider the choice of the filter size as a separated and more general problem. For videos, we think that the use of motion flow could increase temporal smoothing, and deal more properly with occluding contours.

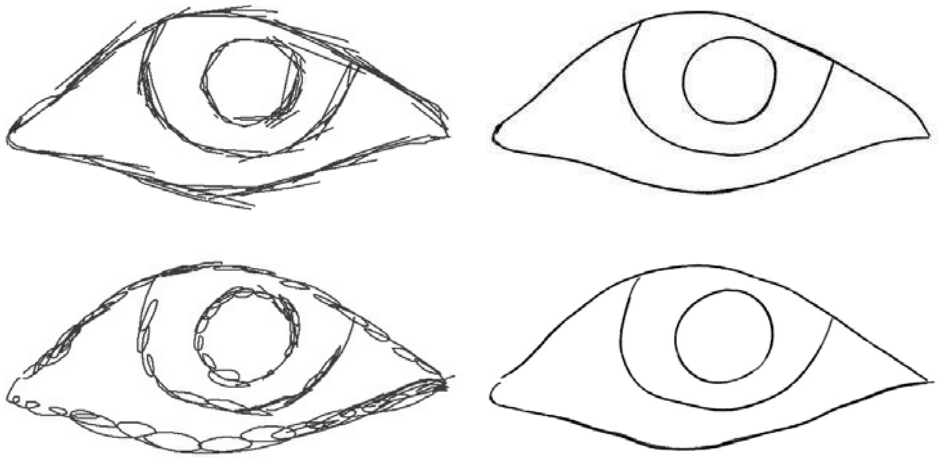
As usual with MLS surface reconstruction, the global scalar field F might present some *unwanted extra zeros* [AA06]. Since, by definition, they happen far away from the input points, they are not a problem for smoothing, resampling, or meshing with an advancing front. If evaluations happen in the whole space (e.g., for raytracing) then solution such as proposed by Guennebaud and Gross [GG07] may be used.

## 5.4 Summary

Thanks to an appropriate regularization, we have demonstrated how the classical structure tensor can be advantageously generalized to achieve higher-order approximations of non-oriented gradient fields. In particular, we have shown that the best trade-off between stability and reconstruction fidelity lies in local isotropic linear approximations. The presented MLS formalism provides continuous solu-



Intermediate constructed curves | Intermediate constructed gradient field



Sketched strokes | Resulting curves

Sketched strokes | Reconstructed curves | Varying width using curvature

**Figure 5.8:** ***Curves from sketched strokes.*** *The top row shows intermediate results and reconstructed gradient field. Gray lines represent drawn strokes, and black lines represent constructed curves. Note that large support size* $50\times50$ *is needed in order to get simplified curves. The second and third rows show that our approach for reconstruction of a 2D curve from drawn strokes is quite robust to the style: straight lines in the second row, scribbling in the third row. Our isotropic linear method estimates differential properties easily, such as tangential curvature we mentioned in the end of Section 5.2. The bottom row shows a simplified curve reconstructed from rough sketches, whose width is varying smoothly depending on tangential curvature.*

tions at arbitrary scales with a low computational complexity. Our approach is particularly effective for local surface and curve reconstructions from sparse or noisy data sets, as well as for image abstraction and stylization as we will demonstrate in the next chapter. They constitute two kinds of applications for which it is seldom possible to orient gradients consistently.

We believe that our approach could benefit many other types of data and applications. Since our MLS formulation works for arbitrary dimensions, it could be used for scientific visualization (e.g., 3D medical imaging). It may serve as a basis for multi-scale image decomposition, using increasing MLS support sizes. For videos, optic flows may be used to combine corresponding gradients across subsequent frames, for instance by advecting the MLS weight functions. Finally, our approach could also benefit various global surface reconstruction methods that rely on analytic gradient fields [KBH06, ACSTD07].

CHAPTER 6

# Structure-preserving image stylization

We have shown a huge potential of image structures for Augmented Reality and importance-driven compositing applications in the previous part, and introduced a novel approach for defining continuous non-oriented gradient fields based on moving lease squares in the previous chapter. The gradient field is regarded as one of the most important image structures in many applications, e.g. image stitching [LZPW04], image editing [PGB03] and image resizing [AS07]. In this chapter, we will present our experiments on filtering-based image stylization using this improved gradient field, such as enhanced shock filter, flow-based color abstraction and difference-of-Gaussian. We will also demonstrate the improvement using our isotropic linear approximation.

In order to further explore other key elements of image structures, we also propose a new technique that analyzes the profile of line features, which permits to distinguish between sharp and smooth features. This work will be presented in Section 6.2 of this chapter, and it was part of the publication [VVC+11].

## 6.1 Introduction

Besides reproducing artistic effects, another important goal of image and video stylization is to make them easier and faster be understood at the first glance. Many automatic stylization approaches have been proposed for efficient visual communication in recent decades [DS02, GRG04, RTF+04, WOG06]. An ideal example to illustrate this purpose is a video abstraction approach introduced by Winnemöller et al. [WOG06]. The authors suggest to abstract imagery by modifying the contrast of visually importance features. Their algorithms include two main steps:

**Color abstraction:** to reduce contrast in low-contrast regions, such as structure-preserving smoothing and color quantization;

**Line drawing:** to increase contrast artificially in high-contrast regions.

Task-based user studies [WOG06] in their paper show that participants are faster at naming abstract images, and quicker in completing a memory game using abstract images, compared to photographs, which verify their approach on efficient visual communication. Based on this prototype, various improvements on both steps have been proposed recently [KLC09, KK11]. We will review them separately in the following part of this section.

### 6.1.1 About color abstraction

Image color abstraction methods can be classified into three categories according to the operating domain: frequency domain, spatial domain and gradient domain. Frequency domain methods, such as discrete Fourier transform [Win78] and wavelet transform [ABMD92], provide a robust but fast tool to analyze the properties in different frequencies. But due to their full-space characteristic, they have to give special processes to preserve image structures [Fat09]. Gradient domain filters manipulate pixel differences instead of pixel values [BZCC10]. Though several non-photorealistic rendering filters in gradient domain has been proposed [BZCC10, MP08], it is still limited since it considers only first-order features due to the gradient definition. Based on the above consideration, we believe that filters in spatial domain will provide more flexibility and robustness. Our work in this chapter mainly focus on spatial domain methods.

The simplest way to abstract an image in spatial domain is to smooth it, for example global Gaussian smoothing. Local Gaussian smoothing can be also used to simulate foveation effect of human eyes [ZWW+10]. To prevent smoothing away image structures, many edge-preserving techniques has been proposed as an improved alternative of direct smoothing. Among of them, bilateral filter [TM98] is the most popular one. It takes both color difference and spatial distance as weights during local averaging of colors. Several papers has pointed out that bilateral filter quickly fails to preserve features if a lot of noises exist [FFLS08, SSD09, GO11]. Kuwahara filter [PPC07] is an alternative that avoid this shortcoming, but it is still difficult to ensure the spatial coherence. Recently, flow-based techniques [KLC09, KKD09] have been proposed to improve the spatial coherence. They first estimate a directional field from the image, and then use it to either rotate or reshape the filtering kernel. In this chapter, we improve flow-based image abstraction techniques by using the proposed new gradient field estimation approach in the previous chapter.

Another simple way to abstract color is color quantification. To avoid introducing hard boundaries, Winnemöller et al. [WOG06] proposed a soft color quantization method, which has temporal coherent behaviors. An extreme color quantization is to convert an image into black and white [MG08, XK08], though it may start to mislead viewers the real image contents.

Besides color information, shape is another important element of image structures can be simplified. Different from 3D scene, simplifying the shape without knowledge of the object geometry is extremely difficult. Morphological opera-

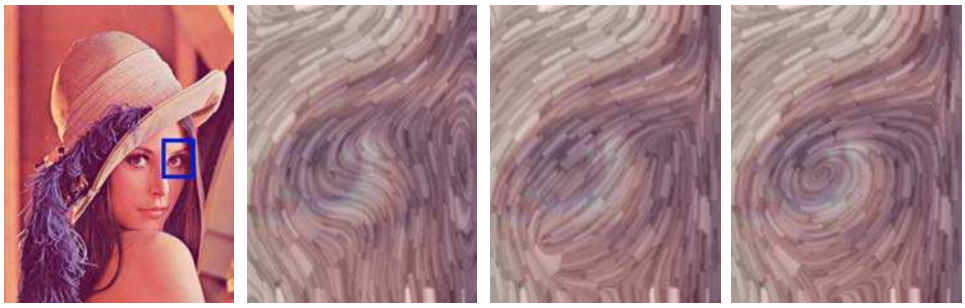
tors, such as erosion, dilation, opening (erosion followed by dilation) and closing (dilation followed by erosion), unify the color in a specific neighborhood region [BKTS06]. Enlarging the size of neighborhood or increasing the iteration will result in larger abstraction. Bousseau et al. [MG08] show that a good design of this neighborhood in videos can improve the temporal coherence. Shock filter has also been used to simplify shapes, for instance Kang et al. [KL08] iteratively apply shock filter guided by a mean curvature flow, but it only provides the ability of shrinking the shape, not expanding or simplification of the shape. In this chapter, we apply shock filter on our MLS gradient fileds to show our improvement on shape simplification. We also illustrate that with our continuous reconstruction of MLS gradient field, the simplified shape evolves dynamically when we zoom in or out the image.

### 6.1.2 About line drawing

Line drawings have always been used for illustration purposes in most scientific and artistic domains. They have also played a fundamental role in the world of animation, mostly because they allow artists to depict the essence of characters and objects with an economy of means. Unfortunately, even when artists restrict drawings to a few clean lines, hand-drawn animations require a considerable amount of skills and time. Line drawing extracted from images or videos provides an cheap but efficient alternative: lines are automatically identified in images or videos, and drawn in a variety of styles. The challenge is then two-fold: extract a set of *salient lines*, and render them in a *spatially and temporally coherent* manner.

For objects in an image or video, most existing line-based rendering techniques consider salient lines as those that best depict its *shape*. According to the recent study of Cole et al. [CGL+08], there is no consensus among various line definitions. In particular, lines drawn by human subjects do not always represent curvature extreme (ridges or valleys), but may also depict inflections (transitions between convex and concave regions). Moreover, lines from different subjects are hardly correlated. The smoother and less pronounced a surface feature is, the less correlated lines will be, until eventually the feature is too smooth to be depicted by any line at all. The only exception occurs with occluding contours that depict infinitely sharp visibility discontinuities. These observations strongly suggest that on average, lines faithfully represent only these features that exhibit *sharp-enough profiles*. However, the feature profile of an object evolves during animation, as the object gets closer or farther from the camera, and is rotated or deformed. We thus suggest to examine the sharpness of their profiles dynamically at each frame.

The second challenge of line extraction is to ensure spatial coherence in the screen and temporal coherence when video is playing. Different from a single 3D object, spatial coherence in the images and videos seems to be a big issue due to the encoded noises and object occlusion. A preprocessed image smoothing, image segmentation, or image parsing [ZZXZ09], might be useful to improve the spatial coherence. Temporal incoherence arises when features are subject to various dis-



Input image | Oriented gradient field @ 66 fps | Constant Appro. @ 60 fps | Isotropic linear Appro. @ 55 fps

**Figure 6.1:** ***Continuous glass pattern*** *on Lena image* $394 \times 510$ *using a* $25 \times 25$ *neighborhood. The brush size of the glass pattern is 10 pixels. Note the general improvement in shape preservation of Lena's eye using our isotropic linear approximation (rightmost), and the pair cancellation issues of direct smoothing of oriented gradient field (the second on the left).*

tortion events during animation: they may either split and merge, or stretch and compress. Especially, if the lines are further rendered by stylized approach, such as stroke-based rendering, two important issues raise:

- features must be tracked accurately and each splitting or merging event must be handled carefully;
- stroke style must be updated during stretching or compression events unless stylization itself will be stretched or compressed.

## 6.2 Filtering-based image stylization

Followed by the work about non-oriented MLS gradient field estimation in the previous chapter, we demonstrate that this gradient field can improve existing filtering-based image stylization techniques in this section. In particular, we show how our approach improves image stylization by systematically comparing the results obtained with our Constant Approximations (CA) and Isotropic Linear Approximations (ILA).

### 6.2.1 Continuous glass pattern

As introduced in the Chapter 2, Continuous Glass Pattern (CGP) achieves artistic effects by replacing the natural texture by a synthetic texture, rather than generating a list of strokes with different attributes [PP09].

In the previous work chapter, we have shown the formula to obtain the CGP $\mathcal{G}_{\mathrm{g}}$, given the input image $I$ and its gradient field g. The full CGP algorithm consists of five steps. Firstly, an edge-preserving smoothing $I_{EPS}$ is employed to remove

noises in the image, we use an anisotropic Kuwahara filter [PPC07]. Secondly, a gradient field g of this smoothed image is computed using first-order Gaussian derivatives. The third step is to generate a convoluting texture by applying Gaussian smoothing with a variance parameter $\sigma$ on a Gauss white noise image. Note that the parameter $\sigma$ controls the size of noise pattern, thus determines the thickness of the brushes in CGP. Then the CGP synthetic texture $\mathcal{G}_{\mathrm{g}}$ is obtained by an integration (Equation 2.2). The final result is achieved by adding $\mathcal{G}_{\mathrm{g}}$ to $I_{EPS}$: $I_{EPS} + \lambda \mathcal{G}_{\mathrm{g}}$, where the parameter $\lambda$ controls the strength of CGP.

Figure 6.1 compares CGP results based on gradient fields computed by our CA, ILA and direct smoothing the oriented gradient field respectively. To compare fairly, all other steps and parameters are the same, $\sigma = 1.67$, $\lambda = 0.75$, and the half length of the convolution $\mathrm{S} = 10$. Direct smoothing the oriented gradient field results in losing image structures, and our CA also tends to oversmooth round features, while our ILA accurately preserve the centrality of the eye accurately.

### 6.2.2 Enhanced shock filter

Shock filter is one of the oldest filters in signal processing, Rudin et al. [OR90] have first introduced it to image processing to enhance the image contrast. Kang and Lee [KL08] have reintroduced it to simplify shapes in images, and Kyprianidis and Kang [KK11] further make use of it to sharpen the image based on the gradient field. Kyprianidis and Kang [KK11] apply 1D shock filter along the tangential direction estimated by either direction smoothing that requires local flip in the neighborhood or relaxing in low-contrast regions. They iterate several times, for both directional field estimation and stylization, which limits their algorithm to a specific style.

We employ the enhanced shock filter pipeline introduced by Kyprianidis and Kang [KK11]. Given an input image and its gradient field, we first apply a 1D Gaussian smoothing along the tangent field with a neighborhood size $\mathrm{S}_t$, and then a 1D shock filter along gradient field with a neighborhood size $\mathrm{S}_s$. In this separate kernel, $\mathrm{S}_t$ controls the spatial coherence, and $\mathrm{S}_s$ controls the thickness of the shock filter. In order to remove the aliasing around the sharpening edges, we additionally apply a 1D Gaussian smoothing along the gradient direction with a very small kernel as a final step.

Thanks to the improvement of our ILA, we are able to improve the enhanced shock filter, as illustrated in Figure 6.2. Small round dots are better preserved in our ILA. Figure 6.3 shows a simplification result of line drawing. The green and red dotted box means the zooming regions with our continuous reconstruction based on MLS formalism. Note how lines dynamically split in zoomed views. And Figure 6.3-(e) and (f) compare ILA and CA in a 4 times zoomed resolution. Since our ILA derives differential quantities such as second-order curvature with a nearly-zero cost, we adapt $\mathrm{S}_s$ to adjust the size of shock filter using this curvature:

$$\mathrm{S}_s = \mathrm{S}_{\min} + (\mathrm{S}_{\max} - \mathrm{S}_{\min}) \tanh(\beta \cdot \kappa_t) \tag{6.1}$$

where $S_{min}$ and $S_{min}$ are minimal and maximal shock filter size, $\beta$ controls the influence of curvature, $\tanh(\cdot)$ is a function maps a positive value into $[0, 1]$, and $\kappa_t \approx 2l$ approximates skeleton curvature and $l$ is half of the coefficient of the quadric term in the formula of our isotropic linear approximation. The line drawing thus has a varying thickness depending on local skeleton curvature. Figure 6.3-(c) exhibits such a result that we use $S_{min} = 2$, $S_{min} = 12$ and $\beta = 1.0$. Figure 6.4 demonstrates that our method naturally exhibits better temporal coherence even though each frame is processed independently.

### 6.2.3 X-DoG

Difference-of-Gaussian (DoG) is a classical method to extract lines from an image [WOG06]. The DoG operator is the difference between two bands of an image $I$ computed by Gaussian smoothing with two different kernels:

$$DoG(\boldsymbol{x}) = G_{\sigma_c} \otimes I(\boldsymbol{x}) - \rho \cdot G_{\sigma_s} \otimes I(\boldsymbol{x}), \tag{6.2}$$

$$G_\sigma \otimes I(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{N}(\boldsymbol{x})} I(\boldsymbol{y})\, e^{-dist(\boldsymbol{x},\boldsymbol{y})^2/2\sigma^2} \tag{6.3}$$

where $\mathcal{N}(\boldsymbol{x})$ is the neighborhood set of pixel $\boldsymbol{x}$, and $dist(\boldsymbol{x}, \boldsymbol{y}) = |\boldsymbol{x} - \boldsymbol{y}|$ is a distance function between $\boldsymbol{x}$ and its neighbor pixel $\boldsymbol{y}$ in image space. $\rho$ determines the sensitivity of the detector. For small values of $\rho$, less noise is detected, but real edges become less prominent. As $\rho \longrightarrow 1$, the detector becomes increasingly unstable. We use $\rho = 0.99$ in this thesis. $\sigma_s$ is always set to $1.6\sigma_c$ to approximate closely to Laplacian-of-Gaussian. The lines are extracted by a soft threshold function:

$$\mathcal{L}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } DoG(\boldsymbol{x}) < \epsilon \\ 1 + \tanh(\varphi \cdot DoG(\boldsymbol{x})), & \text{otherwise} \end{cases} \tag{6.4}$$

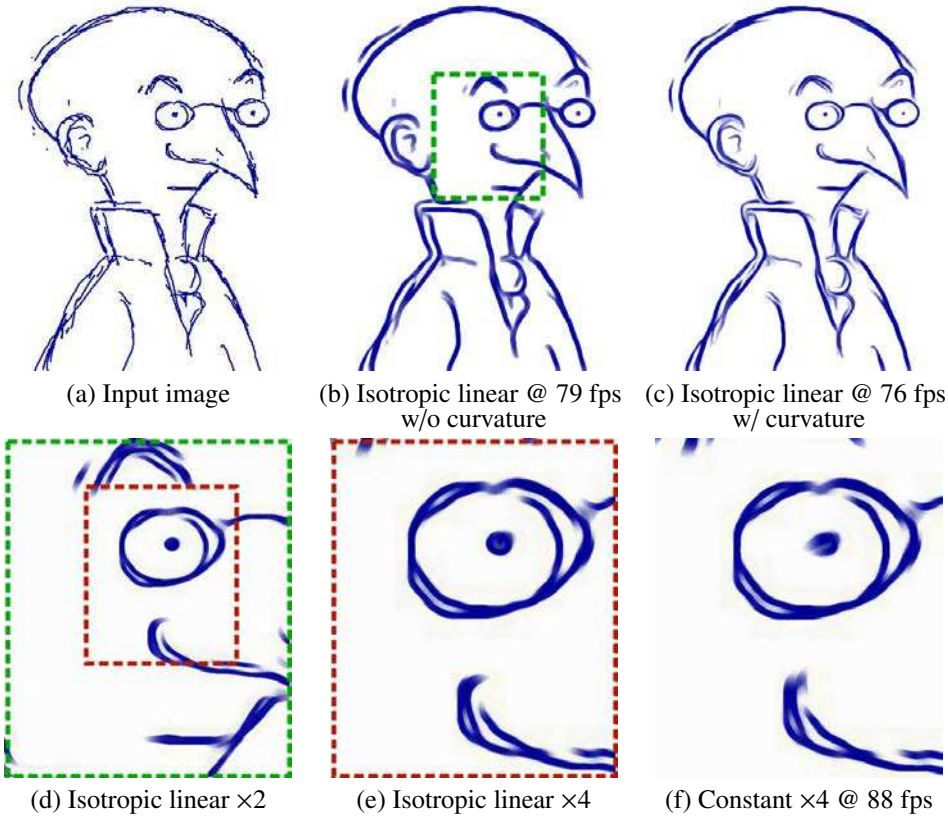
Kang et al. [KLC07] have improved the spatial coherence by guiding DoG filter using a gradient field. They separate the DoG operator into two passes, one is an 1D DoG operator along the gradient field using $\sigma_c$, and the other is an 1D Gaussian smoothing along the tangent field using $\sigma$. Based on this separate filtering kernel, $\mathcal{N}$ in Equation 6.3 becomes a 1D neighborhood, a streamline tracing along the gradient field, and $dist(\boldsymbol{x}, \boldsymbol{y})$ is the 1D tracing distance from $\boldsymbol{y}$ to $\boldsymbol{x}$. Note that parameter $\sigma_c$ controls the sparsity of lines, and $\sigma$ determines the spatial coherence aligned the tangent field.

Recently, Winnemöller et al. [Win11] have further generalized difference-of-Gaussian to provide more ability to convert the image in various styles using the same formula (X-DoG). Figure 6.5 exhibits that a black-and-white stylization can be obtained by setting $\rho < 1$, $\varphi \gg 1$, and $\epsilon < 0$ in X-DoG filter. Though X-DoG filter based on our CA sharpens the blurry image, it tends to oversmooth the original structures. While the one based on our ILA preserves well the curly features when sharpening. The improvement is even more observable when an emboss effect is employed, as shown in Figure 6.5-(b).

(a) LIC vis. of gradient field @77 fps | @72 fps

(b) Enhanced Shock filter @63 fps | @60 fps

**Figure 6.2:** ***Comparison between CA and ILA with enhanced Shock filter*** *on a* $500 \times 347$ *image using a* $9 \times 9$ *neighborhood for the MLS gradient field. We use a* 9 *pixel size for uniform Shock filtering. Our MLS constant approximation extends classical techniques based on structure tensor smoothing [KKD09] that tends to oversmooth the details (left part). The use of non-oriented gradients leads to easy-to-implement improvements without the burden of having to coherently orient them. Constant approximations are outperformed by our novel isotropic linear approximations (right part) that better preserves the image structures.*



(a) Input image

(b) Isotropic linear @ 79 fps w/o curvature

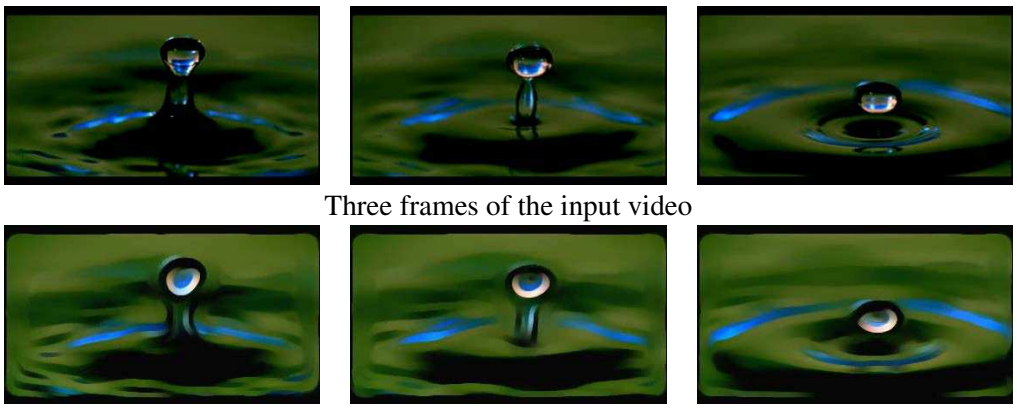
(c) Isotropic linear @ 76 fps w/ curvature

(d) Isotropic linear ×2

(e) Isotropic linear ×4

(f) Constant ×4 @ 88 fps

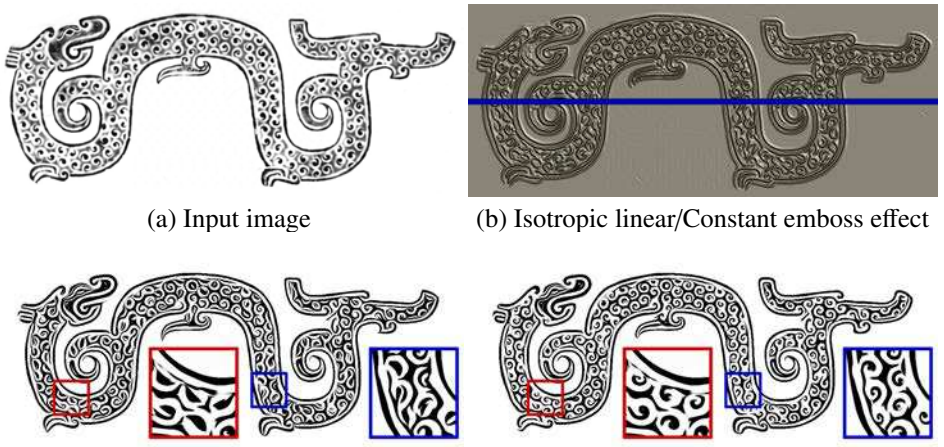
**Figure 6.3:** ***Enhanced Shock filter for line drawing simplification*** *on a* $422 \times 499$ *image using a* $23 \times 23$ *neighborhood for the MLS gradient field. We use* 2 *for tangential smoothing, and a* 10 *pixel size for uniform Shock filtering. (b), (d) and (e) show our ILA results under different resolutions with our continuous reconstruction. Note that how the lines are splitting when the resolution increases. Compared to CA (f), our ILA (b,d,e) improves the overall sharpness. Furthermore, it permits to locally adjust the shock size based on curvature to vary stylization (c).*



Three frames of the input video

Video results @ 55 fps

**Figure 6.4:** ***Enhanced Shock filter on video.*** *on a* $366 \times 206$ *video using a* $25 \times 25$ *neighborhood. We use a* 10 *pixel size for uniform Shock filtering. The companion video shows the temporal coherence of our method.*



(a) Input image

(b) Isotropic linear/Constant emboss effect

(c) Constant X-DoG @ 45 fps

(d) Isotropic linear X-DoG @ 40 fps

**Figure 6.5:** ***X-DoG filters:*** *on a* $916 \times 379$ *image using a* $29 \times 29$ *neighborhood. The original image is an ink rubbing of an engraved dragon, and the size of X-DoG filter is 9 pixels. The difference of ILA and CA is easily observed in emboss effect (b). And we use the infinite zoom ability of our approach to show how the curls are consistently better preserved using ILA (d) compared to CA (c).*

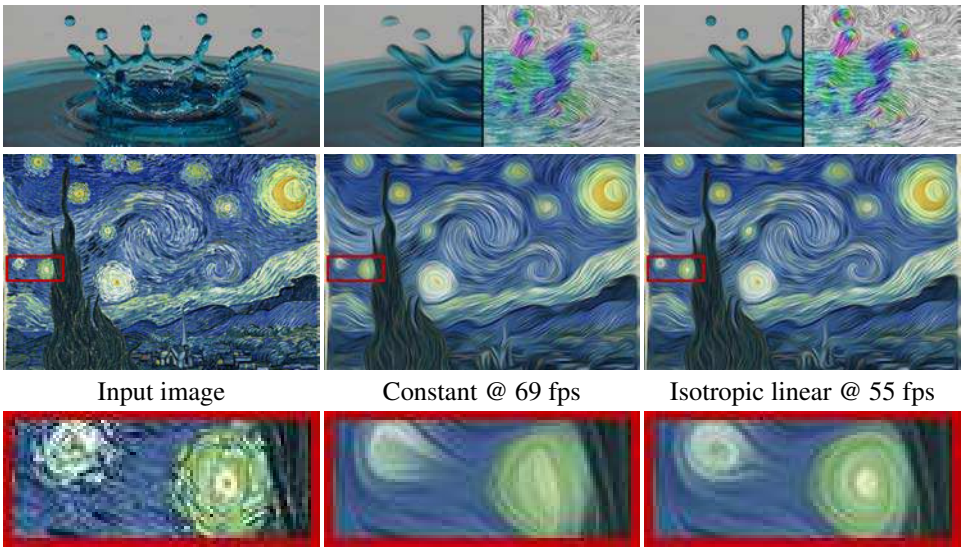


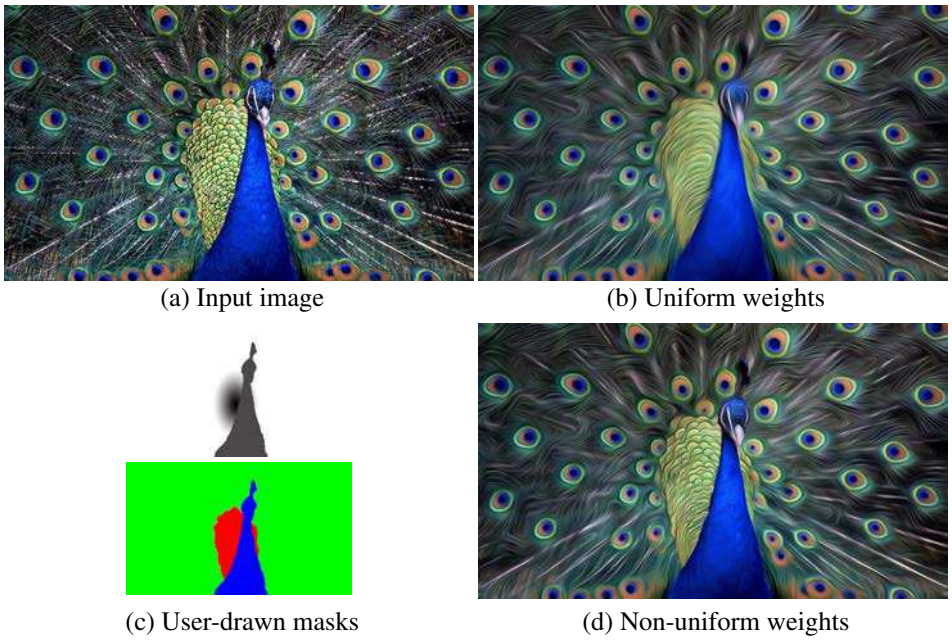

**Figure 6.6:** ***Flow-based abstraction*** *on a* $500 \times 223$ *image (on the top) and a* $500 \times 327$ *image (the second row) using a* $33 \times 33$ *neighborhood. Note the curly structures, such as the centers of stars and waterdrops, are better preserved using our isotropic linear approximation.*

### 6.2.4 Flow-based image abstraction

A simple structure-preserving color abstraction could be done by bilateral filter. In order to be more spatial coherent, flow-based bilateral filter is proposed to better preserve original structures. In our implementation, we first apply an 1D bilateral filter along the tangent field, and then an 1D bilateral filter along the gradient field. In order to avoid color blending issue, we use a large neighborhood size aligned the tangent field, and a small neighborhood size aligned the gradient field. Figure 6.6 exhibits our flow-based color abstraction on water splashing and Van Gogh's famous painting "The Starry Night". Compared to our ILA, CA tends to fail to preserve curly structures, such as centers of stars in Figure 6.6.

One main limitation of our ILA is that the user has to determine the best support size parameter. Too small support size will not ensure the improvement, while too large support size will tend to oversmooth the original structures even using our ILA, as shown in Figure 6.7-(b). Since the oversmoothing artifacts are always first observed around the object boundaries and small texture details when increasing the support size, a knowledge of image discontinuities and feature scales would helps to overcome this limitation. Figure 6.7 illustrates the use of masks to clip the neighborhood at discontinuities, and to locally adjust the filter radius.

As already noted, our CA method can be seen as a **continuous** variant of classical *structure tensor* smoothing [KKD09]. This is a key advantage the enables



(a) Input image (b) Uniform weights

(c) User-drawn masks (d) Non-uniform weights

**Figure 6.7:** ***Non-uniform weights for adaptive flow-based abstraction*** *By adjusting the neighborhood size using a hand-drawn scale image (c-top) and hand-selected segmentation (c-bottom), we locally improve the preservation of small features (like on the peacock body and head) and the preservation of original discontinuities (like between the body and the tail feathers).*

high-quality dynamic zooms without any particular overhead since the cost of our approach is linear in term of number of evaluations, i.e., in term of the number of pixels of the output image. This contrasts with approaches based on a global formulation (e.g., harmonic smoothing) for which the whole input image has always to be considered. This ability is visible in Fig 6.1, but also in the insets of Fig 6.5 using the eXtended Difference of Gaussians (X-DoG) filter [Win11]. It is better seen in the video where the lines produced by the coherence-enhancing *shock* filter [KK11] naturally separate or merge depending on the zooming level. This filter is applied to a line-drawings image in Fig 6.3. Note that such inputs provide a sparse information since most of the pixels present a null gradient.

## 6.3 Line feature analysis

We have pointed out the importance of line features in image representation, image-based modeling Augmented Reality and image stylization in the previous work chapter. But, the study of Cole et al. [CGL+08] has shown that although occluding contours are expected to be depicted in virtually all line drawings, other surface features are not systematically drawn. A simple use of occluding contours

is not enough though. Regarding this issue, we make *no attempt* at defining a new kind of image line feature in this section. Instead, our contribution consists in first providing a *generic definition* for most common image line features, then extending it to identify *feature profiles* (Section 6.3.1). We then show how these generic image line features are extracted in real-time using an implicit approach that works on the GPU (Section 6.3.2).

### 6.3.1 Feature definitions

**Feature skeleton** In the previous work chapter, we have derived our first observation that features are identified as maxima of a differential geometry invariant in a tangential direction. We call the loci of such maxima the feature *skeleton*. It is defined by

$$\mathcal{S} = \left\{ \mathbf{s} \in \mathbb{R}^2 \;\middle|\; \frac{\delta h(\mathbf{s})}{\delta \boldsymbol{\theta}(\mathbf{s})} = 0,\ \frac{\delta^2 h(\mathbf{s})}{\delta \boldsymbol{\theta}(\mathbf{s})^2} < 0 \right\}, \tag{6.5}$$

where $\mathcal{S} \subset \mathbb{R}^2$, $h : \mathbb{R}^2 \to \mathbb{R}$ is a $C^2$ height function, and $\boldsymbol{\theta} : \mathbb{R}^2 \to \mathbb{S}^2$ is a $C^1$ direction field. Both $h$ and $\boldsymbol{\theta}$ are easily instantiated to produce existing image and surface feature types.

For images, edges are obtained by taking the maximum of the gradient magnitude in its corresponding direction. This gradient can be luminance gradient, computed by Gaussian derivatives, or color gradient as we discussed in Chapter 2. Ridges and valleys are second-order features, and thus require to compute a curvature tensor $\mathbf{H}$ from which principal curvatures $k_{\max}$ and $k_{\min}$ and directions $\mathbf{t}_{\max}$ and $\mathbf{t}_{\min}$ are extracted. Ridges (resp. valleys) are obtained as maxima of $k_{\max}$ (resp. $-k_{\min}$) in the $\mathbf{t}_{\max}$ (resp. $\mathbf{t}_{\min}$) direction. Inflections are third-order features, and thus require to compute a curvature-variation tensor $\mathbf{C}$ from $\mathbf{H}$, and extract a curvature gradient $\mathbf{v}_{\max}$ from $\mathbf{C}$. They are then obtained as maxima of $\mathbf{v}_{\max}$ in its corresponding direction.

All above image feature definitions can be generalized for surface as surface edges, ridges/valleys, and surface inflections with a slight modification. For instance, surface edges are obtained by taking maxima of the surface gradient $\mathbf{g_n} = \left(-n_x/n_z, -n_y/n_z\right)$ in its corresponding direction, where $\mathbf{n} = (n_x, n_y, n_z)$ is the surface normal expressed in screen-space. Surface inflections are similar to Demarcating Curves [KST08], but are defined in screen-space.

**Feature profile** An advantage of using Equation 6.5 is that we can now reason on abstract features without having to focus on a particular definition. As an example, we take the "ripple" function illustrated in Figure 6.8. It is obvious from this example that the feature skeleton is not sufficient if one wants to convey differences between height field oscillations.

Our second observation is that all the required information to make this distinction is contained in the direction field. Indeed, classic differential geometry [dC76] tells us that for each non-singular point $\mathbf{x}$ of a direction field $\boldsymbol{\theta}$, there exists a unique

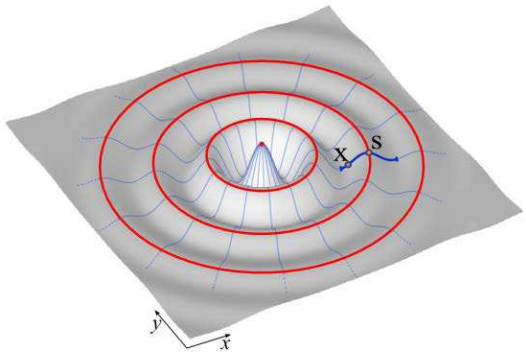



**Figure 6.8:** ***Simple ripple example:*** *we show the height function* $h(\mathbf{x}) = cos(|\mathbf{x}|)/(1 + 0.2|\mathbf{x}|)$*, a subset trajectories (in pale blue) of its directional field* $\boldsymbol{\theta}(\mathbf{x}) = \mathbf{x}/|\mathbf{x}|$*, the corresponding feature skeleton* $\mathcal{S}$ *(in red), and a feature profile (in dark blue) that goes both through points* $\mathbf{x} \notin \mathcal{S}$ *and* $\mathbf{s} \in \mathcal{S}$*.*

curve $c_{\mathbf{x}}(t)$ that passes through $\mathbf{x}$ and which tangent is everywhere equal to $\boldsymbol{\theta}$. Such a curve is called a trajectory (or integral curve); a subset of trajectories is drawn as pale blue curves in Figure 6.8. However, a trajectory may cross multiple times the feature skeleton $\mathcal{S}$. To identify the unique feature *profile* corresponding to a point $\mathbf{s} \in \mathcal{S}$, we clamp its trajectory to feature mimima (or singularities) on each side of $\mathbf{s}$. This is illustrated in Figure 6.8 by the dark blue curve. The feature profile $p_{\mathbf{s}} : (t_-, t_+) \rightarrow \mathbb{R}$ at a point $\mathbf{s} \in \mathcal{S}$ is defined as the height function along the truncated trajectory:

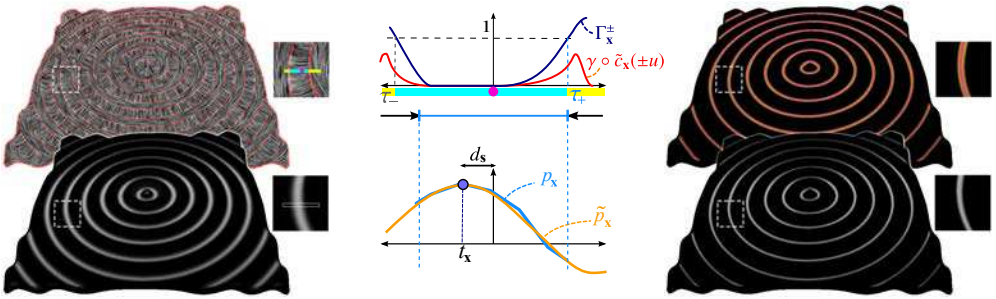
$$p_{\mathbf{s}}(t) = h \circ c_{\mathbf{s}}(t),\ t \in (t_-, t_+), \tag{6.6}$$

where $t_+$ (resp. $t_-$) is the positive (resp. negative) parametric location of either the closest minimum or nearest singularity.

An interesting property of Equation 6.6 is that it also applies to any non-singular and non-minimal point $\mathbf{x} \notin \mathcal{S}$. Hence, because of the unicity of a trajectory, for each such point $\mathbf{x}$ lying in the span of a feature profile centered at $\mathbf{s}$ (the dark blue curve in Figure 6.8), $p_{\mathbf{s}}(t)$ and $p_{\mathbf{x}}(t)$ are equal up to a parametric translation. In other words, a feature skeleton and profile can be obtained implicitly at $\mathbf{x}$ by analyzing a neighborhood along its trajectory. We make use of this property to extract feature skeleton and profiles in parallel at each pixel.

### 6.3.2 Implementation details

The extraction of features as defined above is done in three stages: 1) we compute $h(\mathbf{x})$ and $\boldsymbol{\theta}(\mathbf{x})$ per pixel for each frame; 2) we build a 1D neighborhood for each pixel $\mathbf{x}$ by following its trajectory; 3) we identify feature skeleton and profile along this neighborhood, using a fitting procedure.




(a) (b) (c)

**Figure 6.9:** ***Feature extraction:*** *(a) Feature data consists of a direction field $\boldsymbol{\theta}$ (here $t_{min}$, displayed on top using LIC with singularities in red), and a height field $h$ (here $-k_{min}$, displayed at bottom in gray-scales). (b) The trajectory $\tilde{c}_{\mathbf{x}}(t)$ is shrunk by a factor $\tau_+$ to stop at feature singularities (top); then profile data $p_{\mathbf{x}}$ is fit using a cubic polynomial $\tilde{p}_{\mathbf{x}}$ (bottom). (c) Profile parameters such as the distance $d_{\mathbf{s}}$ to the skeleton (top) and profile height $\tilde{p}_{\mathbf{x}}(t_{\mathbf{x}})$ (bottom) are spatially and temporally coherent.*

**Feature data** We have introduced the feature definition in different orders in Section 6.3.1. In practice, we compute gradients by applying first-order Gaussian derivatives on the image luminance. For second-order features, we compute a curvature tensor $\mathbf{H}$ by applying second-order Gaussian derivatives. For third-order features, we apply a Sobel filter to mean curvature $H = tr(\mathbf{H})$ to compute $\mathbf{v}_{\max}$ .

Having identified $h$ and $\boldsymbol{\theta}$ per pixel for a particular choice of image feature, we are only one step away from inspecting pixel neighborhoods: we must first locate feature singularities. Singularities of $\boldsymbol{\theta}$ are approximated with the mean angular variation in a 8-pixel neighborhood around each pixel: $\gamma_{\boldsymbol{\theta}}(\mathbf{x}) = 1 - \Sigma_{i=1}^{8} |\boldsymbol{\theta}(\mathbf{x}).\boldsymbol{\theta}_i(\mathbf{x})|/8$, with $\boldsymbol{\theta}_i(\mathbf{x})$ the orientation at neighboring pixel $i$. They are approximated by $\gamma_d(\mathbf{x}) = ||\mathbf{g}_d(\mathbf{x})||$. Feature singularities are then identified by the union of directional and contour singularities: $\gamma(\mathbf{x}) = \max(\gamma_{\boldsymbol{\theta}}(\mathbf{x}), \gamma_d(\mathbf{x}))$. Per-pixel feature data is displayed in Figure 6.9-a, using Line Integral Convolution [CL93] (LIC) for $\boldsymbol{\theta}$, which is a valid display since all our direction fields are defined modulo $\pi$. We show singularities in red; in this case they appear at places where principal curvatures are of equal magnitude (*i.e.*, at inflection points).

**Profile sampling** The second stage takes advantage of the observation made at the end of Section 6.3.1: because each non-generic and non-singular pixel $\mathbf{x}$ belongs to a unique trajectory $c_{\mathbf{x}}(t)$, we can walk along $c_{\mathbf{x}}(t)$ to find the feature profile it belongs to. In practice, we consider a first-order Taylor expansion of $c_{\mathbf{x}}(t)$ (*i.e.*, a linear neighborhood): $\tilde{c}_{\mathbf{x}}(t) = \mathbf{x} + t\ \boldsymbol{\theta}(\mathbf{x})$. This approximation is all the more valid for points in the vicinity of $S$ where we have observed that trajectories are close to simple lines. In our system, we measure $p_{\mathbf{x}}(t)$ along $\tilde{c}_{\mathbf{x}}(t)$ at $2k + 1$ samples (henceforth named $t_i, i = -k..k$) distributed uniformly on each side of $\mathbf{x}$ (we use

$k = 4$).

However, care must be taken not to go through a feature singularity. To deal with this issue, we take an approach similar to anisotropic diffusion: we shrink $\tilde{c}_{\mathbf{x}}(t)$ as soon as it comes close to a feature singularity. To do that, we first accumulate $\gamma$ values on each side of $\mathbf{x}$:

$$\Gamma^{\pm}_{\mathbf{x}}(t_i) = \sum_{k=0}^{i} \gamma \circ \tilde{c}_{\mathbf{x}}(t_{\pm k})$$

The neighborhood is then shrunk so that no feature singularity is crossed. This is done by identifying the location $\tau_+$ (resp. $\tau_-$) at which $\Gamma^+_{\mathbf{x}}$ (resp. $\Gamma^-_{\mathbf{x}}$) is greater than a threshold $\Gamma_{\max}$ (we use $\Gamma_{\max} = 1$), as illustrated at the top of Figure 6.9(b). The shrinking factor is then taken to be the minimum of $|\tau_-|$ and $|\tau_+|$. The shrunk neighborhood is resampled using $2k + 1$ uniformly distributed samples in order to have enough information for profile fitting.

**Profile fitting** The goal of the third stage of analysis is to identify the location of a potential feature skeleton along the 1D neighborhood, with additional profile information. We do this by fitting an analytic profile function $\tilde{p}_{\mathbf{x}}$ to profile data measured at $t_i$ along $\tilde{c}_{\mathbf{x}}(t)$. In practice, we take a least-squares approach, minimizing a per-pixel profile energy on the GPU:

$$E(\mathbf{x}) = \sum_{i=-k}^{k} (h \circ \tilde{c}_{\mathbf{x}}(t_i) - \tilde{p}_{\mathbf{x}}(t_i))^2.$$

We use a cubic polynomial for the analytic profile function (see Figure 6.9-b), since it has just enough degrees of freedom to identify surrounding extrema: $\tilde{p}_{\mathbf{x}}(t) = at^3 + bt^2 + ct + d$. Having a simple analytic expression for the profile at $\mathbf{x}$ allows us to identify characteristic profile properties. The profile generally exhibits two extrema $t_{\alpha,\beta} = (-b \pm \sqrt{b^2 - 3ac})/3a$. The skeleton location $t_{\mathbf{x}}$ is easily obtained by picking the one extrema for which the second-order derivative $d^2\tilde{p}_{\mathbf{x}}(t)/dt^2 = 6at + 2b$ is positive (when a single minimum is found, we ignore the pixel). Profile height and curvature are then simply given by $\tilde{p}_{\mathbf{x}}(t_{\mathbf{x}})$ and $d^2\tilde{p}_{\mathbf{x}}(t_{\mathbf{x}})/dt^2$ (since $d\tilde{p}_{\mathbf{x}}(t_{\mathbf{x}})/dt = 0$).

Figure 6.9-(c) displays results of the fitting process: the per-pixel distance to the nearest feature skeleton $d_{\mathbf{s}} = \|\mathbf{x} - \tilde{c}_{\mathbf{x}}(t_{\mathbf{x}})\|$ is shown with a color gradient, and profile height $\tilde{p}_{\mathbf{x}}(t_{\mathbf{x}})$ is displayed in gray-scales. Observe how both estimates are consistent across a feature profile, illustrating the spatial coherence of per-pixel fitting.

### 6.3.3 Results and discussion

Although line style may be strongly correlated to surface features, it is nonetheless independent of the feature extraction process. Our approach separates feature extraction and rendering to allow the user easily control the line-based stylization without considering how they are extracted. For the sake of legibility, we first show

(a) Input image (b) Edges (c) Ridges & Valleys (d) Inflections

**Figure 6.10:** ***Various of luminance features.*** *All features are extracted under the assumption that features can be located at maxima of some differential geometry invariant. Here we exhibit all the possible feature lines our approach can achieve. Luminance edges are first-order features, ridges and valleys are second-order ones, and inflection represents third-order features which are start to be less frequent in line-drawing literature. Note that we render ridges as bright colors to reveal that ridges are consist of pixels with maximal luminance in their neighborhood.*

(a) Luminance of a 3D object (b) Lee et al. 's approach (c) Our approach

**Figure 6.11:** ***Comparison with Lee et al. [LMLH07].*** *Rendering luminance valleys from (a) with the method of Lee et al. (b) produces satisfying results for thick-enough lines (left) but raises noise issues with thin lines (right). Opacity thresholds are chosen to obtain the cleanest possible lines. With Implicit Brushes (c), feature extraction and line stylization are controlled independently, which produces clean line renderings for any choice of line thickness.*

two results with line-based stylization. Figure 6.10 exhibits various of line features blended with the input image, based on our uniform extracting framework. This is contrast with previous image-based techniques where extraction and stylization are part of a single process. Take FDoG line extraction as an example, less lines will be detected if their width is reduced. Figure 6.11 illustrates a comparison between our method and the method of Lee et al. [LMLH07] with identical input data and similar styles. With the method of Lee et al., only opacity can be modulated after the 2D fitting process, and one must modify feature extraction to control line thickness. As a result, rendering of thin lines requires the identification of small features, which raises noise issues. With our technique, feature extraction is independent of stylization choices, which allows us to create clean renderings even with thin lines.

In order to achieve various line-based rendering styles, we employ a convolution approach that mimics the contact of a brush on canvas and easily takes fea-

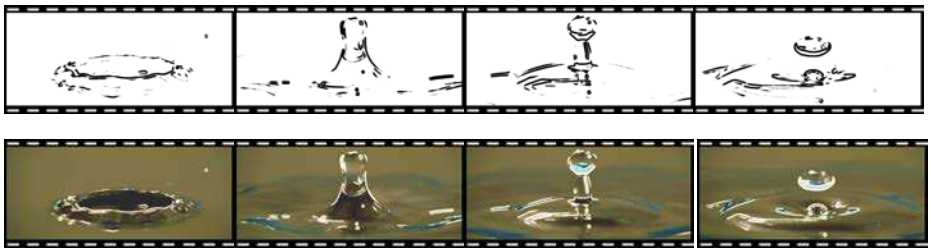


**Figure 6.12:** ***A stylized video.*** *Frames of a water drop video where luminance edges are depicted with lines of varying thickness, either drawn in black (top), or drawn in white over the original image (bottom), both with a disk footprint.*

(a) (b) (c)

**Figure 6.13:** ***Image enhancement:*** *(a) Original image. (b) Enhanced using LoG. (c) Enhanced using LoG footprint.*

ture profile into account. Intuitively, our stylization technique consists in stamping brush footprints of various size, orientation, colors and opacities at points that are close-enough to a feature skeleton with a sharp-enough feature profile. This stamping process is done by a convolution implicitly, which can be computed in parallel. The stylized lines that emerge from this process inherit the spatial and temporal coherence of image features.

Figure 6.12 shows our result on four frames of a video using a round solid footprint. Though we only use a disk footprint in this case, lines are starting to adapt their thickness according to the sharpness of the profiles. We believe also that identifying features properties at every point in the image is an efficient way to control noise reduction. As a first experiment, we have tried to first use a Laplacian of Gaussians function as a footprint, and then convolute implicitly taking the sharpness of the profile as the weights for each pixel, finally add the results of the convolution back into the image to increase the contrast. Since profiles only near features are sharp enough, the contrast in this regions are enhanced, thus the noise in the regions far away from any features will not be boosted. It results in a compelling and controllable image enhancement technique that exaggerates details only at feature locations, as seen in Figure 6.13. This direction of research deserves future endeavor.

The performance of our system in this chapter is mainly dependent on fill-rate: it varies with image resolution, the number of anisotropic diffusion iterations, and footprint size. For example, using a NVidia G-480, it runs at 82 fps at a resolution

of $1024 \times 768$ with a footprint of radius 10 pixels. Performances drop down with increasing iterations: it runs at 56, 40 and 31 fps for 10, 20 and 30 iterations respectively.

Our stylization technique produces temporal coherent lines from images and videos in a variety of styles. In comparison, other methods are either limited in terms of stylization abilities [ND04, LMLH07], since they only provide thickness control; or they are prone to temporal artifacts even on static objects [KDMF03, BCGF10], mainly with split or merge events. The flexibility of our approach comes at a price though: user control is less direct with our convolution-based rendering than with stroke-based techniques. For instance, perturbations are produced via a global screen-space texture; and stylized lines automatically end at feature endpoints based on the choice of weight function. In other words, there is no simple solution for applying a stroke texture along a line. Although it is a limitation that prevents accurate local control of line style, it may also be seen as an advantage for the applications we target in this paper: style is designed once and for all and applied either in real-time for dynamic systems, or as a batch process for compositing pipelines.

## 6.4 Summary

We have presented two structure-preserving image stylization techniques, that improves filtering-based color abstraction and line-based rendering. Thanks to the higher-order approximation of gradient fields and robust profile analysis, we are able to stylize images and videos in both spatial and temporal coherent manner. Since our approach works in screen space, it can naturally analyze 3D objects using their depth, normals or luminance buffers.

Our line feature extraction technique works for a range of image feature types, including edges, ridges, valleys and inflections. However, the choice of feature has an influence on a feature profile extent: indeed, with feature definitions of increasing order, more singularities occur, which leads to more profile clamping on average. Another limitation of our approach in this chapter is that it ignores junctions, precisely because they are located at directional singularities. Due to the lack of junction identification knowledge, color oversmoothing is difficult to avoid for image abstraction on junctions, and in the same vein, the continuity of a line is impossible to guarantee if it intersect other lines. One of the main future work would thus be to analyze the multiple orientation fields instead of only one direction, for instance steerable filter [FAoTMLVG91].

Another direction of research deserves future endeavor is to investigate our approach in multiple scale space. We require the user to select the filter size in filtering-based image stylization, and to assign a threshold to define a neighborhood size for computing the profile sharpness. Moreover, both image abstraction and line drawing rely on a directional field that is reconstructed in a specific support size. An arbitrary value for any of them cloud lead to a unexpected result. We believe

that local image structures are better observed in a specific scale. We thus have to find an appropriate scale for each position of images or videos. Before developing an automatic algorithm to directly estimate this scale, an easier way may be build a Gaussian scale space hierarchy of structures [OBBT07] to study how our approach behaves in increasing scales.

# Part III

# Discussion and conclusion

CHAPTER 7

# Conclusion and future work

Along this thesis, we have explored the use of contours and gradients, as image structures, for Focus+Context rendering in Augmented Reality and importance-driven image synthesis using multiple rendering styles. We have also introduced a new local approximation method for discrete non-oriented gradient fields that better preserve image structures, and a feature profile analysis approach via fitting techniques that permits to distinguish between sharp and smooth features. We have further demonstrated how our improved image structure analysis benefits various image and video stylization applications.

We have first introduced an improved Focus+Context compositing approach for Augmented Reality in Part I. In Chapter 3, we have presented a cutaway-based approach for visualizing underground pipelines, which demonstrates that the use of screen segmentation and extracted lines can provide depth cues and reveal correct occluding order. Our approach does not require an accurate reconstruction of the 3D environment and runs on-line on modern hardware. Depth cues to reveal occlusion relationships is provided by using characteristic features extracted from video frames. To enhance the perception of occluding order, the extracted features are either directly rendered, or used to create hybrid blending masks: we thus ensures that the resulting visual cues are clearly noticeable.

We have introduced an importance-driven image synthesis approach in Chapter 4 that further extends Focus+Context compositing to a larger range of applications, not only AR, but also on images, videos and 3D scenes. An importance map, either derived from saliency estimation or designed by the user, is introduced both in the creation of the multiple styles and in the final composition. Our approach accommodates a variety of stylization techniques, such as color desaturation, line drawing, blurring, edge-preserving smoothing and enhancement. We have shown that how an importance map guides both stylization and synthesis, and how feature lines play a key role in revealing shapes.

The key of our success in compositing Focus+Context in AR introduced in Part I, is the use of gradients and contours. But, the improvement is limited to the quality of gradient field estimation and line extraction, for instance the spatial

and temporal coherence, and flexibility for user control. To overcome some limitations, we have introduced a new gradient field estimation method and a feature line profile analysis method, and demonstrated their advantages in image stylization applications in Part II.

In Chapter 5, we have presented a new approach for defining continuous and non-oriented gradient fields from an image. Our approach is built on a moving least square formalism where the key is the computation of higher-order local approximations of non-oriented input gradients. In particular, we have shown that our novel isotropic linear approximation outperforms its lower-order alternative: image structures are much better preserved, and instabilities are significantly reduced. We believe that its ease of implementation (on both CPU and GPU), and its small performance overhead will find a widespread usage in graphics applications.

In Chapter 6, we have first demonstrated that our MLS gradient field improves various image stylization techniques, such as continuous glass pattern, flow-based image abstraction, extended DoG and enhanced shock filter. We have later used the gradient information for a feature line extraction approach, that not only extracts locations of features but also analyzes their profiles. The profile is analyzed via a 1D fitting along the gradient field, and permits to distinguish between sharp and smooth features. Profile parameters are then mapped to stylistic parameters such as brush orientation, size or opacity to give rise to a wide range of line-based styles. Finally, stylized lines are rendered in real time by a convolution of a brush footprint along the feature skeleton implicitly. Our approach is also naturally temporal coherent, without requiring preprocessing or tracking.

## 7.1 Future Work

All our proposed approaches in this thesis are some first steps toward a longer-term project that is, to extend the classical image-based applications, to create a bridge between input images and target applications by image structures. Image structure analysis methods that we have presented in this thesis, such as gradient fields and lines, are only dedicated to two kinds of applications that are Augmented Reality and image stylization. I personally believe that many further extensions are possible and will allow to really build the bridge between source data and target applications. It would be interesting first to process many other data sources, such as videos, volume data. For higher dimension data, the main challenge is to overcome the computation limitation. For other applications, it might be important to investigate other image structures, such as scales, motion flow in video streams. Richer is our understanding of image structures, larger is the range of data source, and wider are the applications we can serve. For a short-term, we are especially interested in the behavior of our MLS gradient field in video spatial-temporal cube, volume data, and 3D mesh model.

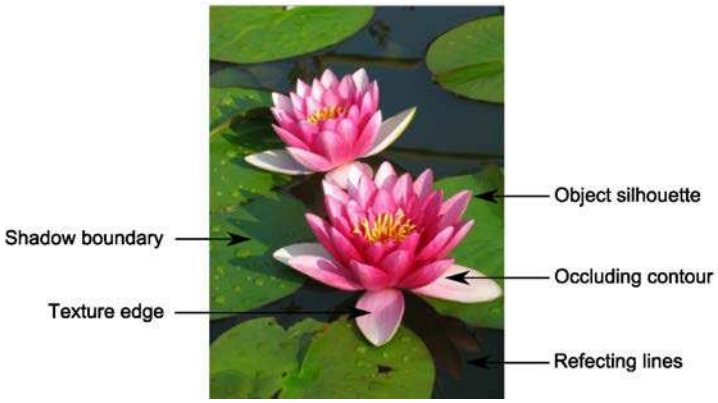



**Figure 7.1:** ***Different types of lines in an image.***

### 7.1.1 Exploring advanced image structures

**Scale.** The most important parameter in our MLS gradient field is the support size. Large support size will definitely produce a very smooth gradient field, but unfortunately it will also easily smooth out some important but small-scale details. In this case, such a global support size will be not adequate, even though the user adjust it carefully. An automatic method to decide this support size locally refers to multi-scale theory in computer vision. Lindeberg's work in 1991 [Lin96] provides us an executable approach of this theory. DeCarlo and Santella [DS02] build a hierarchical segmentation in scale-space, and automatically select the correct level of the hierarchy according to a perceptual model relying on eye movement data. Orzan et al. [OBBT07] compute a perceptually meaningful hierarchy of image edges using Gaussian scale-space theory, and demonstrate its use in various applications. Therefore, scale-space theory will be useful to automatically determine local support sizes.

**Different lines.** Lines extracted in an image have many different types according to what they represent, such as silhouettes of objects, contours produced by occlusion, edges in object textures, folds of objects or even boundaries of shadows. Figure 7.1 illustrate different kinds of lines exist in a single image. Classifying them will certainly benefit line-based stylization and to improve depth cues in Focus+Context rendering for AR. As a pioneer work, Geusebroek et al. [GvdBSG01] exploit the Gaussian scale-space paradigm for color images to define a framework for robust measurement of object reflectance. Illumination and geometrical invariant properties are derived from the reflectance model. But, it is limited to several strict imaging conditions, thus difficult to work on any natural image.

**Multiple directions.** The MLS gradient field we have introduced in this thesis is based on the assumption that only one gradient direction exists for each position of

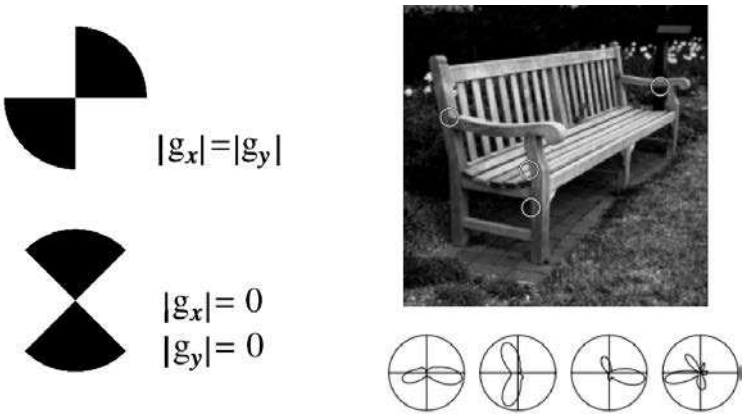



**Figure 7.2:** ***Using multiple directions.*** *The left figure shows two simple examples where traditional gradient computation using x and y directional derivatives does not find the correct gradient directions. The right figure illustrates a steerable wedge filters result [SF96], where 18 basis filters are used. Polar plots of oriented energy maps are shown in the bottom figures. They respectively identify a horizontal edge, a vertical edge, a corner and a "T" junction respectively.*

an image. In fact, this is no more the case on the corners or junctions. Figure 7.2-Left shows that traditional gradient computations using directional derivatives fail at junctions: the directional derivatives along x- and y- frames ($g_x$ and $g_y$) have either zero magnitude or equal magnitude, which leads to detect a completely wrong direction. Steerable filter [FAoTMLVG91] is an efficient architecture for creating filters at arbitrary orientation by a simple linear combination of basis filters. The output of this resulting filter is computed analytically for any orientation. Steerable wedge filter [SF96] is an extended version that fixed the symmetric limitation of steerable filter. An example of steerable wedge filter is shown in Figure 7.2-Right.

### 7.1.2 Structures beyond image

In parallel, it is also important to investigate structure estimation for a larger range of data source. Nowadays, we can capture information beyond traditional RGB color, like depth. Considering time for videos is also a natural next step to extend our MLS gradient field.

We have experimented our MLS gradient field on video in a frame-by-frame manner. Though it seems to be quite temporal coherent in the regions where features exist, popping and flickering artifacts are observed in the region which looks uniform but encodes a lot of noises in fact. One solution is to enlarge the neighborhood size in such places where there is no image features near-by, to enforce the accuracy in each frame. But estimating a gradient field in a "uniform" region is not a natural way to completely solve the flickering issue, since the random noises are fully different in each frame. A natural way would be to take the video as a 3D

cube, and apply the MLS reconstruction algorithm directly on the 3D data. In this case, it might be necessary to use the optical flow. But, the computation of a 3D MLS algorithm might immediately increase due to the cubical neighborhood.

An algorithm that takes the video as a 3D cube has a large limitation, that it can only be done off-line, since not only the previous frames but also the future frames are needed to process the current frame. An approach that uses only the previous frames and the current one, as suggested by Paris [Par08], might provide an online alternative solution that still ensures the temporal coherence.

### 7.1.3 Interactive image structures

It would be also interesting to invite the user into the loop for extracting image structures. For image and video stylization, this would provide the artists with more freedom to create their desired stylized images. Zhang et al. [ZHT07] introduce an interactive tensor field design tool that allows the user to create tensor fields over 3D surfaces either from sketching or by modifying the tensor field of the original image. This manipulation is done by controlling singularities. For instance, merging or removing singularities can result in locally smoothing this tensor field. But, image singularities are not only degenerates points, but also different kinds of lines. Figure 6.7 in the previous chapter have shown that object contours can locally improve the preservation of original boundary discontinuities in flow-based image abstraction.

Most of previous work discusses creation and manipulation for different image structures separately. But, these structures usually impact each other in a single image. For instance, gradient field around object boundaries always follow the direction of silhouettes, no matter inside or outside the object, and feature scales are usually small in the regions where gradient fields have large variation. A promising research direction is thus to explore an interactive tool that integrates different user-defined image structures, such as object contours, gradient field, feature scales, and even multiple directions. For instance, the computation of a gradient field should only consider color information in the same object region not across object boundaries, to better preserve the original structures. And using a smaller neighborhood size in places where users are more sensitive will preserves more user-interested details, like faces and texts [ZZXZ09].

Another important element for a good interactive tool is to consider how much effort is required for the user to finish an interaction task. For this purpose, an accurate result constructed from rough interactions will certainly outperform any system that requires accurate interactions: this is not the case in most previous work. Our curve reconstruction approach introduced in Section 5.3 of Chapter 5 shows a huge potential to simplify rough sketched strokes. But, in order to become an easy-use but powerful interaction tool, the most challenge work is to analyze many important features, such as singularities, scales, multiple directions, in a unified interactive framework based on a moving least square formulism, and provide a WYSIWYG feedback in real time.

# Publications


[CLLP12] Chen Jiazhou, Liu Yanli, Lin Naiyang, Peng Qunsheng. Photorealistic Visualization of Underground Pipelines based on Augmented Reality (in Chinese). Journal of Computer-Aided Design & Computer Graphics, 2012.

[CCPP12] Chen Leiying, Chen Jiazhou, Pan Bin, Peng Qunsheng. Semantic-Based User Interests Computation (in Chinese). Proceedings of CAD/CG (Journal of Image and Graphics), 2012.

[VVC+11] Romain Vergne, David Vanderhaeghe, Chen Jiazhou, Pascal Barla, Xavier Granier, and Christophe Schlick. Implicit Brushes for Stylized Line-based Rendering. Computer Graphics Forum (Proceedings of Eurographics), 2011. (**3rd Best Paper award in Eurographics 2011**)

[CCG+11] Chen Jiazhou, Chen Yujun, Xavier Granier, Wang Jingling, Peng Qunsheng. Importance-Driven Composition of Multiple Rendering Styles. Proceedings of Computer-Aided Design and Computer Graphics (CAD/Graphics), 2011.

[CGLP10] Chen Jiazhou, Xavier Granier, Lin Naiyang, Peng Qunsheng. On-line visualization of underground structures using context features. Proceedings of Virtual Reality Software and Technology(VRST)[short paper], 2010.

[ZQC+10] Zhong Fan, Qin Xueying, Chen Jiazhou, Hua Wei, Peng Qunsheng. Confidence-Based Color Modeling for Online Video Segmentation. Asian Conference on Computer Vision(ACCV), 2009.

[ZQC+09] Zhong Fan, Qin Xueying, Chen Jiazhou, Mo Mingzhen, Peng Qunsheng. Real-time post-processing for online video segmentation (in Chinese). Chinese Journal of Computers, 2009.

# Bibliography


[AA04a] Anders Adamson and Marc Alexa. Approximating bounded, non-orientable surfaces from points. In *Shape Modeling International*, pages 243–252, 2004.

[AA04b] Marc Alexa and Anders Adamson. On normals and projection operators for surfaces defined by point sets. In *Proceedings of Eurographics Symposium on Point-Based Graphics*, pages 149–156, 2004.

[AA06] Anders Adamson and Marc Alexa. Anisotropic point set surfaces. In *Afrigraph '06: Proceedings of international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 7–13, 2006.

[AAC+06] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin, and R. Szeliski. Photographing long scenes with multi-viewpoint panoramas. In *ACM Transactions on Graphics*, volume 25, pages 853–861. ACM, 2006.

[ABMD92] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 1(2):205–220, 1992.

[ACSTD07] Pierre Alliez, David Cohen-Steiner, Yiying Tong, and Mathieu Desbrun. Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of Eurographics Symposium on Geometry Processing 2007*, pages 39–48, 2007.

[ADA+04] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. In *ACM Transactions on Graphics*, volume 23, pages 294–302. ACM, 2004.

[Ait10] Miika Aittala. Inverse lighting and photorealistic rendering for augmented reality. *The Visual Computer*, 26(6-8):669–678, 2010.

[AK04] Nina Amenta and Yong Joo Kil. Defining point-set surfaces. *ACM Transactions on Graphics*, 23(3):264–270, 2004.

[AS07] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. In *ACM Transactions on Graphics*, volume 26, page 10. ACM, 2007.

[AST09] Ben Avery, Christian Sandor, and Bruce H. Thomas. Improving spatial perception for augmented reality x-ray vision. In *IEEE Conference of Virtual Reality (VR)*, pages 79–82, 2009.

[BA04] M. A. Badamchizadeh and A. Aghagolzadeh. Comparative study of unsharp masking methods for image enhancement. In *International Conference on Image and Graphics (ICIG)*, pages 27–30. IEEE Computer Society, 2004.

[BCGF10] Pierre Bénard, Forrester Cole, Aleksey Golovinskiy, and Adam Finkelstein. Self-similar texture for coherent line stylization. In *Proceedings of NPAR*, pages 91–97, 2010.

[BF08] Michael Burns and Adam Finkelstein. Adaptive cutaways for comprehensible rendering of polygonal scenes. *ACM Transactions on Graphics*, 27(5):124:1–124:9, 2008.

[BFO92] Michael Bajura, Henry Fuchs, and Ryutarou Ohbuchi. Merging virtual objects with the real world: seeing ultrasound imagery within the patient. In *ACM Transactions on Graphics*, pages 203–210, 1992.

[BG87] J. Bigün and G. H. Granlund. Optimal orientation detection of linear symmetry. In *Proceedings of IEEE International Conference on Computer Vision*, pages 433–438, 1987.

[BH04] Ryan Bane and Tobias Hollerer. Interactive tools for virtual x-ray vision in mobile augmented reality. In *International Symp. on Mixed and Augmented Reality (ISMAR)*, pages 231–239. IEEE Computer Society, 2004.

[BKTS06] A. Bousseau, M. Kaplan, J. Thollot, and F.X. Sillion. Interactive watercolor rendering with temporal coherence and abstraction. In *Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, pages 141–149. ACM, 2006.

[BN92] T. Beier and S. Neely. Feature-based image metamorphosis. *Computer Graphics*, 26(2):35–42, 1992.

[BNTS07] Adrien Bousseau, Fabrice Neyret, Joëlle Thollot, and David Salesin. Video watercolorization using bidirectional texture advection. *ACM Transactions on Graphics*, 26(3), 2007.

[BRV+10] Stefan Bruckner, Peter Rautek, Ivan Viola, Mike Roberts, Mario Costa Sousa, and Meister Eduard Gröller. Hybrid visibility compositing and masking for illustrative rendering. *Computers and Graphics*, 34(4):361–369, 2010.

[BTM06] Pascal Barla, Joëlle Thollot, and Lee Markosian. X-toon: An extended toon shader. In *International Symposium on Non-Photorealistic Animation and Rendering*, 2006.

[BVL+06] Thomas Brox, Rein Van Den Boomgaard, Francois Lauze, Joost Van De Weijer, Joachim Weickert, and Pierre Kornprobst. Adaptive structure tensors and their applications. In Joachim Weickert and Hans Hagen, editors, *Visualization and image processing of tensor fields*, volume 1, chapter 2, pages 17–47. Springer, January 2006.

[BWHN07] Christoph Bichlmeier, Felix Wimmer, Sandro Michael Heining, and Nassir Navab. Contextual anatomic mimesis hybrid in-situ visualization method for improving multi-sensory depth perception in medical augmented reality. In *International symposium Mixed and Augmented Reality (ISMAR)*, pages 1–10. IEEE Computer Society, 2007.

[BZCC10] P. Bhat, C.L. Zitnick, M. Cohen, and B. Curless. Gradientshop: A gradient-domain optimization framework for image and video filtering. *ACM Transactions on Graphics*, 29(2):10, 2010.

[Can87] J. F. Canny. A computational approach to edge detection. *Readings in computer vision: issues, problems, principles, and paradigms*, pages 184–203, 1987.

[CCG+11] Jiazhou Chen, Yujun Chen, Xavier Granier, Jingling Wang, and Qunsheng Peng. Importance-driven composition of multiple rendering styles. In *Computer-Aided Design and Computer Graphics (CAD/Graphics)*, pages 79–86. IEEE Computer Society, 2011.

[CCPP12] Leiying Chen, Jiazhou Chen, Bin Pan, and Qunsheng Peng. Semantic-based user interests computation (to appear). *Journal of Image and Graphics*, 2012.

[CCT+09] T. Chen, M.M. Cheng, P. Tan, A. Shamir, and S.M. Hu. Sketch2photo: internet image montage. In *ACM Transactions on Graphics*, volume 28, page 124, 2009.

[CDF+06] Forrester Cole, Doug DeCarlo, Adam Finkelstein, Kenrick Kin, Keith Morley, and Anthony Santella. Directing gaze in 3d models with stylized focus. *Proceedings of Eurographics Symposium on Rendering*, pages 377–387, 2006.

[CGL+08] F. Cole, A. Golovinskiy, A. Limpaecher, H.S. Barros, A. Finkelstein, T. Funkhouser, and S. Rusinkiewicz. Where do people draw lines? In *ACM Transactions on Graphics*, volume 27, page 88. ACM, 2008.

[CGLP10] Jiazhou Chen, Xavier Granier, Naiyang Lin, and Qunsheng Peng. On-line visualization of underground structures using context features. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology (VRST)*, pages 167–170. ACM, 2010.

[CH06] Chris Coffin and Tobias Hollerer. Interactive perspective cutaway views for general 3d scenes. In *Symp. 3D User Interfaces (3DUI)*, pages 25–28. IEEE Computer Society, 2006.

[CL93] B. Cabral and L.C. Leedom. Imaging vector fields using line integral convolution. In *Proceedings of 20th annual conference on Computer graphics and interactive techniques*, pages 263–270. ACM, 1993.

[CLLP12] Jiazhou Chen, Yanli Liu, Naiyang Lin, and Qunsheng Peng. Photorealistic visualization of underground pipelines based on augmented reality (to appear). *Journal of Computer-Aided Design and Computer Graphics*, 2012.

[CM02] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[CPD07] J. Chen, S. Paris, and F. Durand. Real-time edge-aware image processing with the bilateral grid. In *ACM Transactions on Graphics*, volume 26, page 103. ACM, 2007.

[CTM08] J. Chen, G. Turk, and B. MacIntyre. Watercolor inspired nonphotorealistic rendering for augmented reality. In *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, pages 231–234. ACM, 2008.

[dC76] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976.

[DCS10] Arindam Dey, Andrew Cunningham, and Christian Sandor. Evaluating depth perception of photorealistic mixed reality visualizations for occluded objects in outdoor environments. In *Symp. 3D User Interfaces (3DUI)*. IEEE Computer Society, 2010.

[DFRS03] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. In *ACM Transactions on Graphics*, volume 22, pages 848–855. ACM, 2003.

[DS02] D. DeCarlo and A. Santella. Stylization and abstraction of photographs. In *ACM Transactions on Graphics*, volume 21, pages 769–776. ACM, 2002.

[ED04] Elmar Eisemann and Frédo Durand. Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics*, 23(3):673–678, 2004.

[FAD02] Chris Furmanski, Ronald Azuma, and Mike Daily. Augmented-reality visualizations guided by cognition: Perceptual heuristics for combining visible and obscured information. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, page 215. IEEE Computer Society, 2002.

[FAoTMLVG91] W.T. Freeman, E.H. Adelson, Massachusetts Institute of Technology. Media Laboratory. Vision, and Modeling Group. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

[Fat09] R. Fattal. Edge-avoiding wavelets and their applications. In *ACM Transactions on Graphics*, volume 28, page 22. ACM, 2009.

[FBS05] J. Fischer, D. Bartz, and W. Straber. Stylized augmented reality for improved immersion. In *Virtual Reality, 2005. Proceedings. VR 2005. IEEE*, pages 195–202. Ieee, 2005.

[FCOS05] Shachar Fleishman, Daniel Cohen-Or, and Cláudio T. Silva. Robust moving least-squares fitting with sharp features. *ACM Transactions on Graphics*, 24(3):544–552, 2005.

[FFLS08] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. In *ACM Transactions on Graphics*, volume 27, page 67. ACM, 2008.

[FHT08] Jan Fischer, Michael Haller, and Bruce H Thomas. Stylized depiction in mixed reality. *The International Journal of Virtual Reality*, 7(4):71–79, 2008.

[FMS93] Steven Feiner, Blair Macintyre, and Dorée Seligmann. Knowledge-based augmented reality. *Commun. ACM*, 36(7):53–62, 1993.

[FWK+95] Steven Feiner, Anthony Webster, Theodore Krueger, Blair MacIntyre, and Keller Edward. Architectural anatomy. *Presence: Teleoperators and Virtual Environments*, 4(3):318–325, 1995.

[G+69] L. Glass et al. Moire effect from random dots. *Nature*, 223(5206):578–580, 1969.

[GG07] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. *ACM Transactions on Graphics*, 26(3), 2007.

[GGG08] Gaël Guennebaud, Marcel Germann, and Markus Gross. Dynamic sampling and rendering of algebraic point set surfaces. *Computer Graphics Forum*, 27(2):653–662, 2008.

[GO11] E.S.L. Gastal and M.M. Oliveira. Domain transform for edge-aware image and video processing. *ACM Transactions on Graphics*, 30(4):69, 2011.

[GRG04] B. Gooch, E. Reinhard, and A. Gooch. Human facial illustrations: Creation and psychophysical evaluation. *ACM Transactions on Graphics*, 23(1):27–44, 2004.

[GvdBSG01] J.M. Geusebroek, R. van den Boomgaard, A.W.M. Smeulders, and H. Geerts. Color invariance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(12):1338–1350, 2001.

[GWYY10] Minglun Gong, Liang Wang, Ruigang Yang, and Yee-Hong Yang. Real-time video matting using multichannel poisson equations. In *Graphics Interface*, 2010.

[Hal04] Michael Haller. Photorealism or/and non-photorealism in augmented reality. In *International conference on Virtual Reality Continuum and its Applications in Industry (VRCAI)*, pages 189–196, 2004.

[HDD+92] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of SIGGRAPH*, pages 71–78, 1992.

[HE04] J. Hays and I. Essa. Image and video based painterly animation. In *Proceedings of symposium on Non-photorealistic animation and rendering*, pages 113–120, 2004.

[HEK07] D. Hoiem, A.A. Efros, and T. Kanade. *Seeing the world behind the image: Spatial layout for 3d scene understanding*. PhD thesis, Robotics Institute, Carnegie Mellon University, August 2007.

[Her98] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of SIGGRAPH*, pages 453–460, 1998.

[HLB05] M. Haller, F. Landerl, and M. Billinghurst. A loose and sketchy approach in a mediated reality environment. In *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 371–379. ACM, 2005.

[HLZ+09] H. Huang, D. Li, H. Zhang, U Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics*, 28(5), 2009.

[IKN98] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1254–1259, 1998.

[JBBL02] Simon Julier, Yohan Baillot, Dennis Brown, and Marco Lanzagorta. Information filtering for mobile augmented reality. *IEEE Computer Graphics and Applications*, 22(5):12–15, 2002.

[JCW11] S. Jeschke, D. Cline, and P. Wonka. Estimating color and texture parameters for vector graphics. In *Computer Graphics Forum*, volume 30, pages 523–532. Wiley Online Library, 2011.

[JDA07] Tilke Judd, Frédo Durand, and Edward H. Adelson. Apparent ridges for line drawing. *ACM Transactions on Graphics*, 26(3):19, 2007.

[JEDT09] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *IEEE International Conf. on Comput. Vision (ICCV)*, 2009.

[JT08] J. Jia and C.K. Tang. Image stitching using structure deformation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):617–631, 2008.

[JTC09] N. Jiang, P. Tan, and L.F. Cheong. Symmetric architecture modeling with a single image. In *ACM Transactions on Graphics*, volume 28, page 113. ACM, 2009.

[KBH06] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of Eurographics Symposium on Geometry Processing 2006*, pages 43–52, 2006.

[KBM+08] Oliver Kutter, Christoph Bichlmeier, Sandro Michael, Ben Ockert, Ekkehard Euler, and Nassir Navab. Real-time volume rendering for high quality visualization in augmented reality. In *International workshop on Augmented environments for Medical Imaging including Augmented Reality in Computer-aided Surgery (AMI-ARCS)*, 2008.

[KCB+05] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Bi-layer segmentation of binocular stereo video. In *conf. Comput. Vision and Pattern Recognition (CVPR)*, pages 407–414. IEEE Computer Society, 2005.

[KCC06] H.W. Kang, C.K. Chui, and U.K. Chakraborty. A unified scheme for adaptive stroke-based rendering. *The Visual Computer*, 22(9):814–824, 2006.

[KD08] J.E. Kyprianidis and J. Döllner. Image abstraction by structure adaptive filtering. *Proceedings of EG UK Theory and Practice of Computer Graphics*, pages 51–58, 2008.

[KDMF03] Robert D. Kalnins, Philip L. Davidson, Lee Markosian, and Adam Finkelstein. Coherent stylized silhouettes. *ACM Transactions on Graphics*, 22(3):856–861, 2003.

[KK11] J.E. Kyprianidis and H. Kang. Image and video abstraction by coherence-enhancing filtering. *Computer Graphics Forum*, 30(2):593–602, 2011.

[KKD09] J.E. Kyprianidis, H. Kang, and J. Döllner. Image and video abstraction by anisotropic kuwahara filtering. *Computer Graphics Forum*, 28(7):1955–1963, 2009.

[KL08] H. Kang and S. Lee. Shape-simplifying image abstraction. *Computer Graphics Forum*, 27(7):1773–1780, 2008.

[KLC07] H. Kang, S. Lee, and C.K. Chui. Coherent line drawing. In *Proceedings of symposium on Non-photorealistic animation and rendering*, pages 43–50, 2007.

[KLC09] Henry Kang, Seungyong Lee, and Charles K. Chui. Flow-based image abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):62–76, 2009.

[KMS07] D. Kalkofen, E. Mendez, and D. Schmalstieg. Interactive focus and context visualization for augmented reality. In *Proceedings of IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10. IEEE Computer Society, 2007.

[KMS09] Denis Kalkofen, Erick Mendez, and Dieter Schmalstieg. Comprehensible visualization for augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 15(2):193–204, 2009.

[KST08] Michael Kolomenkin, Ilan Shimshoni, and Ayellet Tal. Demarcating curves for shape illustration. *ACM Transactions on Graphics*, 27(5):1–9, 2008.

[KSW06] Jens Krüger, Jens Schneider, and Rudiger Westermann. Clearview: An interactive context preserving hotspot visualization technique. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):941–948, 2006.

[KTO04] Yoshinari Kameda, Taisuke Takemasa, and Yuichi Ohta. Outdoor see-through vision utilizing surveillance cameras. In *International symposium on Mixed and Augmented Reality (ISMAR)*, pages 151 – 160. IEEE Computer Society, 2004.

[LD08] Y. Luo and R. Duraiswami. Canny edge detection on nvidia cuda. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–8. Ieee, 2008.

[Lin96] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. In *Proceedings of Computer Vision and Pattern Recognition*, pages 465–470. IEEE Computer Society, 1996.

[LLY09] H. Lee, C.H. Lee, and K. Yoon. Motion based painterly rendering. In *Computer Graphics Forum*, volume 28, pages 1207–1215. Wiley Online Library, 2009.

[LMLH07] Yunjin Lee, Lee Markosian, Seungyong Lee, and John F. Hughes. Line drawings via abstracted shading. *ACM Transactions on Graphics*, 26(3):18, 2007.

[LRC+03] David Luebke, Martin Reddy, Johnathan Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner. *Level of Detail for 3D Graphics*. Morgan Kaufmann, 2003.

[LSG+03] Mark A Livingston, J. Edward Swan II, Joseph L Gabbard, Tobias H. Höllerer, Deborah Hix, Simon J Julier, Yohan Baillot, and Dennis Brown. Resolving multiple occluded layers in augmented reality. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 7–10. IEEE Computer Society, 2003.

[LSnZ+07] Tie Liu, Jian Sun, Nan ning Zheng, Xiaoou Tang, and Heung yeung Shum. Learning to detect a salient object. In *Proceedings of Computer Vision and Pattern Recognition*, pages 1–8. IEEE Computer Society, 2007.

[LZPW04] A. Levin, A. Zomet, S. Peleg, and Y. Weiss. Seamless image stitching in the gradient domain. *Computer Vision–ECCV 2004*, pages 377–389, 2004.

[Mag85] Jan R. Magnus. On differentiating eigenvalues and eigenvectors. *Econometric Theory*, 24(1):179–191, 1985.

[MDGD+10] Patrick Mullen, Fernando De Goes, Mathieu Desbrun, David Cohen-Steiner, and Pierre Alliez. Signing the unsigned: Robust surface reconstruction from raw pointsets. *Computer Graphics Forum*, 29(5):1733–1741, 2010.

[MG08] D. Mould and K. Grant. Stylized black and white images from photographs. In *Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, pages 49–58. ACM, 2008.

[MH80] D. Marr and E. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217, 1980.

[MHM+09] D. Mahajan, F.C. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur. Moving gradients: a path-based method for plausible image interpolation. In *ACM Transactions on Graphics*, volume 28, page 42. ACM, 2009.

[MKS06] Erick Mendez, Denis Kalkofen, and Dieter Schmalstieg. Interactive context-driven visualization tools for augmented reality. In *International symposium on Mixed and Augmented Reality (ISMAR)*, pages 209–218. IEEE Computer Society, 2006.

[MP08] J. McCann and N.S. Pollard. Real-time gradient-domain painting. *ACM Transactions on Graphics*, 27(3):93, 2008.

[MS09] Erick Mendez and Dieter Schmalstieg. Importance masks for revealing occluded objects in augmented reality. In *ACM Symp. Virtual Reality Software and Technology (VRST)*, pages 247–248, 2009.

[ND04] M. Nienhaus and J. Döllner. Blueprints: Illustrating architecture and technical parts using hardware-accelerated non-photorealistic rendering. In *Proceedings of Graphics Interface 2004*, pages 49–56. Canadian Human-Computer Communications Society, 2004.

[Nea04] A. Nealen. An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation. Technical report, 2004.

[NJLM06] A. Ni, K. Jeong, S. Lee, and L. Markosian. Multi-scale line drawings from 3d meshes. In *Proceedings of symposium on Interactive 3D graphics and games*, pages 14–17, 2006.

[OBBT07] Alexandrina Orzan, Adrien Bousseau, Pascal Barla, and Joëlle Thollot. Structure-preserving manipulation of photographs. In *Proceedings of international symposium on Non-photorealistic animation and rendering (NPAR)*, pages 103–110, 2007.

[OBS04] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Transactions on Graphics*, 3(23):609–612, 2004.

[OBW+08] A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin. Diffusion curves: a vector representation for smooth-shaded images. In *ACM Transactions on Graphics*, volume 27, page 92. ACM, 2008.

[OGG09] Cengiz Oztireli, Gael Guennebaud, and Markus Gross. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum*, 28(2):493–501, 2009.

[OH12] Peter O'Donovan and Aaron Hertzmann. Anipaint: Interactive painterly animation from video. *IEEE Transactions on Visualization and Computer Graphics*, 18:475–487, 2012.

[OR90] S. Osher and L.I. Rudin. Feature-oriented image enhancement using shock filters. *SIAM Journal on Numerical Analysis*, pages 919–940, 1990.

[Par08] Sylvain Paris. Edge-preserving smoothing and mean-shift segmentation of video streams. In *Proceedings of ECCV*, pages 460—-473. Springer Berlin / Heidelberg, 2008.

[PGB03] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *ACM Transactions on Graphics*, volume 22, pages 313–318. ACM, 2003.

[PP09] G. Papari and N. Petkov. Continuous glass patterns for painterly rendering. *IEEE Transactions on Image Processing*, 18(3):652–664, 2009.

[PPC07] G. Papari, N. Petkov, and P. Campisi. Artistic edge and corner enhancing smoothing. *IEEE Transactions on Image Processing*, 16(10):2449–2462, 2007.

[Pra87] Vaughan Pratt. Direct least-squares fitting of algebraic surfaces. In *Proceedings of SIGGRAPH*, pages 145–152, 1987.

[PSA+04] Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael Cohen, Hugues Hoppe, and Kentaro Toyama. Digital photography with flash and no-flash image pairs. *ACM Transactions on Graphics*, 23(3):664–672, 2004.

[RSI+08] Tobias Ritschel, Kaleigh Smith, Matthias Ihrke, Thorsten Grosch, Karol Myszkowski, and Hans-Peter Seidel. 3d unsharp masking for scene coherent enhancement. *ACM Transactions on Graphics*, 27(3), 2008.

[RTF+04] R. Raskar, K.H. Tan, R. Feris, J. Yu, and M. Turk. Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. In *ACM Transactions on Graphics*, volume 23, pages 679–688. ACM, 2004.

[SCDM10] C. Sandor, A. Cunningham, A. Dey, and V.V. Mattila. An augmented reality x-ray system based on visual saliency. In *Proceedings of IEEE International Symposium on Mixed and Augmented Reality*, pages 27–36. IEEE Computer Society, 2010.

[SD02] Anthony Santella and Doug DeCarlo. Abstracted painterly renderings using eye-tracking data. In *Proceedings of international symposium on Non-photorealistic animation and rendering*, pages 75–83, 2002.

[SF96] E.P. Simoncelli and H. Farid. Steerable wedge filters for local orientation analysis. *IEEE Transactions on Image Processing*, 5(9):1377–1382, 1996.

[SKSK09] R. Schmidt, A. Khan, K. Singh, and G. Kurtenbach. Analytic drawing of 3d scaffolds. In *ACM Transactions on Graphics*, volume 28, page 149. ACM, 2009.

[SMK+09] Gerhard Schall, Erick Mendez, Ernst Kruijff, Eduardo Veas, Sebastian Junghanns, Bernhard Reitinger, and Dieter Schmalstieg. Handheld augmented reality for underground infrastructure visualization. *Personal Ubiquitous Computer*, 13(4):281–291, 2009.

[SOS04] Chen Shen, James F. O'Brien, and Jonathan R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Transactions on Graphics*, 23(3):896–904, 2004.

[SSD09] K. Subr, C. Soler, and F. Durand. Edge-preserving multiscale image decomposition based on local extrema. *ACM Transactions on Graphics*, 28(5):147, 2009.

[TM98] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of IEEE International Conf. Comput. Vision (ICCV)*. IEEE Computer Society, 1998.

[VFSG06] Ivan Viola, Miquel Feixas, Mateu Sbert, and Meister Eduard Gröller. Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics*, 12:933–940, 2006.

[VKG05] Ivan Viola, Armin Kanitsar, and Meister Eduard Gröller. Importance-driven feature enhancement in volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):408–418, 2005.

[VVC+11] R. Vergne, D. Vanderhaeghe, J. Chen, P. Barla, X. Granier, and C. Schlick. Implicit brushes for stylized line-based rendering. In *Computer Graphics Forum*, volume 30, pages 513–522. Wiley Online Library, 2011.

[WCL+10] S. Wang, K. Cai, J. Lu, X. Liu, and E. Wu. Real-time coherent stylization for augmented reality. *The Visual Computer*, 26(6):445–455, 2010.

[Wei03] J. Weickert. Coherence-enhancing shock filters. *Pattern Recognition*, pages 1–8, 2003.

[Win78] S. Winograd. On computing the discrete fourier transform. *Mathematics of Computation*, 32(141):175–199, 1978.

[Win11] H. Winnemöller. Xdog: advanced image stylization with extended difference-of-gaussians. In *Proceedings of symposium on Non-Photorealistic Animation and Rendering*, pages 147–156, 2011.

[WOG06] Holger Winnemöller, Sven C. Olsen, and Bruce Gooch. Real-time video abstraction. *ACM Transactions on Graphics*, 25(3):1221–1226, 2006.

[Wys82] G. Styles W. Wyszecki. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. John Wiley and Sons Ltd., 1982.

[XK08] J. Xu and C.S. Kaplan. Artistic thresholding. In *Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, pages 39–47. ACM, 2008.

[Zek93] S. Zeki. *A Vision of the Brain*. Oxford Univ Press, 1993.

[ZHT07] E. Zhang, J. Hays, and G. Turk. Interactive tensor field design and visualization on surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):94–107, 2007.

[ZMJ+09] Hanli Zhao, Xiaoyang Mao, Xiaogang Jin, Jianbing Shen, Feifei Wei, and Jieqing Feng. Real-time saliency-aware video abstraction. *The Visual Computer*, 25:973–984, 2009.

[ZQC+09] Fan Zhong, Xueying Qin, Jiazhou Chen, Mingzhen Mo, and Qunsheng Peng. Real-time post-processing for online video segmentation. *Chinese Journal of Computers*, pages 261–267, 2009.

[ZQC+10] F. Zhong, X. Qin, J. Chen, W. Hua, and Q. Peng. Confidence-based color modeling for online video segmentation. *Computer Vision–ACCV 2009*, pages 697–706, 2010.

[ZWW+10] X. Zhang, Z. Wang, R. Wang, Z. Yang, W. Chen, and Q. Peng. Real-time foveation filtering using nonlinear mipmap interpolation. *The Visual Computer*, 26(6):923–932, 2010.

[ZZXZ09] Kun Zeng, Mingtian Zhao, Caiming Xiong, and Song-Chun Zhu. From image parsing to painterly rendering. *ACM Transactions on Graphics*, 29(1):2:1–2:11, 2009.