Deep convolutional networks for inverse problems in image and video restoration

Benjamin Naoto Chiche

université PARIS-SACLAY

Deep convolutional networks for inverse problems in image and video restoration

Réseaux de neurones convolutifs profonds pour problèmes inverses en restauration d'images et de vidéos

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°127 Astronomie et Astrophysique d'Île-de-France (AAIF)
Spécialité de doctorat: Astronomie et Astrophysique
Graduate School : Physique, Référent : Faculté des sciences d'Orsay

Thèse préparée en collaboration avec le DAP/AIM (Université Paris-Saclay, CEA, CNRS) et SAFRAN Electronics & Defense, sous la direction de **Jean-Luc STARCK**, directeur de recherche, la co-direction de **Joana FRONTERA-PONS**, ingénieure de recherche, et la co-supervision d'**Arnaud WOISELLE**, ingénieur Safran Electronics & Defense

Thèse soutenue à Paris-Saclay, le 24 octobre 2022, par

Benjamin Naoto CHICHE

Composition du jury

Pascal Larzabal Professeur des universités, ENS Paris-Saclay - Université Paris-Saclay	Président
Florence Tupin Professeure, Télécom Paris	Rapporteure & Examinatrice
Christine Guillemot Directrice de recherche, Inria Rennes – Bretagne Atlantique	Rapporteure & Examinatrice
Jean-Luc Starck Directeur de recherche, CEA - CosmoStat	Directeur de thèse

THESE DE DOCTORAT
NNT : 2022UPASP115

université PARIS-SACLAY

ÉCOLE DOCTORALE
Astronomie
et Astrophysique
d'Île-de-France (AAIF)

Titre : Réseaux de neurones convolutifs profonds pour problèmes inverses en restauration d'images et de vidéos

Mots clés : Vidéo, Apprentissage profond, Apprentissage automatique, Restauration

Résumé : La restauration d'images et de vidéos regroupe de nombreuses tâches—comme le débruitage, la déconvolution et la super-résolution, pour ne citer que quelques exemples—qui permettent des applications de grand intérêt dans divers domaines de la recherche et de l'industrie (par exemple, les industries de la santé, l'armée, la création, les jeux et la recherche en astrophysique). Tous les problèmes de restauration sont modélisés dans le cadre mathématique des problèmes inverses, dans lequel les modèles directs spécifient les dégradations reliant les données corrompues observées aux données originales. Ces problèmes sont classiquement résolus sur la base de régularisations choisies à la main pour atténuer leur caractère mal posé et d'algorithmes itératifs qui minimisent les sommes de termes d'attache aux données et de régulariation. L'apprentissage profond et les réseaux neuronaux convolutifs (CNNs) ont récemment augmenté de manière significative les performances de restauration d'images et de vidéos. Ces réseaux peuvent notamment apprendre l' *a priori* sur l'image ou la vidéo à reconstruire à partir de données, *i.e.*, de paires de données dégradées et originales. Le modèle direct est utilisé afin de générer ces paires dans ce cadre d'apprentissage profond. Même si la régularisation apprise permet généralement d'obtenir de meilleures performances qu'une régularisation manuelle et que les CNNs sont plus rapides que les algorithmes itératifs (donc plus adaptés aux applications pratiques), les CNNs sont utilisés comme des boîtes noires et manquent d'interprétabilité. De plus, ils manquent également de flexibilité dans l'utilisation de la connaissance du modèle direct, contrairement à la résolution classique de problèmes inverses. Dans certaines situations où le modèle direct est simple et bien caractérisé, les méthodes classiques peuvent encore être plus performantes que les méthodes basées sur l'apprentissage profond. Certaines approches plus récentes sont *hybrides*, combinant les avantages des deux méthodes de manière complémentaire. Certaines d'elles permettent de concevoir, par exemple, un CNN unique et interprétable qui peut gérer de manière flexible les connaissances à priori des dégradations.

Ce travail étudie les architectures de réseaux de neurones pour résoudre les problèmes de restauration d'images et de vidéos. Premièrement, nous expliquons les principes des méthodes de restauration d'images et de vidéos classiques, puis basées sur l'apprentissage profond, puis hybrides. Ensuite, nous nous concentrons sur le problème inverse de la super-résolution vidéo : nous passons en revue sa résolution traditionnelle et sa résolution dans l'état de l'art basée sur l'apprentissage profond. Comme première contribution, nous proposons un réseau de super-résolution vidéo hybride qui combine les avantages de la résolution classique avec la puissance de représentation des CNNs. Comme deuxième contribution, nous proposons un réseau de super-résolution vidéo récurrent adapté à la super résolution de longues vidéos dans lesquelles certaines parties de la scène bougent à peine (ce type de vidéo peut être rencontré dans des applications telles que la vidéosurveillance) et introduisons une nouvelle base de données de test de telles vidéos. En effet, nous montrons que les réseaux de super-résolution vidéo récurrents existants présentent des instabilités sur ces vidéos. Enfin, nous nous concentrons sur la déconvolution de séries temporelles d'images en radio-interférométrie, afin de permettre une meilleure détection des sources astronomiques transitoires. Ces sources, qui apparaissent et disparaissent au fil du temps, sont très intéressantes pour les astrophysiciens car elles sont associées à des phénomènes physiques de haute énergie. Comme troisième contribution, nous proposons deux architectures de réseaux de neurones qui peuvent faire de la modélisation spatiale et temporelle pour résoudre ce problème de déconvolution.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Astronomie et Astrophysique d'Île-de-France (AAIF)

Title: Deep convolutional networks for inverse problems in image and video restoration
Keywords: Video, Deep learning, Machine learning, Restoration

Abstract: Image and video restoration regroups numerous tasks—such as denoising, deconvolution, and super-resolution, to give a few examples—that enable applications that are of high interest in diverse research and industrial areas (*e.g.*, health, military, creative, gaming industries, and research in astrophysics). All restoration problems are modeled in the mathematical framework of inverse problems, in which forward models specify degradations connecting the observed corrupted data to the original data. These problems are classically solved using hand-crafted regularizations to mitigate their ill-posedness and iterative algorithms that minimize sums of data fidelity and regularization terms. Deep learning (DL) and Convolutional Neural Networks (CNNs) have recently significantly increased image and video restoration performance. These networks can learn the regularization from data, *i.e.*, pairs of degraded and original data. The forward model is used in order to generate these pairs in this DL framework. Even though the learned regularization generally enables better performance than a hand-crafted one and CNNs are faster than iterative algorithms (thus more suitable for practical applications), CNNs are used as black boxes and lack interpretability. Moreover, they also lack flexibility in using knowledge of the forward model, contrary to classical inverse problem-solving. In some situations where the forward model is simple and well characterized, classical methods can still perform better than DL-based ones. Some more recent approaches are *hybrid*, blending the advantages of both methods in a complementary way. Some of them enable to design, for instance, a single and interpretable CNN that can flexibly manage knowledge about degradations.

This work investigates neural network architectures to solve image and video restoration problems. First, we explain the principles of classical, DL-based, and hybrid image and video restoration methods. Second, we focus on the Video-Super-Resolution (VSR) inverse problem: we review its traditional solving and state-of-the-art solving based on DL. As our first contribution, we propose a hybrid VSR network that mixes the advantages of classical solving with the representation power of CNNs. As our second contribution, we propose a recurrent VSR network adapted for super-resolving long videos in which some parts of the scene barely move (this kind of video can be encountered in applications such as video surveillance) and introduce a new test dataset of such videos. We demonstrate that existing recurrent VSR networks present instabilities on such videos. Finally, we focus on the deconvolution of image time series in radio interferometry, to enable better detection of transient astronomical sources. They are sources that appear and disappear over time and are highly interesting for astrophysicists because they are associated with high-energy physical phenomena. As our third contribution, we propose two neural network architectures that can do spatial and temporal modeling to solve this deconvolution problem.

Remerciements

Je remercie Arnaud Woiselle pour son aide qui m'a été si précieuse, dans plusieurs domaines. Les nombreuses discussions que j'ai eues avec Arnaud tout au long de ma thèse m'ont permis d'approfondir mes connaissances théoriques et pratiques en apprentissage profond et en traitement d'images. De plus, Arnaud m'a grandement aidé à prendre correctement en main les outils fournis par l'entreprise. Enfin, Arnaud m'a fait aussi découvrir le sport qu'est l'escalade, et les sessions escalade que nous avons effectuées ensemble avec d'autres membres de l'entreprise font partie des bons souvenirs de cette préparation de thèse.

Je remercie Joana Frontera-Pons pour toute son aide, qui m'a permis d'approfondir mes connaissances en optimisation et à mieux rédiger des articles scientifiques. Joana m'a aussi aidé à prendre en main les outils fournis par le CEA, par exemple le calculateur Jean Zay de l'IDRIS. Enfin, j'ai beaucoup apprécié le travail avec Joana dont la bienveillance m'a permis de me consacrer sereinement à ma recherche.

Je remercie Jean-Luc Starck pour m'avoir accompagné dans l'intégration au sein du CEA. Là encore, ma bonne entente avec Jean-Luc m'a été psychologiquement très favorable tout au long de ma préparation. Chaque fois que j'ai été au CEA, j'ai beaucoup apprécié nos échanges, que ce soit sur le travail ou sur d'autres choses. À l'occasion d'un deuil familial qui m'a frappé pendant ma préparation de thèse, j'ai beaucoup apprécié la présence de Jean-Luc et d'autres membres du laboratoire et de l'entreprise à mes côtés, et je leur en suis très reconnaissant.

Je remercie Julien Girard pour m'avoir beaucoup aidé dans mes travaux concernant la radio interférométrie. Julien m'a transmis avec pédagogie sa passion pour la radio interférométrie, et cela m'a permis de découvrir cette discipline et d'élargir mes horizons. J'ai beaucoup apprécié de discuter avec Julien, aussi bien sur des sujets techniques que sur d'autres sujets qui ne concernaient pas seulement la radio interférométrie.

Je remercie Pascal Larzabal, Christine Guillemot et Florence Tupin pour avoir accepté d'être membres de mon jury de thèse. J'ai beaucoup apprécié l'intérêt que Pascal, Christine et Florence ont accordé à ma demande.

Je remercie mon supérieur hiérarchique en entreprise, François Guillot, qui m'a grandement aidé à intégrer l'entreprise et à accomplir les tâches administratives nécessaires. Chaque fois que j'avais besoin d'aide, François répondait présent. Les journées des doctorants qu'il a organisées à l'entreprise ont été des moments très enrichissants.

Je remercie Laurent Verstraete et Thierry Fouchet d'avoir répondu systématiquement et rapidement à toutes mes questions concernant les procédures administratives au sein de l'école doctorale.

Je voudrais remercier tous les membres du pôle traitement d'images de Safran Electronics & Defense et les personnes travaillant au 2e étage du bâtiment 709 du CEA. J'ai beaucoup apprécié les échanges que j'ai eus au sein de ces deux équipes dans une ambiance à la fois très conviviale et très studieuse.

Enfin, je voudrais remercier très chaleureusement mes amis et ma famille.

Acronyms

BPTT BackPropagation Through Time.

CNN Convolutional Neural Network.

DL Deep Learning.

DUF Dynamic Upsampling Filters.

EDVR Video Restoration framework with Enhanced Deformable convolutions.

FRVSR Frame-Recurrent Video Super-Resolution.

FRVSR-MD Frame-Recurrent Video Super-Resolution for Multiple Degradations.

GT Ground Truth.

HL Hard Lipschitz constraint.

HR High Resolution.

LR Low Resolution.

MD Multiple Degradations.

MRVSR Middle Recurrent Video Super-Resolution.

MSE Mean-Squared-Error.

NLRB Non-Local Residual Block.

NN Neural Network.

PFNL Progressive Fusion Video Super-Resolution.

PSF Point Spread Function.

PSNR Peak Signal-to-Noise Ratio.

RED REgularization by Denoising.

RLSP Recurrent Latent Space Propagation.

RSDN Recurrent Structure-Detail Network.

SFT Spatial Feature Transform.

SFTMD Spatial Feature Transform for Multiple Degradations.

SISR Single Image Super-Resolution.

SL Soft Lipschitz constraint.

SOTA State-Of-The-Art.

SRN Spectral Rank Normalization.

SRN-C Spectral Rank Normalization for Convolutional layer.

SSIM Structural Similarity Index Measure.

UVSR Unrolled Video Super-Resolution.

VSR Video Super-Resolution.

Contents

List of Figures

List of Tables

Introduction

Given an observed image (resp. video) presenting physical degradations—*e.g.*, blurs and/or noise—, the image (video) restoration consists in reconstructing its underlying, original, and clean image (video). This task is important in many industrial applications that involve image (video) acquisitions. Indeed, these acquisitions involve sensors that always introduce additional physical corruptions when acquiring data. This way, image (video) restoration applications exist in, *e.g.*:

- Medical industry: computed tomography (CT) and Magnetic Resonance Imaging (MRI) reconstructions restore latent images from subsampled Fourier measurements.
- Astrophysics: for instance, a radio interferometer deployed to observe stars in the sky corrupts the sky image with its Point Spread Function (PSF) and noise. Denoising and inverting the effect of this PSF, *i.e.*, deconvolution, is particularly interesting to astrophysicists.
- Military industry: in modern warfare, the soldier is enhanced with optronic sensors feeding him with real-time video fluxes that he should observe and analyze continuously. His capabilities will be directly related to the quality of these images, which should be enhanced to optimize the system's range or to minimize the cognitive effort associated with watching these videos over a long period.
- Creative industry: recently, the resolution of various displays has dramatically increased, from high definition television HDTV (1920×1080), which currently dominates the market, to ultra high definition television UHDTV (4K or even 8k). In this context, there is a great need for Video Super-Resolution, which consists in converting Low Resolution (LR), low-quality videos into high-resolution, noise-free videos that can be pleasantly viewed on these High Resolution (HR) devices.
- Gaming industry: VSR allows to display HR details based on LR images that consume less data.

Mathematically speaking, image (video) restoration problems belong to the class of inverse problems. The starting point of an image (video) restoration problem is the forward image (video) formation model, which explicitly models physical degradations and connects observed data to the unknown and underlying data one wants to estimate. Classical restoration methods directly derive a minimization problem from this forward model. The function to be minimized is a sum of two terms. The first term is a data fidelity term that ensures the fidelity of the solution (*i.e.*, restored data) to the forward model. The second term, a regularization term, is needed to make the solution robust to noise and unique. The coefficient associated with this term, called the regularization hyperparameter, controls the trade-off between the two terms. The minimization is mostly performed using an iterative algorithm coming from convex optimization theory, such as gradient descent. While being mathematically grounded, classical methods present some drawbacks. First, iterative algorithms are generally slow and not suitable for real-time applications. Second, one has

to carefully design the regularizer and the regularization hyperparameter, which requires advanced knowledge about the inverse problem in question and can be time-consuming. More recent restoration methods based on Deep Learning (DL) can overcome these drawbacks. A DL-based reconstruction technique involves a Convolutional Neural Network (CNN) that can learn the function that associates the degraded image (video) to the corresponding Ground Truth (GT) image (video). This learning relies on degraded and GT image (video) pairs that can be simulated based on the forward model. Implementing this CNN can benefit from efficient GPU-based architectures, increasing their suitability for real-time applications. Moreover, the CNN learns the regularization from data, suppressing the need for carefully hand-crafting a regularizer and a regularization parameter. However, DL-based methods also present some drawbacks. As an example, contrary to a classical method, a CNN that only takes the degraded data as input lacks flexibility in dealing with heterogenous degradations. In other words, it cannot perform well when degradations' parameters vary. Moreover, a CNN is used as a black box and thus lacks interpretability. Given these advantages and disadvantages of classical and DL-based restoration methods, more recent, *hybrid* approaches combine and can benefit from both paradigms in a complementary way. For example, some of them allow designing a restoration CNN that can manage multiple degradations.

As part of the first part of this thesis which presents the context of recent trends in image and video restoration, the first chapter of this thesis presents a general overview of how different restoration methods exploit forward models of inverse problems to complete image (video) restoration tasks. We also expose how one can evaluate their performance.

Once the context of image and video restoration has been presented, the second part of this thesis tackles our contributions regarding some video restoration tasks. More specifically, the second chapter of this thesis focuses on the VSR problem. First, we present how classical and State-Of-The-Art (SOTA) DL-based methods solve this problem. Then, as our **first contribution**, we propose a new VSR neural network based on deep unrolling. This technique derives CNN architectures inspired by iterative algorithms used in classical restoration methods. Thus, it is a hybrid approach. We measure and discuss the performance of our newly introduced network in some experimental settings. Then, as our **second contribution**, we show for the first time that SOTA recurrent VSR networks diverge on long sequences with low motion (*i.e.*, videos in which some parts of the scene barely move). We propose a solution to this instability issue in the form of a recurrent VSR network that is more adapted to tackle long sequences with low motion. We propose a new test dataset of low-motion sequences to show its superior performance compared to existing SOTA recurrent VSR networks.

In the third chapter, as our **third contribution**, we design Neural Networks (NNs) to reconstruct time series of radio interferometry images to help detect transient astronomical sources. These sources appear and disappear over time and are particularly interesting for astrophysicists as they are associated with high-energy physical phenomena such as pulsars, rotating radio transients (RRATs), solar-system magnetized objects, and "fast radio bursts" (FRB). We formulate this search as a deconvolution inverse problem of image time series. Indeed, the radio interferometry images obtained via aperture synthesis imaging are corrupted by the PSF of the radio interferometer and additive noise, and the morphology of

this PSF varies in time because the sky rotates over the instrument during the observation. We propose two new NNs to manage multiple degradations and do both spatial and temporal modelings to solve this inverse problem. We show their superior performance on our simulated data over CLEAN, the most used classical algorithm in the radio interferometry community.

We conclude this manuscript with a summary of the results and a discussion about some perspectives.

Part I

Context

Chapter 1

Solving inverse problems of image and video restoration

In image/video restoration, we are given an observed image/video. This observation has been corrupted by physical degradations. The goal is to recover an image/video that is the nearest to its underlying original image/video, free of these degradations. An image/video restoration problem belongs to the class of inverse problems. The physical degradations are specified by the forward model of the inverse problem. The forward model of an image

restoration problem has the following form [100]:

$$y = \mathcal{D}_\sigma(Ax) \tag{1.1}$$

where:

- $y \in \mathcal{R}^N$ is the observed data, corresponding to an image of size $N = N_1 \times N_2$ arranged in a lexicographic order;
- $x \in \mathcal{R}^M$ is the underlying data, corresponding to an image of size $M = M_1 \times M_2$ arranged in a lexicographic order;
- $A \in \mathcal{R}^{N \times M}$ is the matrix defining the linear degradation operator;
- $\mathcal{D}_\sigma : \mathcal{R}^N \mapsto \mathcal{R}^N$ defines other degradations such as nonlinear ones or the effect of the noise. This operator is parameterized by σ.

In case of a video restoration problem, we need to account for the additional time dimension. The forward model has the following form [77].

$$y_j = \mathcal{D}_{\sigma_j}(A_j F_{u_{t \to j}} x_t) \qquad t - J \leq j \leq t + J \tag{1.2}$$

where:

- $2J + 1$ is the total length of the observed video;
- the indeces j and t indicate time steps;
- y_j represents the observed video frame at time step j;
- x_t represents a reference video frame at time step t;
- $F_{u_{t \to j}}$ denotes the warping operator with regard to optical flow $u_{t \to j}$. It models motion between two clean frames at time steps j and t.

The goal is to find a function F that inverts the forward model, using or not the knowledge about degradations. In image restoration, this function verifies the following equation:

$$\hat{x} = F(y, A, \sigma) \tag{1.3}$$

In case of a video restoration problem:

$$\hat{x}_t = F(y_{t-J}, ..., y_{t+J}, A_{t-J}, ..., A_{t+J}, \sigma_{t-J}, ..., \sigma_{t+J}) \tag{1.4}$$

Video restoration differs from image restoration because the fusion of several degraded frames produces a clean, underlying image. Moreover, it is important that reconstructed

frames with successive time steps $\hat{x}_{t-J}, ..., \hat{x}_t, ..., \hat{x}_{t+J}$ are temporally consistent between them. This temporal coherence ensures a minimization of flickering artifacts when observing the restored video.

In this thesis, we suppose the degradation operators and the noise levels (A and σ in the image restoration case; $\{A_j\}_{t-J\leq j\leq t+J}$ and $\{\sigma_j\}_{t-J\leq j\leq t+J}$ in the video reconstruction case) are known. Therefore, we do not need to estimate them. In other words, we deal with *non-blind* scenarios, unless otherwise specified.

1.1 Classical methods

1.1.1 Principles

For many image restoration problems, the forward model in Eq. (1.1) reduces to the linear additive noise model:

$$y = Ax + n \tag{1.5}$$

where n is a vector of noise. In most cases, it can be modeled as a zero-mean Gaussian noise with variance σ^2.

The form of the matrix A is specific to the inverse problem. In the case of the image denoising problem, A is reduced to the identity matrix. In the case of image deblurring, A is a square matrix that models the blur. This blur can stem from motion between the scene and the camera, the defocus of an optical imaging system, lens imperfections, and atmospheric turbulences. If the blur is spatially invariant, the product Ax is equivalent to a convolution of x with the Point Spread Function (PSF) of the sensor. In the case of image super-resolution, A models a blurring followed by a low-resolution acquisition and can be composed into a square matrix modeling the blur and a "flat" matrix modeling the down-sampling effect.

If A is invertible, a naive solution consists in applying the inverse of the linear degradation to the observation [100]:

$$\hat{x} = A^{-1}(Ax + n) = x + A^{-1}n \tag{1.6}$$

If the linear degradation is a convolution, A is a block-circulant matrix and may be diagonalized by the 2D Discrete Fourier Transform (DFT) matrix, which drastically reduces the computational cost of the inversion when the dimensions of A are powers of 2 thanks to the fast Fourier transform algorithm. However, if the matrix A is ill-conditioned (whether it is block-circulant or not), the inverse filtered noise $A^{-1}n$ may become very large so that its effect becomes important. Thus, the inverse filtering amplifies the noise leading to an irregular image. In practice, the inverse filtering is not robust even when the noise is light, but it works perfectly when no noise degrades the image. Fig. 1.2 illustrates these statements.

If A is not invertible, one can compute its pseudo inverse, which allows to compute a solution to the following least squares problem:

$$\hat{x} = \arg\min_x ||y - Ax||_2^2 \tag{1.7}$$

and there are two possible scenarios:

- if A has a rank $r = N < M$, we have an under-determined problem. This is for example the case for an image-super resolution problem. In this case, the solution is not unique and belongs to an affine space in the direction of $ker(A)$. The solution with minimum norm is most often selected: $\hat{x} = A^T(AA^T)^{-1}y$ and $A^T(AA^T)$ is called pseudo inverse of A.
- if A has a rank $r = M < N$, we have an over-determined problem. There is no exact solution if $y \notin Im(A)$. The least squares problem, however, has a unique solution given by $\hat{x} = (A^TA)^{-1}A^Ty$. The pseudo-inverse of A is therefore $(A^TA)^{-1}A^T$.

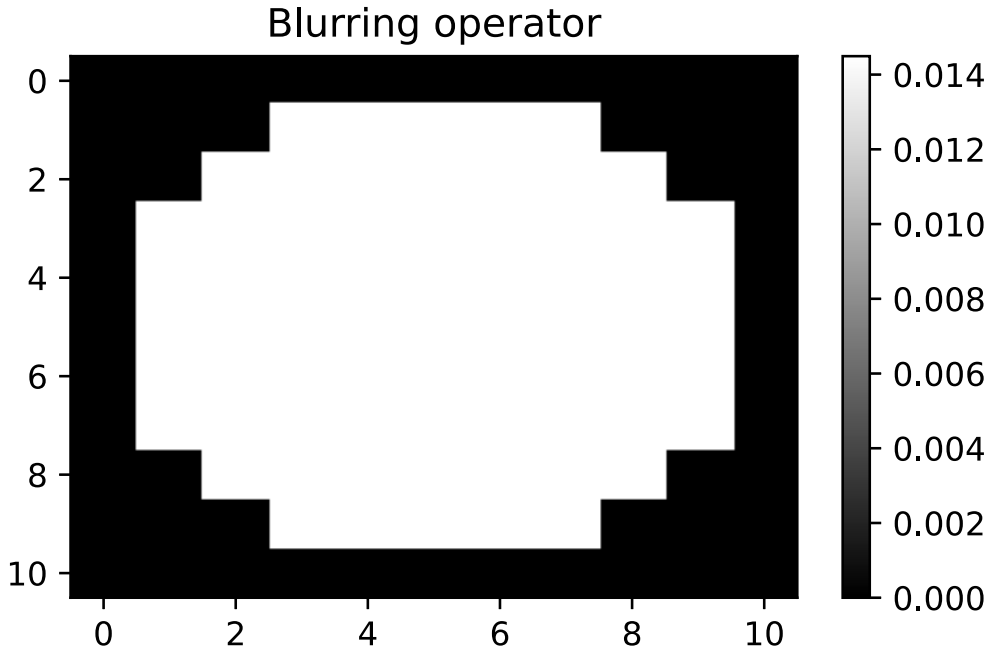


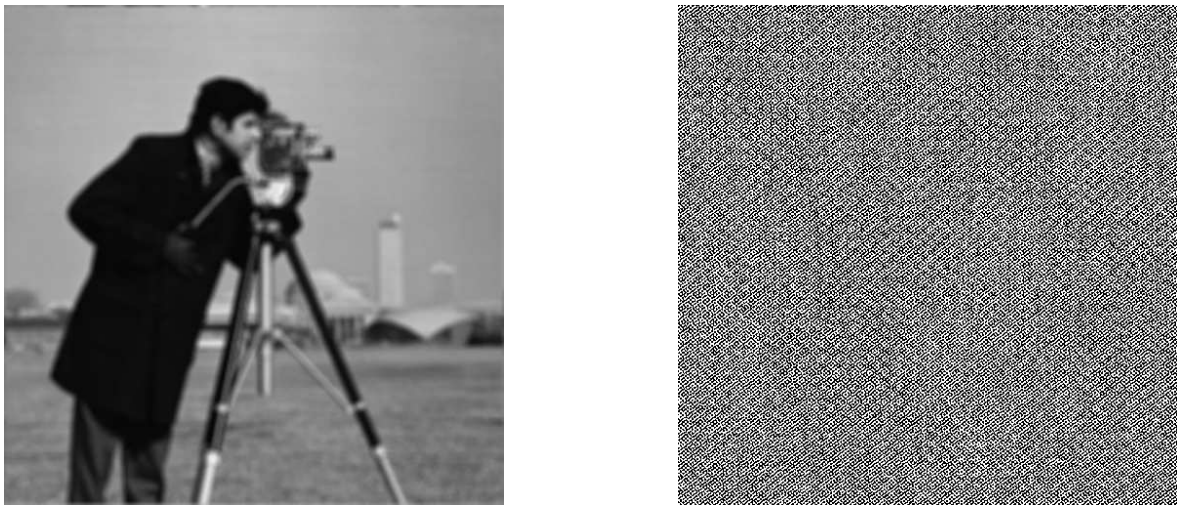
Figure 1.1: A blur kernel.

In both cases, the pseudo inverse filter often leads to irregular solutions, similar to the inverse filter. Therefore, whatever the dimensions and the rank of A are, there is a need for 1) stabilizing the solution, *i.e.*, making it robust to noise, and 2) guaranteeing its uniqueness. This need motivates the following alternative problem formulation:

$$\hat{x} = \arg\min_x ||y - Ax||_2^2 + \lambda r(x) \tag{1.8}$$

(a) $\sigma = 0$.

(b) $\sigma = 2.5 \cdot 10^{-3}$.



(c) $\sigma = 1$.

Figure 1.2: Deconvolution based on inverse filtering at different noise levels σ. The noise is white and Gaussian. The blur kernel described in Fig. 1.1 has been used. Left: degraded image. Right: deconvolved image. Even if the noise is imperceptible, the inverse filtering amplifies the noise.

(a) $\lambda = 0,025$

(b) $\lambda = 0,1$

Figure 1.3: Deconvolution based on Tikhonov filtering, with $L = I$. λ denotes the regularization parameter. The noise is white, additive and Gaussian with standard deviation $\sigma = 1$. The blur kernel described in Fig. 1.1 has been used.

Classical (also called *variational*) restoration methods solve this problem. $r : \mathbb{R}^M \mapsto \mathbb{R}^+$ denotes the regularization term, and $\lambda > 0$ is the regularization parameter that adjusts the trade-off between the data fidelity and the degree of regularity. r reflects our prior knowledge about the solution space, enforcing some constraints on the solution. Our prior knowledge should guide the tuning of r, which should describe as best possible natural image statistics. Therefore, r is also called *prior* or *prior term*. One of the most popular and simple regularizations is the Tikhonov regularization [121], where $r = ||Lx||_2^2$. L can be the identity matrix I or a highpass operator such as derivative or Laplacian. This regularization involves a closed form solution the inverse problem: $\hat{x} = (A^T A + \lambda L^T L)^{-1} A^T y$ which makes use of the Tikhonov linear filter $(A^T A + \lambda L^T L)^{-1} A^T$. When $L = I$, high-energy solutions are penalized. The Wiener deconvolution [142] indeed corresponds to a particular case of the Tikhonov filtering. The regularization parameter λ is a hyperparameter of the Tikhonov filtering. Figs. 1.3, 1.4b and 1.4c show that this filtering produces stable solutions even in the presence of noise. However, as illustrated by Figs. 1.3a and 1.4b, the filtering generates colored noise (*i.e.*, the noise with a non-flat power spectrum). As shown by Figs. 1.3b and 1.4c, this noise can be reduced by increasing the regularization parameter λ, but this also attenuates the high frequencies, *i.e.*, overly smooths edges.

The mixed performance of the Tikhonov regularization mostly stems from the fact that this regularization does not correctly describe natural image statistics. Indeed, this regularization imposes equivalent Gaussian assumption for both noise and image gradients [142, 145], which is mostly not respected. Therefore, studies in the literature design more sophisticated r to better capture natural image statistics and solve the problem thanks to iterative algorithm-based convex optimization. The Total Variation (TV) regularization [107] is one of the most popular and successful regularizations. It penalizes the total amount of change in the image as measured by the norm of the magnitude of the gradient, *i.e.*, $r(x) = ||\nabla x||_1$ where ∇ is the gradient operator. Consequently, it encourages solutions to contain uniform

(a) Degraded

(b) Tikhonov ($L = I$ and $\lambda = 0.025$)

(c) Tikhonov ($L = I$ and $\lambda = 0.1$)

(d) Tikhonov ($L = I$ and $\lambda = 0.2$)

(e) TV ($\lambda = 0.2$)

(f) TV ($\lambda = 4/3$)

Figure 1.4: Deconvolution based on Tikhonov and TV regularizations. The noise is white, additive and Gaussian with standard deviation $\sigma = 5$. λ denotes the regularization parameter. The blur kernel described in Fig. 1.1 has been used. The TV regularization is based on the Split Bregman algorithm [44, 47].

regions. Moreover, it tends to preserve edges in the reconstruction, as it does not severely penalize steep local gradients. Figs. 1.4e and 1.4f show deconvolutions based on the TV regularization. They make use of the Split Bregman algorithm [44, 47]. Fig 1.4f shows that a larger value for λ makes the regularization heavier and leads to an image with flatter regions. One can expect that TV regularization suffers when trying to reconstruct textured regions.

Below summarizes regularizations and iterative algorithms that are used in some studies in the literature:

- Authors of [3] use a non-smooth regularizer, which allows both wavelet-based or total-variation regularizations. They resort to the half-quadratic splitting algorithm for their optimization.
- The study [22] solves some image restoration problems based on the total variation regularization and a first-order primal-dual algorithm.
- The work [95] uses the total variation regularization and the alternating direction method of multipliers (ADMM) algorithm [16].
- Authors of [147] proposed to learn a regularizer as a sparse dictionary and use the back-projection algorithm [61].

As classical methods directly derive a minimization problem from the forward model and explicitly model the regularization term, they are also called *model-based* methods.

The examples shown by Figs 1.3 and 1.4 illustrate the importance of having a good image prior. This task is not easy, requiring expert knowledge about the inverse problem. Moreover, the choice of the image prior depends on the types of images encountered in the inverse problem. In other words, certain types of regularization work efficiently for some particular kinds of images but are not always suitable for more general images. For instance, maximum entropy regularizations that produce sharp reconstructions of point objects are adopted to reconstruct star fields in astronomical images [15]. Moreover, the previous examples also illustrate the need for carefully hand-tuning the regularization hyperparameter in classical image restoration tasks. This tuning also requires expert knowledge and time-consuming experimental trials.

1.1.2 Bayesian interpretation

There is a Bayesian interpretation of the problem in Eq. (1.8). One can see x and y as realizations of random vectors X and Y. x can be estimated based on the Maximum A Posteriori (MAP) strategy: the estimate of x should maximize the posterior probability distribution $\mu_{X|Y=y}$. Based on Bayes theorem:

$$\begin{aligned} \hat{x} \in \arg\max_x \mu_{X|Y=y}(x) &\iff \hat{x} \in \arg\max_x \frac{\mu_{y|X=x}(y)\mu_X(x)}{\mu_Y(y)} \\ &\iff \hat{x} \in \arg\max_x \mu_{y|X=x}(y)\mu_X(x) \end{aligned} \tag{1.9}$$

where $\mu_{Y|X=x}(y)$ is the *likelihood* function, $\mu_X(x)$ is the *prior distribution* and $\mu_Y(y)$ is the *marginal distribution*, assumed to be nonzero. By applying the logarithm function:

$$\text{Eq. (1.9)} \iff \hat{x} \in \arg\min_x -\log \mu_{Y|X=x}(y) - \log \mu_X(x) \tag{1.10}$$

The first term is related to data fidelity, and the second indicates the prior term. when n is a zero-mean Gaussian noise with variance σ^2, we have:

$$\mu_{Y|X=x}(y) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp -\frac{||Ax-y||_2^2}{2\sigma^2} \tag{1.11}$$

Therefore,

$$-\log \mu_{Y|X=x}(y) \propto \frac{1}{2\sigma^2}||Ax-y||_2^2 \tag{1.12}$$

By letting $-\log \mu_X(x) = \frac{1}{2\sigma^2}\lambda r(x) = \lambda' r(x)$, Eqs. (1.8) and (1.10) becomes equivalent, and we obtain an alternative problem formulation:

$$\hat{x} = \arg\min_x \frac{1}{2\sigma^2}||y-Ax||_2^2 + \lambda' r(x) \tag{1.13}$$

1.2 Deep learning methods

Deep learning (DL) and convolutional neural networks (CNNs) have recently enabled significant progress in image and video restoration performance. In contrast to classical methods, the idea of DL-based image and video reconstruction lies in the following: the inversion of the degradation, the prior and the regularization hyperparameter are learned from a dataset of natural images. Therefore, there is no more hand-crafted regularization. This approach presents the following advantages. First, the learned prior better captures statistics of natural images than the hand-crafted one. Second, one does not need to try to search for a good hand-crafted prior and an appropriate regularization hyperparameter. Instead, the DL-based approach requires effort into dataset collection and neural network training.

1.2.1 Supervised learning

In DL-based image/video restoration, Convolutional Neural Networks (CNNs) take a degraded image/video as input and output an estimate of its underlying original data. It is a standard regression problem where network features and channels contribute to the final output.

CNN

A CNN is a feedforward neural network composed of d 2D convolutional layers and interspaced nonlinear activation functions. Most of the time, these functions are the Rectifier Linear Unit (ReLU) function. A convolutional layer applies filters to the input image or feature maps. Each filtering produces an output feature map. Each filter is a convolution mask with weights $w_1, ..., w_K$, where $K = k^2 \cdot \text{nb\_channels\_in}$ with k being the kernel size and nb_channels_in is the number of input feature maps. Each filtering is accompanied by an addition of the bias term w_0. The size of the filter slides over the two spatial dimensions is called stride size.

A convolution operation is local, *i.e.*, extracts interactions between the central pixel and its neighboring pixels across input channels. Successive convolutional layers allow to extract interaction between pixels farther and farther in spatial locations. They are contained in the receptive field of CNN. If all convolutional layers have the stride of size 1, the size of the receptive field can be computed based on the following formula:

$$\text{receptivefield} = d \cdot (k - 1) + 1 \tag{1.14}$$

After a convolutional layer, the activation layer applies a nonlinear function at each pixel of the feature maps. This nonlinearity is essential to provide the CNN with the capability of approximating a function that is not necessarily regular. Indeed, an image to be reconstructed is a non-regular function, as it can contain edges and structures. The ReLU activation layer applies the following function: $f : x \mapsto max(0, x)$. By using this activation function, the CNN approximates a function with a piecewise affine function. Authors of [164] showed that a CNN with this activation function can approximate any continuous function to an arbitrary accuracy when the depth of this network is large enough. This is indeed a variant of the universal approximation theorem for neural networks. The ReLU activation has replaced other activation functions such as tanh or sigmoid that had been traditionally used. The ReLU activation demonstrates better performance for the following reasons. First, contrary to tanh or sigmoid, as the ReLU activation is not saturating, it does less trigger the vanishing gradient problem when the number of layers grows. Thus, the first layer can more efficiently receive the errors coming from the last layers to tune all weights between layers. Second, the ReLU activation function can accelerate the training speed of deep neural networks compared to traditional activation functions because the derivative of ReLU is 1 for positive input. Since this is constant, deep neural networks do not need additional time to compute error terms during training.

All weights of a CNN constitute its parameters, which are optimized at training time. Searches for high-performing CNN architectures (structures and ordering of layers) for various computer vision tasks have been an active field of research in recent years.

Training data

In most settings, parameters of a restoration network are learned in a supervised manner from training data, constituted of pairs of degraded image/video and GT original image/video. Preparing a set of enough large clean, non-corrupted images/videos is not difficult in some scenarios. We can then prepare the training set by simulating their corresponding degraded versions based on the forward model of the inverse problem. Therefore, neural networks indirectly capture the forward model based on pairs of degraded and GT images/videos: these networks are *data-based*. The most popular public training sets for image restorations are:

- DIV2K [5];
- Flickr2K [127].

In video restoration, popular training datasets are:

- Vimeo-90K [146];
- REDS [91];
- the one that is used in [42, 110, 28];
- MM522 [150].

Loss functions

To summarize, a neural network is a function $\mathcal{N}_\theta$ which realizes the following mapping: $\hat{x} \mapsto \mathcal{N}_\theta(y)$, where θ denotes the set of parameters of the network. In the case of supervised learning, these parameters are optimized at training time in such a way that the *loss* function $\mathcal{L}(x, \hat{x})$ is minimized. This function quantifies the distance between $\hat{x}$ and x. This metric can be:

- the Mean-Squared-Error (MSE) [42, 93, 28], where $\mathcal{L}(x, \hat{x}) = \frac{1}{M}||\hat{x} - x||_2^2$. This loss is the average over all pixels of the L2 distance and is generally the default choice. A variant of the L2 loss, called Huber loss, has been used in studies such as [66]. The Peak Signal-to-Noise Ratio (PSNR), which is expressed in dB, is a popular metric that is related to the MSE by the following relation: $\mathrm{PSNR}(x, \hat{x}) = 10 \log_{10} \frac{\max_i(x_i)}{\mathrm{MSE}(x,\hat{x})}$.

- L1 loss: $\mathcal{L}(x, \hat{x}) = ||\hat{x} - x||_1$. This loss function has been used in [76]. This loss can be a better proxy regarding perceptual quality than the L2 loss (see below for a more detailed explanation).

- Charbonnier loss: this has been used in [134, 150, 71]. This is a differentiable variant of L1 loss: $\mathcal{L}(x, \hat{x}) = \sqrt{(\hat{x} - x)^2 + \epsilon^2}$. ϵ is set to a small value, typically 10^{-3}.

- Structural Similarity Index Measure (SSIM) [138]: it is a better measure of perceived visual quality than PSNR since it is based on how the human eye extracts structural information from an image. Studies such as [11, 31] use it as the loss function to train an image restoration network. However, the authors of [162] showed that this loss could cause brightness changes or shifts of colors due to its lack of sensitivity to uniform bias. Moreover, they also showed that this loss could provoke noisy artifacts around edges or splotchy artifacts on flat regions, depending on a hyperparameter. Finally, the authors of [72, 13] showed that it still fails to capture and accurately assess image quality with respect to the human visual system, as PSNR.

- Perceptual loss: studies such as [72, 13] stated that L2-based loss favors overly smooth reconstructions with poor perceptual quality. This tendency is because minimizing L2 encourages finding pixel-wise averages of plausible solutions. Fig. 1.5 illustrates this phenomenon. Therefore, the L2 loss does not encourage perceptually optimized restorations. Additionally, authors of [162] show that networks trained with L2 loss generate splotchy artifacts of flat regions. Given this observation, the authors of [72] propose to use the sum of adversarial loss and content loss. The former is the generative loss: the lower this loss is, the more similar to the natural image the reconstructed image is. This loss involves an auxiliary discriminator network in the Generative Adversarial Network (GAN) framework [49]. The latter is the L2 distance between the feature representations of the reconstructed image and the GT image. These representations are based on the ReLU activation layers of the pre-trained 19-layer VGG network [119]. Authors of [160] proposed another metric based on deep feature maps, called LPIPS.

  However, encouraging perceptually optimized reconstructions during training comes with some drawbacks. On the one hand, the GAN framework increases memory usage, as the discriminator network is jointly trained with the reconstructing network. On the other hand, regarding VGG and LPIPS losses, computing the distance between feature representations involves inferences of the pre-trained network, increasing the training time.

The standard L2 loss theoretically maximizes the L2-based metrics such as PSNR. However, loss functions other than L2 loss can sometimes enable better performance even in metrics such as PSNR. As an example, the authors of [76] decided to use L1 loss because it empirically enabled better convergence, thus better-performing networks. Authors of [162] observe similar results and provide an explanation of this. They showed that this is due to the smoothness and the local convexity properties of the two loss functions: with L2 loss, the optimization gets stuck more easily in a local minimum, while with L1 loss, it is easier to reach a better minimum.

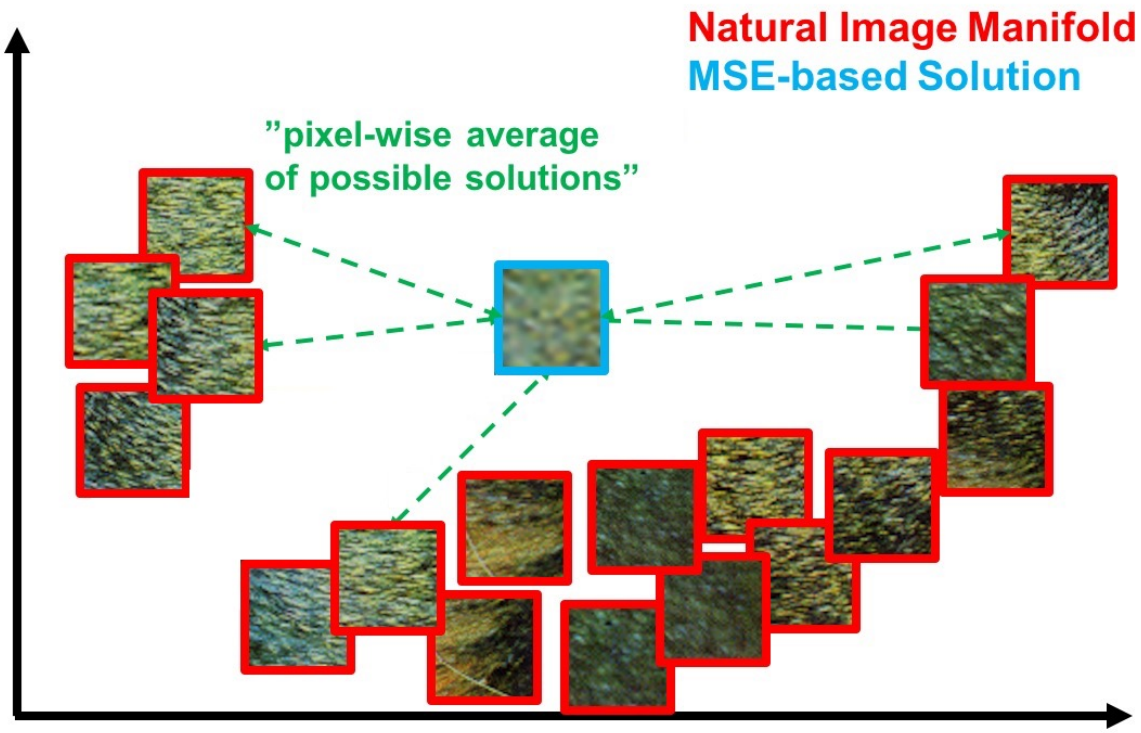


Figure 1.5: Illustration of patches from the natural image manifold (red) and restored patches obtained with the MSE train loss (blue). The MSE-based solution appears overly smooth due to the pixel-wise average of possible solutions in the pixel space. This illustration is adapted from [72].

Weight optimization

The neural network's parameter set $\theta = \{w_j\}$ is optimized to minimize the loss. This optimization is based on the gradient descent algorithm. At each iteration step i, the weight w_j is updated based on the following rule:

$$w_j^i = w_j^{i-1} - \eta \frac{\partial \mathcal{L}}{\partial w_j} \tag{1.15}$$

The updates should be terminated when all of these weights have converged. $\frac{\partial \mathcal{L}}{\partial w_j}$ is called the *gradient*. η is called the *learning rate*. It is decreased with increased iteration steps to help convergence. At each iteration step, $\frac{\partial \mathcal{L}}{\partial w_j}$ is estimated based on 1) a random set of examples (y, x) called mini-batch 2) the chain rule that allows backpropagation. The number of examples in the mini-batch is called batch size and noted B. A larger batch size allows smoother convergence by enabling a more accurate gradient estimate. However, a smaller batch size means faster convergence because the parameters are updated after each backpropagation.

One needs to keep in mind that there is no theoretical guarantee that the loss function minimized is convex. This means that the optimization can be stacked in a local minimum. To avoid this, one can augment the update with a momentum term. This term keeps the update process moving in the same direction by taking into account past gradients. This augmentation helps escape from local minima pits. One of the most popular momentum optimizations is the Adam optimization [70]. It keeps track of an exponentially decaying average of past gradients and squared gradients.

Hyperparameter tuning

To make a restoration model converge, one needs to tune the training hyperparameters properly. Among them are the batch size B and the learning rate η.

Similarly to the convolution operator, a convolutional image restoration network can manage input images of any size. Therefore, to save memory, the training uses small, cropped images with a particular crop size. However, a smaller crop size means more present border effects of convolutions. Moreover, this size should be larger than the receptive field of the network to leverage the network's full capacity. These considerations introduce a new training hyperparameter, the *crop size*. In the same way as batch size, a larger crop size enables smoother convergence.

Batch size is also limited by memory. Memory usage at training time is thus determined by the crop size and the batch size. A trade-off between them is therefore needed.

The learning rate should be as high as possible at the beginning in order to speed up the training. However, if it is too high, it can make the convergence harder or can cause instability. Generally, the learning rate should be low when the batch size is large. Therefore, these hyperparameters have to be jointly tuned.

The choice of the loss function impacts the magnitude of the gradient. This choice, therefore, influences the subsequent hyperparameter tuning.

In the case of a recurrent video restoration network [42, 110, 28, 93], the training involves BackPropagation Through Time (BPTT). This technique consists of unrolling through T past time steps the recurrent network and backpropagating over this unrolled network. This technique is highly demanding in terms of memory usage, which proportionally increases with T. Therefore, T should be jointly tuned with other hyperparameters impacting memory usage.

To conclude this part, one should tune these hyperparameters based on repeated experiments and trials. Tab. A.1 in the Appendix summarizes the training settings of SOTA VSR networks.

Training strategies

A good weight initialization can accelerate training and/or avoid local minimums. The Xavier initialization is the most used initialization scheme in image and video restoration [46].

Increasing training data variability helps restoration networks increase their generalization capability and avoid overfitting, *i.e.*, fitting overly complex functions to the training data (even though overfitting is generally not a problem in restoration tasks). Data augmentation enables increasing this variability. This technique applies a random composition of flipping and transposition to each training image/video crop. In video restoration, this variability can also be increased based on temporal augmentation[66]: this technique randomly samples video frames at different time steps to improve motion variability.

An image or video restoration problem is a standard regression problem, where network features and channels all contribute to the final output. Therefore, dropout [55], which induces a significant information loss within the network, is rarely used at training time.

Connection with MAP inference

Authors of [27] proposed a trainable nonlinear reaction diffusion (TNRD) model for image restoration. This model learns a modified fields of experts [106] image prior by unrolling a fixed number of gradient descent steps. Authors of [158] showed that an image restoration CNN is a generalization of one-step TNRD, and most of their parameters represent the image prior. Moreover, they empirically demonstrated that a single CNN could handle multiple scales super-resolution, image deblocking and image denoising.

1.2.2 Other learning strategies

Supervised learning requires having a collection of clean images/videos. In some applications, having such a collection is not straightforward. Other restoration methods do not

present this requirement.

Unsupervised learning

Authors of [118] proposed to train a small CNN for each image to be super-resolved. The input LR image is further blurred and downsampled several times in a hierarchical manner, accompanied by data augmentation. These re-degradations generate several training pairs that are used in order to fit the CNN. The trained CNN is then deployed to super-resolve the input LR image. Deep Image Prior (DIP) [128] is another unsupervised image restoration technique that fits a CNN to a degraded image. It is more detailed in Sec. 1.3.3.

The drawback of these unsupervised methods is that they require fitting a CNN for each input image. This requirement makes them slow and less attractive for practical applications.

Self-supervised learning

In their VSR problem, the authors of [96, 8] proposed two different self-supervised learning methods, which only necessitate a training dataset of LR video. Their common idea is to re-degrade the estimated HR frame based on the image formation model and compare it to the input LR frame to compute the loss. However, when applied naively, this approach results in trivial solutions. The work in [96] circumvents this problem by not including the reference frame in the input of the input encoding CNN. The study in [8] avoids the problem by re-degrading the input LR frames and giving them to the super-resolution network. The output is compared to the input LR frame to compute an additional auxiliary loss.

1.2.3 Common architectural blocks in DL

Global residual/skip connection

Authors of [69] and [158] respectively used this technique for their image super-resolution and denoising problem. This technique consists in learning to reconstruct the residual image $r = x - y$, instead of x. This residual learning makes sense because the degraded and original images are highly correlated and share the same information to a large extent. As an illustration, in the Single Image Super-Resolution (SISR) problem, an HR image can be decomposed into low-frequency information (corresponding to LR image) and high-frequency information (residual image or image details). Input and output images share the same low-frequency information. In problems such as denoising or SISR, if the network learns to reconstruct x instead of r, this model learns to both carry the input to the end layer and reconstruct residuals. By handing the input to the end, the network only needs to learn to reconstruct the residuals, enabling much faster training with even better accuracy.

Residual blocks

A residual block [54] is a set of a first convolution layer, followed by a ReLU activation layer that is in turn followed by another convolution layer. A residual connection links the input of this block with the output of the block. Such a design ensures a fluent gradient flow and can preserve the texture information over a deep network. The study [109] empirically found that residual blocks enable faster convergence than stacked convolution layers.

Pixel shuffling and unshuffling layers

Pixel shuffling and unshuffling operations [42] are also respectively called space-to-depth and depth-to-space [110, 117] transformations. The shuffling reduces the channel dimension C of a tensor t with factor r^2 and extends both spatial dimensions H and W with factor r. The unshuffling is the inverse of this operation. These operations are summarized below:

$$t^{LR} \in \mathbb{R}^{H \times W \times Z} \quad \xrightarrow{\times r} \quad t^{HR} \in \mathbb{R}^{rH \times rW \times Z/r^2} \tag{1.16}$$

$$t^{HR} \in \mathbb{R}^{H \times W \times Z} \quad \xrightarrow{/r} \quad t^{LR} \in \mathbb{R}^{H/r \times W/r \times r^2 Z} \tag{1.17}$$

Fig. 1.6 illustrates these operations. They are helpful in inverse problems such as SISR or VSR. Indeed, in these problems the linear degradation involves a downsampling operation which makes observed image y have reduced spatial resolutions compared to the corresponding underlying image x. We say in this case that y lives in the low-resolution (LR) space and x lives in the high-resolution (HR) space. Pixel shuffling and unshuffling allow the tensor to switch between these spaces.

An interesting feature of these operations is that it keeps local integrity. All pixels along the channel dimension in the LR space are rearranged in their corresponding local HR interpolation area, and vice versa.

In studies such as [42, 110, 117, 28], to gain computational efficiency, most of the convolution operations are done in the LR space and pixel shuffling is executed at the last stage to output an HR image. In [42, 110], to feedback recurrent information from the HR space, the pixel unshuffling operation is used.

Transposed convolution layer

Apart from pixel unshuffling, zero-filling [40] and interpolation, another operation can upsample the spatial dimensions of feature maps. Transposed convolution is such an operation. A transposed convolution layer broadcasts input feature maps via the kernel regrouping learnable parameters. This layer is used in studies like [110] to map feature maps in LR-space to HR-space. Transposed convolution is also called fractionally strided convolution, because the stride over the output is equivalent to the fractional stride over the input. For instance, a stride of 2 over the output is 1/2 stride over the input.

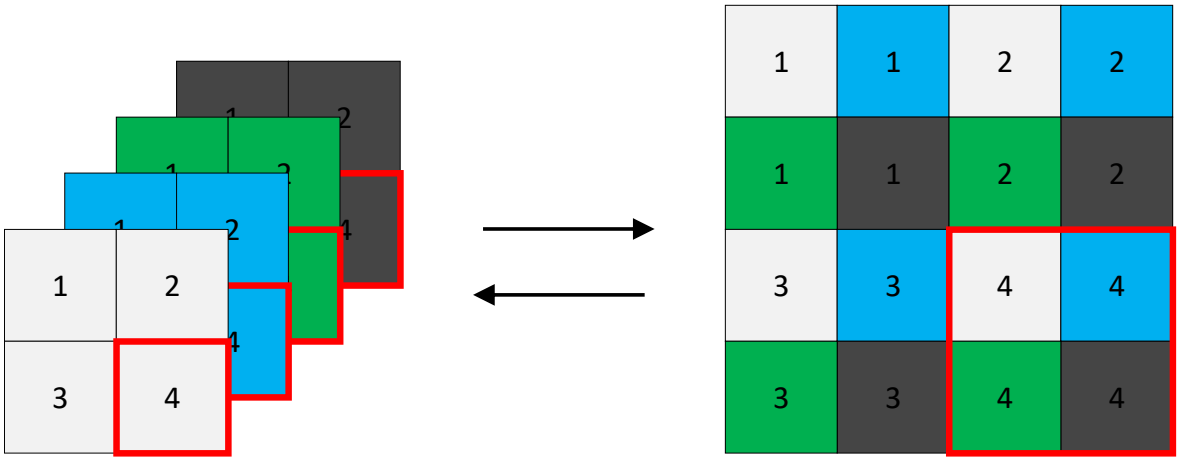


Figure 1.6: Illustration of pixel shuffling and unshuffling. Figure taken from [42].

Batch normalization

Batch normalization consists in:

- centering and scaling the input using the empirical mean and variance of the mini-batch containing the input;
- affine-transforming the centered and re-scaled input using learnable parameters;

at training time. At test time, the input undergoes the same operation but the centering and scaling operations use the empirical mean and variance over the whole train set.

This technique has been extensively used and showed that it could significantly increase performance in high-level computer vision tasks such as classification and object detection. However, the authors of [76] suggested not using this technique when dealing with image restoration problems. They stated that this is because batch normalization eliminates range flexibility from networks by normalizing the features. They experimentally verified that not using batch normalization layers significantly increases the performance. Moreover, not using batch normalization layers enables to reduce GPU memory usage significantly . Their baseline model without batch normalization saved around 40 % of memory usage during training, compared to SRResNet [72]. Therefore, under limited computational resources, they could use a larger model with better performance than models with batch normalization layers. For all of these reasons, we do not use batch normalization in our work.

Multi-scale architectures

Multi-scale architectures divide the input image features into several scales, process them independently and finally aggregate them in a coarse-to-fine manner. Features go from a scale to a lower one by a downsampling operation and to a higher one by an upsampling operation. Multi-scales analysis was also used in past studies before the DL era. In DL, these

architectures have been used for numerous tasks such as image segmentation (U-Net [105]), image restoration [156], deformable convolution-based feature alignment [134], temporal and spatial attention [134] and optical flow estimation [120].

Multi-scale architectures allow to capture long-term spatial interactions in a coarse manner at lower scale. At higher scale, these interactions are finer. At lower scale, they are coarse. As an example, these architectures can be useful to model large-motion [134, 120].

1.3 Blending classical and DL-based methods

Model-based approaches use the forward model to design mathematical tools that are directly related to the problem itself, and use the knowledge of the model to guide the solution. The main implications are:

- some guarantees (quality, convergence) and full explainability/interpretability of the results;
- a need to add regularization/prior terms;
- any deviation between the model and reality may result in a sub-optimal or completely unusable algorithm when applied to real data (*e.g.*, different blur shape);
- iterative algorithms may require many iterations to converge, making these approaches unusable in some practical cases due to the computational cost.

In the case of purely data-based methods, the forward model is used to generate data, *i.e.*, synthesize degraded image/video starting from the corresponding GT one. Still, the restoration process does not use a priori information from the model itself, only its inputs/outputs. The consequences are:

- a lack of directly given physical information about the degradations;
- no need to be able to inverse the forward model, which can be as complex as needed;
- there is no need for hand-crafted regularization terms, as the regularization is learned based on the data itself;
- the computational cost may be very high for some networks. But efficient architectures (GPU) can be used.

From the MAP point of view (Eq. (1.13)), classical methods produce a solution that can be formulated as [157]:

$$\hat{x} = F(y, A, \sigma; \theta) \tag{1.18}$$

where θ denotes additional parameters of the MAP inference. λ' can be indeed absorbed into σ. However, in DL-based inverse problem solving, most neural networks instead realize the mapping $\hat{x} = F(y, \theta)$ [145, 69, 42, 110, 134]. In these cases, θ, which denotes the parameters of a neural network, can be seen as the parameters of the MAP inference. These networks are, indeed, blind. Knowledge about degradations (A and σ) is only indirectly captured based on training data. This aspect can disadvantage DL-based approaches in a multiple degradation scenario, *i.e.*, when degradations or parameters of degradations change (are not homogeneous) over training and test sets, such blind models can perform poorly.

Recent studies have tried to breach the gap between the two paradigms, to benefit from both of them in a complementary way. In particular, this allows a restoration model to benefit from both the prior learning capability of neural networks and the flexibility of classical methods in non-blindly handling multiple degradations. Compared to a blind one, this non-blind model can explicitly incorporate the knowledge about the degradation and adapt to it, enabling better performance. From a realistic application-oriented point of view, having a blind model that only performs well in a unique and predefined degradation scenario is not viable. First, it is rare to encounter such an ideal scenario. Second, it is unrealistic to prepare several pre-trained models, each specialized for a unique degradation, to deal with multiple degradation scenarios. Third, having a model that can flexibly manage multiple degradations is interesting for practical applications. The user can give information about degradations as input parameters.

The following specifies some strategies to design such a model.

1.3.1 Giving physical knowledge at the network's input

One can give knowledge about A and σ directly to the network's input [157, 51] or indirectly by transforming the input degraded image based on physical priors [121, 145]. In the former, the network directly uses knowledge about degradations to operate transformations on feature maps, similarly to the attention mechanism [119, 57, 161]. In the latter, the input degraded image is first filtered based on a hand-crafted prior. The network learns the mapping between this prior and clean image.

The techniques Multiple Degradations (MD) [157] and Spatial Feature Transform for Multiple Degradations (SFTMD) [51] belong to the first family. The methods used in [121, 145] based on the Tikhonov filter belong to the second family. The following explains these techniques.

MD

Authors of [158] pointed out that CNN mainly models the prior information and empirically demonstrated that a single model could handle multiple scale super-resolution, image deblocking, and image denoising. In other words, they indicated that the parameters of the MAP inference mainly model the prior, and therefore, CNN can deal with multiple degradations via a single model. Given this, in the context of a single image super-resolution

problem with varying noise levels and blur kernels, the authors of [157] proposed to encode knowledge about degradations in degradation maps and give them to their model's input along with the degraded image. When super-resolving an image, the blur kernel used in the degradations is vectorized and projected on a Principal Components Analysis (PCA) basis. The PCA projection matrix is learned on 10000 randomly generated anisotropic Gaussian kernels. The PCA projected vector is concatenated with the noise level, and the whole vector is stretched to give degradation maps. These maps are concatenated with the degraded image and given to a feedforward super-resolving network. Degradation maps contain warping information, thus enable the network to have spatial transformation capability. Indeed, these maps can be seen as the output of a spatial transformer as in [64]. By anchoring the model with degradation maps, the non-blind model generalizes easily to unseen degradations and can control the trade-off between the data fidelity and regularization terms.

SFTMD

Authors of [51] proposed another way to give information about degradations and handle multiple degradations based on the Spatial Feature Transform (SFT) layer, inspired by [135]. The SFT layer provides an affine transformation for its input feature maps F_{in} conditioned on the knowledge about A and σ, *i.e.*, the degradation maps $F^{(A,\sigma)}$, which is computed similarly to MD [157]. The affine transformation involves scaling and shifting operations:

$$\text{SFT}(F_{in}, F^{(A,\sigma)}) = \gamma \odot F_{in} + \beta \tag{1.19}$$

where γ and β are estimated by convolutional layers and $\odot$ denotes the Hadamard product.

Transforming the network's input with regularized inversion

The work [121] proposed Tikhonet, with maps Tikhonov filtered input image to the corresponding estimated clean image. The study [145] proposed a network in which its first layer's weights are initialized with the separable Tikhonov filter. These weights are tuned based on training data, and this strategy gives better restoration than random initialization in their study. Similarly, the work [114] proposed to apply a regularized inverse filter and remove the resulting artifacts with a multi-layer perceptron. Authors of [65] first operate a filtered back projection and then use a U-Net [105] in their low-view CT reconstruction problem. These approaches involve single-step filtering followed by feedforward network propagation. Therefore, they are fast. However, their first step inversions involve handcrafted priors that mostly do not correctly describe natural image statistics and lead to an irregular solution corrupted by colored noise. Moreover, they control the weight of the regularization based on a hyperparameter that has to be manually tuned.

1.3.2 Iterative algorithms combined with deep-learning networks

Some recent approaches attempted to blend model-based and data-based methods to benefit from both. This blending allows designing a flexible technique that can manage multiple degradations while learning prior from data. The following details these methods.

Using iterative algorithms from convex optimization and replacing the proximity operator related to the prior by one or several DNNs

One can use an iterative algorithm from convex optimization and replace the proximity operator related to the prior with one or several DNNs. The following frameworks ease this:

- plug-and-play [130, 34, 165]: this approach first reformulates and solves the MAP problem using an iterative algorithm based on variable splitting. In the context of this splitting, each iteration involves alternated data and prior subproblems. The plug-and-play framework replaces the prior subproblem with any off-the-shelf Gaussian denoiser. Unlike traditional image restoration methods that employ hand-crafted image priors, it can implicitly define the plug-and-play prior by the denoiser. As expected, this denoiser can be a CNN. The study [159] plugs several denoising networks trained each for a different noise level to adapt for variation of the penalty parameter in its half quadratic splitting. In contrast, authors of [85] plug the same unique CNN denoiser at each step. The work [108] proves the convergence of plug-and-play methods under certain conditions;
- REgularization by Denoising (RED) [104, 103]: this approach consists in defining the explicit regularizer $r(x)$ as follows: $r(x) = \frac{1}{2}x^T(x - f(x))$ where f is an image denoiser. Under mild conditions on f, the gradient of the regularization term can be easily computed, equaling the denoising residual $x - f(x)$. Any inverse problem can be managed while calling the denoising engine iteratively.

This approach uses CNNs as denoisers in a mathematically defined framework of an iterative algorithm. Therefore, it presents high interpretability. However, it still suffers from the drawbacks of an iterative algorithm. First, it involves hand-designed hyperparameters to control the convergence and stability of the iterative algorithms. Second, as it requires numerous iteration steps to restore an image, it is slow.

Deep unrolling

Deep unrolling consists of unfolding the iterative loop of a classical iterative algorithm that solves the MAP problem (Eq. (1.13)) with a given number of iterations, replacing some operators in each unfolded step by CNNs and representing all operations as layers of a neural network. This network can then be trained and optimized as any other network, learning

from data while keeping the knowledge of the inverse problem in its internal structure. Based on this technique, several works unfolded the following algorithms:

- (proximal) gradient descent [27, 36, 81, 93];
- alternating directions method of multipliers [148];
- primal-dual methods [2];
- half-quadratic splitting [156];
- alternating minimization [4].

For different unrolled optimization methods, CNNs in each unfolded step play different roles. As an illustration, in proximal gradient settings, the learned CNN in each unrolled step is interpreted as a learned proximal operator. In contrast, in unfolded gradient descent network, the learned CNN in each step is interpreted as the gradient of the regularizer. These roles being clear and explicitly integrated in a classical algorithm, deep unfolded networks present higher interpretability than black box-like CNNs that take degraded data as input and output the corresponding restored ones.

Deep unfolding optimizes the parameters end-to-end by minimizing the loss function over a large training set. Therefore, on the one hand, the number of unrolled iteration steps is limited by the memory. On the other hand, compared to the approach that uses an iterative algorithm and replaces the proximity operator related to the prior by frozen DNN denoisers, the number of unrolled iteration can be fewer while still enabling better performance. Moreover, fewer iterations mean increased restoration speed. Finally, unrolling architectures usually need a smaller number of parameters than purely data-driven approaches, as they can leverage the knowledge about the model. Moreover, the CNNs in unrolled steps can share weights, enabling to further reduce number of parameters [156].

1.3.3 Deep image prior (DIP)

Authors of [128] states that for a surjective $g(\theta) = x$, Eq. (1.8) is equivalent to:

$$\hat{\theta} = \arg\min_{\theta} ||y - Ag(\theta)||_2^2 + R(g(\theta)) \tag{1.20}$$

One can define $g(\theta)$ as $f_\theta(z)$, where f is a CNN with parameters θ and z is a fixed input. This enables to replace the regularizer R with the implicit prior captured by the neural network, leading to the following formulation:

$$\hat{\theta} = \arg\min_{\theta} ||y - Af_\theta(z)||_2^2 \tag{1.21}$$

θ is optimized according to the problem based on gradient descent, starting from random initialization. z is filled with random noise and fixed. We highlight this is an unsupervised

approach, therefore not requiring pre-training based on pairs of degraded and original images. Instead, this approach fits a network for an image to restore.

Authors of [84] builds on DIP but adds an explicit prior, which enriches the overall regularization effect, leading to better-recovered images. This regularization is based on RED [104].

An obvious drawback of these approaches is that they are slow. They fit a CNN on a single image to be restored. It is clear that the role of CNN here is to define the image space of possible solutions. Therefore, these approaches present high interpretability.

1.4 Performance evaluation

Once designed and/or trained, image and video restoration methods need to be evaluated. This allows to compare different methods and choose the best one to deploy it on practical applications. These evaluations should be based on quantitative and qualitative considerations. Moreover, they should use a validation/test set to simulate a realistic application-oriented scenario.

1.4.1 Numerical evaluations

PSNR

The Peak Signal-to-Noise Ratio (PSNR) is the most common metric used in order to evaluate the fidelity of the reconstruction regarding the GT numerically. This metric is computed as follows:

$$\mathrm{PSNR}(x, \hat{x}) = 10 \log_{10} \frac{\max_i(x_i)}{\mathrm{MSE}(x, \hat{x})} \tag{1.22}$$

One can see that minimizing MSE also maximizes PSNR.

SSIM

The SSIM index[138] is a better measure of perceived visual quality than PSNR since it is based on how the human eye extracts structural information from an image. This index is computed on several windows of an image. The metric between two windows x and y of same size is:

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{1.23}$$

where:

- μ_x is the average of x;
- μ_y is the average of y;
- σ_x is the standard deviation of x;
- σ_y is the standard deviation of y;
- σ_{xy} is the covariance of x and y;
- $c_1 = (k_1\Delta)^2$ and $c_2 = (k_2\Delta)^2$ are two values that stabilize the division with small denominator. Δ is the dynamic range of pixel values. We typically set $k_1 = 0.01$ and $k_2 = 0.03$.

Perceptual metrics

MSE-based metrics like PSNR favor overly smooth solutions with poor perceptual quality. This phenomenon is because minimizing MSE encourages finding pixel-wise averages of plausible solutions [72, 13]. Therefore, these metrics are not good proxies for the perceptual quality of the restored image. Metrics that better capture this perceptual quality are thus needed. Given this observation, the authors of [72] proposed the VGG loss, which is the Euclidean distance between the feature representations of a reconstructed image and the GT image. These representations are based on the ReLU activation layers of the pre-trained 19-layer VGG network [119]. VGG stands for Visual Geometry Group. Authors of [160] proposed another metric based on deep feature maps, called LPIPS.

1.4.2 Qualitative evaluations

We saw that numerical metrics such as PSNR or SSIM are not the best proxies of quality of the reconstruction. Even if one uses a perceptual metric—that are not always easy to compute and interprete—, qualitative evaluation based on human eyes should be always done. One must pay attention whether areas of images with important amount of information (structures, edges, textures, high frequency details, ...) are well reconstructed.

1.4.3 Consideration of the temporal dimension

In video restoration, one should also evaluate perceptual quality regarding temporal consistency. A temporally consistent restored video shows less unpleasant flickering artifacts. This quality can be assessed in two ways. The first method is based on visual inspection and involves plotting temporal profiles. A temporal profile is produced by taking the same horizontal row of pixels from several successive frames in the restored video and stacking them vertically into a new image. Flickering artifacts in the video appear as jitter and jagged lines in the temporal profile. A sharper and less noisy temporal profile means less flickering artifacts, *i.e.*, better temporal consistency [110]. The second method regroups numerical metrics, including:

- vector norm differences of warped frames [26];
- tOF and tLP [29]. tOF measures the pixel-wise difference of motions estimated from sequences, *i.e.*, is the L1 norm between the optical flow between x_t and x_{t-1} and the one between $\hat{x}_t$ and $\hat{x}_{t-1}$. The authors who introduced these metrics proposed to use the Farneback algorithm [38] for optical flow computation. tLP measures perceptual changes over time using deep feature maps, *i.e.*, is the L1 norm between the LPIPS between x_t and x_{t-1} and the one between $\hat{x}_t$ and $\hat{x}_{t-1}$.

1.4.4 Speed and memory size

The amount of time required to restore an image or a video is an important point to consider from a realistic application-oriented point of view. Measuring such quantity for a restoration method is thus relevant. DL-based methods can benefit from efficient GPU-based parallel architectures and are generally faster than classical iteration-based methods. However, DL-based methods require storing networks' weights and intermediate feature maps in memory, thus requiring more memory than classical methods.

The runtime of a method can be directly measured and expressed in unities such as ms for a fixed input image size. Moreover, it can also be quantified based on FLOating-Ooint operations (FLOPs), *i.e.*, the total number of floating point operations required for a single forward pass. The higher the FLOPs, the slower the model. This measure can also be expressed in MAC (Multiply–accumulate operations).

1.4.5 Interpretability/explainability

A CNN that takes an input degraded image or video and outputs the restored one works as a black box model. Indeed, the underlying regression is implicitly learned via end-to-end training, and it is hard to discover what is actually learned inside the networks by examining the network parameters, which are usually of high dimensionality, and what are the roles of individual parameters [90]. In other words, this CNN is difficult to interpret. This is problematic concerning its industrial deployment. To certify a specific restoration model in an industrial context, this model needs a certain explainability of results and some guarantees about considerations such as quality or convergence. To meet these requirements, the model needs to be interpretable. As an example, in their deep unrolling framework, the authors of [31, 11] derived a theoretical upper bound on the restoration error when the input is perturbed.

1.4.6 Test datasets

Evaluations of a restoration model should use validation/test sets that simulate a realistic application-oriented scenario. To facilitate comparisons, open-source and public datasets are shared among the research community. The most popular public test sets for image restoration are:

- Set14 [154];
- Set5 [12];
- BSD68 [83];

and the most popular test video benchmarks are:

- Vimeo-90K-T [146];
- REDS4 [91];
- Vid4 [77];
- UDM10 [150].

1.5 Conclusion

There exist two paradigms of inverse problem-solving in image and video restoration. The first one regroups model-based (also called "classical") methods that directly derive from the forward model of the inverse problem as a minimization problem. This problem is ill-posed, therefore model-based methods rely on hand-crafted priors that require expert knowledge about the inverse problem and time-consuming regularization hyperparameter tuning. Moreover, they use iterative optimization algorithms that are time-consuming and not suitable for some practical applications. The second paradigm regroups data-based, *i.e.*, DL-based methods that learn to inverse the forward model by being trained on pairs of degraded and original image/video. These methods learn data-based prior that better captures natural image/video statistics than a handcrafted one. They additionally suppress the need for carefully tuning the regularization hyperparameter. Moreover, once trained, they only require a forward propagation to restore an image video and can benefit from efficient GPU-based computing architectures. From this point of view, they are more adapted for practical applications than model-based methods. However, contrary to model-based methods, they present low explainability due to their black box nature. Moreover, unlike classical approaches, DL-based methods that take only the degraded image/video as input lack the flexibility to deal with multiple degradations. Given the advantages and disadvantages of both paradigms, recent hybrid approaches combine and benefit from them in a complementary way. These approaches regroup the following methods:

- the ones that non-blindly give the knowledge about the forward model to the input of the CNNs;
- the ones that use classical iterative algorithms and replace the prior-related denoiser in each iteration step with a denoising NN.
- (deep unrolling) the methods that unfold for a fixed number of iterations a classical iterative algorithms and replace some operators with CNNs. The unfolded iterations together form a whole CNN that is trained based on data.

- (DIP & DeepRED) the methods that rely on a classical algorithm and an implicit prior captured by a CNN.

Tab. 1.1 summarizes the advantages and disadvantages of different approaches. These key differences should guide the choice of a method when facing realistic image/video restoration applications. Regarding interpretability, compared to the methods that connect DL with the perfectly interpretable framework of iterative algorithms, the ones that give the knowledge about the forward model to the input of the CNNs are less interpretable. In MD or SFTMD, even if a CNN can flexibly manage multiple degradations, this model is still used as a black box. Regarding a method that first transforms the input with regularized inversion, in the subsequent stage the denoising CNN is still used as a black box.

Method	Classical	DL	DL + degradation knowledge	Classical + CNN denoiser	Deep unfolding	DIP
Inference speed	Slow	Fast	Fast	Slow	Fast	Slow
Supervised pretraining	No	Yes	Yes	Yes	Yes	No
Flexibility	Yes	No	Yes	Yes	Yes	Yes
Interpretability	++	0	+	++	++	++
Learned prior	No	Yes	Yes	Yes	Yes	Yes
Regularization parameter tuning	Yes	No	No	Yes	No	No

Table 1.1: Advantages and disadvantages of inverse problem-solving methods.

The performance of an image/video restoration method should involve both qualitative and quantitative evaluations. They must be based on an appropriately chosen test dataset. Regarding quantitative evaluation, one should be aware that different numerical metrics measure different things.

Part II

Contributions

Chapter 2

Video Super-Resolution

In Video restoration, a dynamic scene with continuous intensity distribution $X(x, y)$ is seen to be warped at the camera lens because of the relative motion between the scene and the camera. The image is blurred both by atmospheric turbulence and the camera lens. These blurring are respectively modeled by continuous PSFs H_{atm} and H_{cam}. After these blurrings, the image is discretized at the CCD resulting in a noisy digitized frame. These degradations are summarized by Fig. 2.1 and the following forward model [40]:

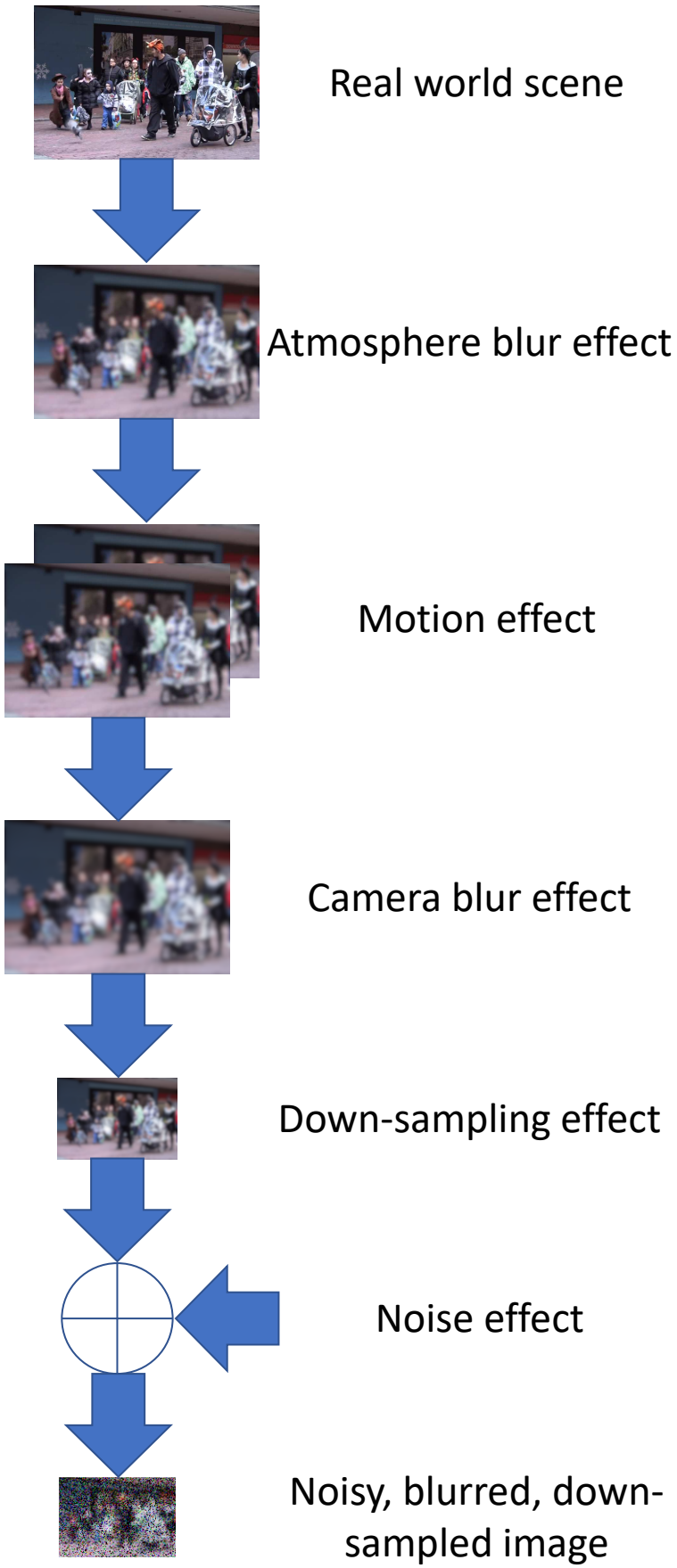


Figure 2.1: Diagram representation of the VSR forward model. Figure inspired from [40].

$$Y[m,n] = [H_{\text{cam}}(x,y) * F(H_{\text{atm}}(x,y) * X(x,y))] \downarrow + V[m,n] \tag{2.1}$$

where $*$ is the two-dimensional convolution operator, F is the warping operator, $\downarrow$ is the discretizing, *i.e.*, downsampling operator, $V[m,n]$ is the system noise, and $Y[m,n]$ is the resulting discrete, noisy and blurred image.

Eq. (2.1) is equivalent to the following model in the pixel domain:

$$y_j = D_j H_j F_{u_{t\to j}} H_t^{\text{atm}} x_t + n_j \qquad t - J \leq j \leq t + J \tag{2.2}$$

where:

- $y_j \in \mathbb{R}^N$ is the data observed at time step j, corresponding to a low-resolution (LR) video frame of size $N = N_1 \times N_2$ arranged in a lexicographic order. $2J + 1$ is the total length of the observed video;
- $x_t \in \mathbb{R}^{s^2 N}$ is the underlying data, corresponding to a high-resolution (HR) reference video frame at time step t, arranged in a lexicographic order;
- $H_t^{\text{atm}} \in \mathbb{R}^{s^2 N \times s^2 N}$ is the matrix defining the atmospheric blur;
- $F_{u_{t\to j}} \in \mathbb{R}^{s^2 N \times s^2 N}$ denotes the warping operator with regard to the optical flow $u_{t\to j}$. This vector field models the motion between the two time steps j and t (detailed later in Sec. 2.2.3);
- $H_j \in \mathbb{R}^{s^2 N \times s^2 N}$ is the matrix defining camera lens blur;
- $D_j \in \mathbb{R}^{N \times s^2 N}$ is the matrix defining the downsampling. This operation samples every s-th pixel in each spatial dimension. s is an integer called scale factor;
- $n_j \in \mathbb{R}^N$ accounts for the presence of the noise, which is mostly assumed to be additive, zero-mean and white Gaussian. Realizations of this noise are assumed to be independent and identically distributed.

In conventional imaging systems (such as video cameras), camera lens blur has a more significant effect than atmospheric blur (which is not true if the scene is very far, let's say more than 10km away from the camera, in applications such as atmospheric turbulence mitigation or astronomical image deconvolution). Therefore, the atmospheric blur is omitted in this chapter, leading to the following formulation:

$$y_j = D_j H_j F_{u_{t\to j}} x_t + n_j \qquad t - J \leq j \leq t + J \tag{2.3}$$

In most situations, the downsampling and camera blurring operations remain constant over time, assuming that the images are obtained from the same camera. Moreover, we further assume that the camera PSF is space-invariant [40, 39]. These assumptions give the following model:

$$y_j = D_s H F_{u_{t\to j}} x_t + n_j \qquad t - J \leq j \leq t + J \tag{2.4}$$

with D_s denoting the downsampling with scale factor s.

2.1 Classical VSR

Supposing H is known, the inverse problem defined by the model in Eq. (2.4) is classically solved by the following alternated optimizations [77]:

$$\hat{x}_t = \arg\min_{x_t} ||D_s H x_t - y_t||_2^2 + \sum_{j=t-J, j\neq t}^{t+J} ||D_s H F_{u_{t\to j}} x_t - y_j||_2^2 + \rho(x_t) \tag{2.5}$$

$$\hat{u}_{t\to j} = \arg\min_{u_{t\to j}} ||D_s H F_{u_{t\to j}} x_t - y_j||_2^2 + \varphi(u_{t\to j}) \qquad t - J \leq j \leq t + J \tag{2.6}$$

where $\rho(x_t)$ and $\varphi(u_{t\to j})$ are hand-crafted priors on the estimated frame and optical flow. For example, [40] used the Bilateral TV prior for $\rho(x_t)$. [77] used sparsity on derivative filter responses to model $\rho(x_t)$ and $\varphi(u_{t\to j})$. N defines the number of frames used in order to produce the estimate $\hat{x}_t$.

In general situations, the whole process of the optimizations is slow, involving alternated iterative algorithms [77]. However, under some assumptions, the VSR can be accelerated [40, 39]. Suppose there are no mobile objects. In some situations—*e.g.*, when motions only come from vibrations of a gazing camera or a panning motion of a faraway scene—, motions are purely translational. In these conditions, H and $F_{u_{t\to j}}$ are block-circulant and commute. Moreover, it is easy and fast to correctly estimate translational motion by maximizing the correlation between the warped degraded frame at time step t and the degraded frame at time step j. We note $F_{t\to j}$ such warping, and this operation is defined by only two parameters (translations in horizontal and vertical directions). We thus only need to solve the remaining problem:

$$\hat{x}_t = \arg\min_{x_t} ||D_s H x_t - y_t||_p^p + \sum_{j=t-J, j\neq t}^{t+J} ||D_s F_{t\to j} H x_t - y_j||_p^p + \rho(x_t) \tag{2.7}$$

Let $H x_t = z_t$. This is the blurred version of x_t. The problem in Eq. (2.7) can be solved in two steps, following the *shift-and-add* algorithm [40]:

1. we first estimate $\hat{z}_t$ from the degraded frames;
2. we then deconvolve $\hat{z}_t$ to output the estimate $\hat{x}_t$.

The finding of z_t solves the following problem:

$$\hat{z}_t = \arg\min_{z_t} ||D_s z_t - y_t||_p^p + \sum_{j=t-J, j\neq t}^{t+J} ||D_s F_{t\to j} z_t - y_j||_p^p \tag{2.8}$$

$||D_s z_t - y_t||_2^2$ has been replaced by $||D_s z_t - y_t||_p^p$ to potentially make the estimator more robust to outliers, *i.e.*, data points with different distributional characteristics than the assumed model. The concept of breakdown can illustrate this [18]. The breakdown point is the smallest proportion of outliers that may force the value of the estimate outside some range. As an illustration, the simple mean estimator's breakdown point is zero, which means that one outlier is enough to make the estimate fall outside any predicted bound. In contrast, the median estimator may achieve a breakdown equal to 0.5, which is the highest value for breakdown points. Therefore, the median estimation may not be impacted by data sets in which outliers constitute less than 50% of the data. Thus, the median estimator is more robust than the mean estimator.

By computing the partial derivative with respect to z_t, from Eq. (2.8) we obtain the following equalities:

$$0 = \begin{cases} D_s^T(D\hat{z}_t - y_t) + \sum_{j=t-J, j\neq t}^{t+J} F_{t\to j}^T D_s^T(D_s F_{t\to j}\hat{z}_t - y_j), & \text{if } p = 2 \\ D_s^T sign(D\hat{z}_t - y_t) + \sum_{j=t-J, j\neq t}^{t+J} F_{t\to j}^T D_s^T sign(D_s F_{t\to j}\hat{z}_t - y_j), & \text{if } p = 1 \end{cases} \tag{2.9}$$

We derive from these equalities that [37]:

- when $p = 2$, $\hat{z}_t$ is the pixel-wise average of measurements after image registration, *i.e.*, copying from the LR grid to the HR grid after proper shifting and zero-filling;
- when $p = 1$, $\hat{z}_t$ is the pixel-wise median of measurements after image registration.

The deconvolution to estimate $\hat{x}_t$ from $\hat{z}$ can be either done by classical or DL-based methods. If one uses a classical method, in the under-determined case (*i.e.*, when $2N + 1 < s$), some pixel locations will have no estimate. For these cases, it is essential to have a regularization term. Authors of [40] proposed a robust regularizer called bilateral TV, which is computationally light and preserves edges. This prior consists in setting:

$$\rho(x_t) = \underbrace{\sum_{l=-P}^{P} \sum_{m=0}^{P}}_{l+m\geq 0} \alpha^{|m|+|l|} ||x_t - S_x^l S_y^m x_t||_1 \tag{2.10}$$

Operators S_x^l and S_y^m shift x_t by l and m pixels in horizontal and vertical directions respectively. They enable presenting several scales of derivatives. The scalar weight $\alpha \in [0, 1]$ is applied to give spatially decaying weights to the linear combination of the regularization terms. The bilateral TV regularization is indeed a generalization of the TV regularization. Authors of [96] proposed to do this deconvolution based on a residual CNN.

2.2 Deep VSR

VSR has recently benefited from DL methods [110, 134, 66, 150, 42, 62, 63]. We will see that these methods can be classified into three paradigms. However, all of them typically consist of the common four components: feature extraction, alignment, fusion, and reconstruction. The challenge mainly lies in the design of the alignment and fusion modules, especially when a video contains occlusion, large motion, and severe blurring. Generally, the fusion occurs after alignment [134, 110, 150, 63] and both tasks are separately done, but some VSR networks operate both tasks simultaneously [42, 66].

In this section, we will first explain the three deep VSR paradigms. We will see that one is more attractive for practical applications than the others. We will therefore explain it in more detail. Then, we will present alignment techniques used in SOTA VSR networks. Finally, we will present fusion techniques elaborated in the deep VSR community.

2.2.1 Classification of deep VSR methods

There are broadly three classes of deep VSR methods. The first one groups **sliding-window based** models. These models [134, 66, 150, 63, 78] take a batch of multiple LR frames as input to fuse them and reconstruct an HR frame. In most cases, this batch contains 5 to 7 LR frames. Therefore, the temporal receptive field—in other words, the number of LR frames used in order to super-resolve a frame—is limited to 7.

In contrast, methods introduced in [110, 42, 62], that build upon **recurrent** models, enable a larger temporal receptive field. In these networks, to super-resolve a frame at time step t, the hidden states and/or output computed in the previous time step $t-1$ are taken as input, in addition to a batch of 1 to 3 LR frames. This recursion allows propagating information through a large number of frames. As their input batch contains fewer LR frames and their network structures are mostly simpler, recurrent methods are faster than sliding-window based methods. Moreover, an inference of a recurrent model presents less redundant computations than the one of a sliding-window based model because each frame is processed only once. Finally, sliding-window based methods generate independent output HR frames, which reduces the temporal consistency of the produced HR frames, resulting in flickering artifacts. This is not the case for recurrent VSR, in which information about the previously super-resolved frame is part of the input at each time step. However, we demonstrate for the first time that contrary to sliding-based ones, recurrent VSR networks present an instability problem on a certain type of video. This point will be investigated in Sec. 2.5.

The third class of deep VSR networks regroups **bidirectional** methods [24, 23, 149]. These models take a batch of an arbitrary number of LR frames and super-resolve all of them after forward and backward propagations of recurrent information. This bidirectional scheme maximizes information-gathering within the batch but presents problems that limit its practical applicability. First, it can only allow for offline processing of this batch. This is contrary to the aforementioned unidirectional recurrent networks that can process the

incoming LR frames online. Therefore, incoming LR frames should be buffered before being processed, and the temporal receptive field is limited to the length of the buffer, similarly to sliding-window based methods. Between consecutive buffers, temporal discontinuity produces flickering artifacts. Second, the need for both forward and backward propagations doubles the computation time.

2.2.2 Recurrent video super-resolution

This section presents SOTA recurrent VSR methods, as they seem to regroup the most attractive networks from an application-oriented point of view.

FRVSR

Authors of [110] were pioneers of deep recurrent VSR. They introduced Frame-Recurrent Video Super-Resolution (FRVSR), in which the previous output frame is warped based on a dense optical flow estimation and fed back as an additional input to a super-resolution network at the next time step. Hence, FRVSR operates *frame-recurrence*. Fig. 2.2 is an overview of FRVSR. Another network estimates the optical flow, and the two networks are jointly trained end-to-end. The loss function is the sum of the supervised super-resolution loss and the unsupervised warping loss. The former is the MSE between the restored frame and the corresponding GT frame at time step t. The latter is expressed as follows:

$$||F_{\hat{u}_{t-1\to t,LR}} y_{t-1} - y_t||_2^2 \tag{2.11}$$

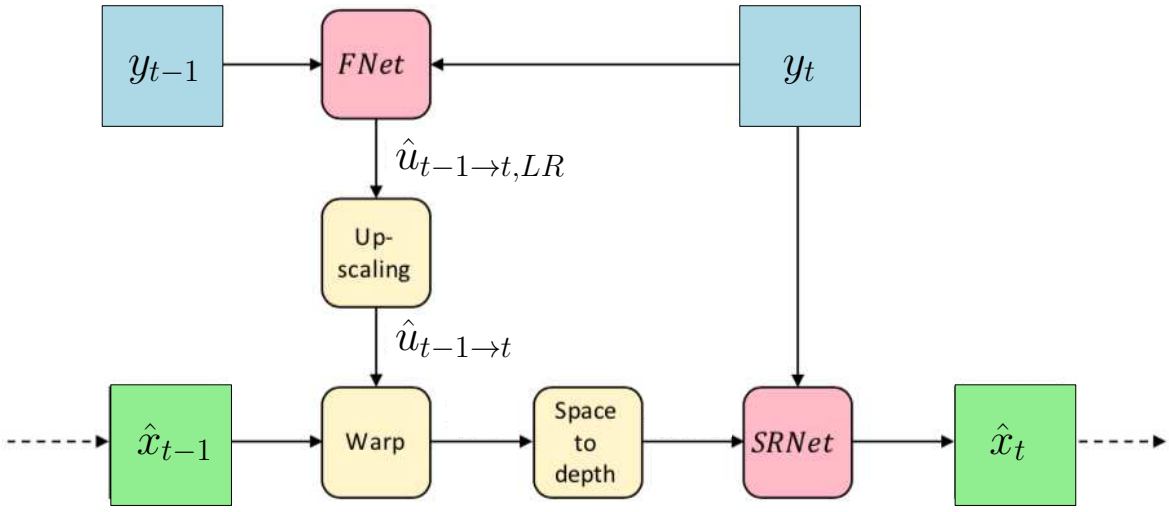


Figure 2.2: Overview of FRVSR. FNet is based on an encoder/decoder style architecture, increasing the receptive field of the convolutions. Its architecture is illustrated in Appendix. A.4. SRNet is based on an architecture with successive residual blocks [54] followed by transposed convolutions for upsampling. Figure adapted from [110].

RLSP

A more recent recurrent VSR architecture called *Recurrent Latent Space Propagation (RLSP)* (RLSP) was introduced in [42]. In this approach, the previous output frame and the previously estimated locality-based hidden state are used as extra input at the next time step. Fig. 2.3 presents the architecture of RLSP. Compared to frame-recurrence, RLSP can be interpreted as maximizing the depth and width of the recurrent connection. In contrast to FRVSR, RLSP is based on implicit motion compensation. The overall architecture is computationally efficient, which enables RLSP to be the fastest VSR network at this time.

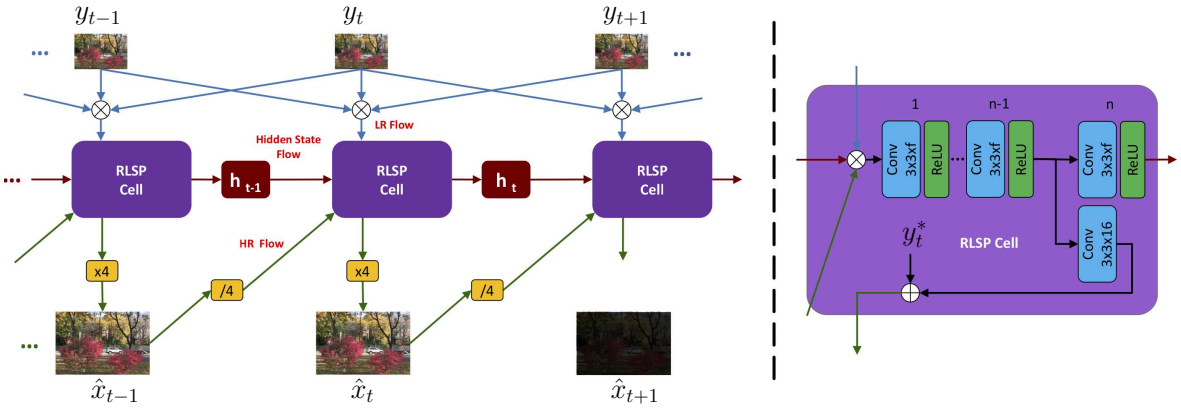


Figure 2.3: RLSP. Figure taken from [42]

RSDN

Recurrent Structure-Detail Network (RSDN) [62] is so far the recurrent VSR network that reportedly performs the best for relatively short sequences, according to its performance on the Vid4 dataset, composed of 4 videos between 34 to 49 frames [77]. Its architecture presents a recurrent hidden state coupled with a hidden-state adaptation module and structure-detail decomposition. The input LR frames and the hidden state are decomposed into structure and detail components and fed to two interleaved branches to reconstruct the corresponding components of HR frames.

2.2.3 Image/feature alignment techniques

Whether the VSR network is sliding-window based, recurrent or bidirectional, frame or feature alignment of the images in the input LR batch with respect to the reference frame to be super-resolved, is an important step. This section details some of the SOTA alignment techniques.

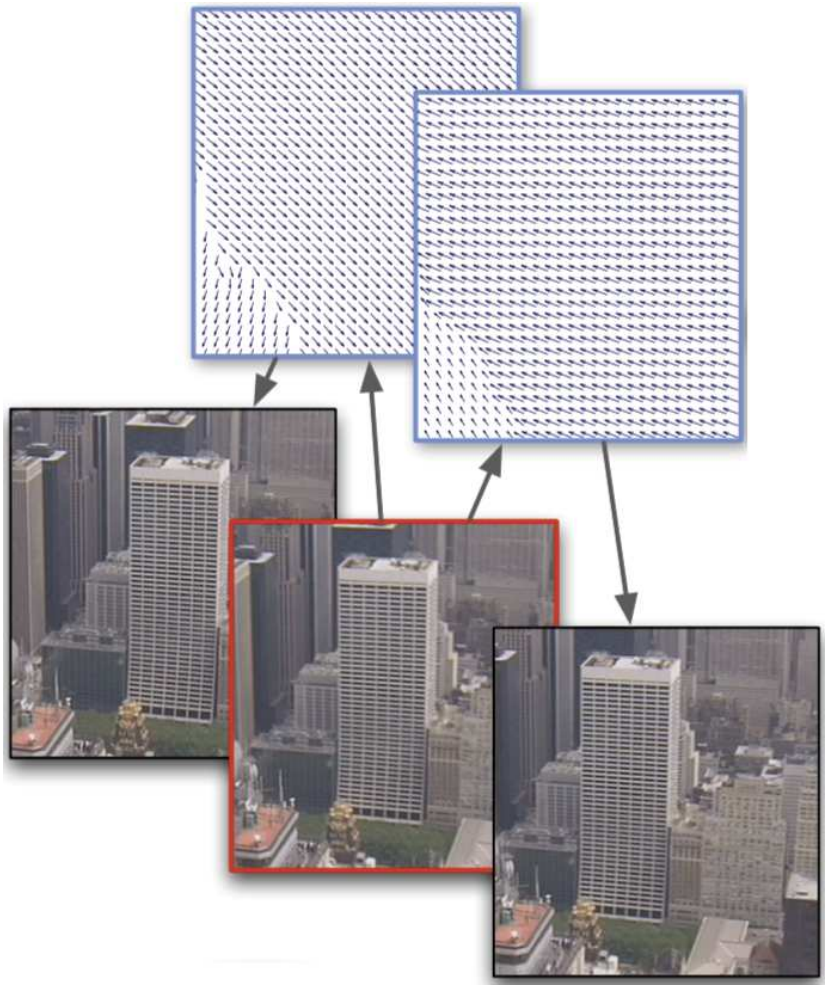


Figure 2.4: Illustration of optical flow. Figure taken from [77].

Optical flow-based alignment

Optical flow is a vector field quantifying the apparent motion of individual pixels on the image plane. It is a good approximation of the true physical motion projected onto the image plane. Fig. 2.4 illustrates this vector field. Estimating the optical flow field between two frames is indeed an inverse problem, and this estimation allows to align the two frames afterward by explicit warping, *i.e.*, motion compensation. Classically, optical flow fields are estimated based on Bayesian inference, iterative algorithms and an explicit regularization [77, 152, 88, 139]. These regularizations are mostly derived from TV regularization. The most recent and successful optical flow estimations are done by neural networks [120, 110, 101, 60]. Some video restoration systems thus include auxiliary DL-based optical flow estimation modules that can be either pretrained [97, 124] or trained end-to-end along with the main video restoration branch [110]. Since we generally do not have access to GT flow in most scenarios, unsupervised warping loss is used as a loss function [110, 124, 151, 79]. This loss is expressed as follows:

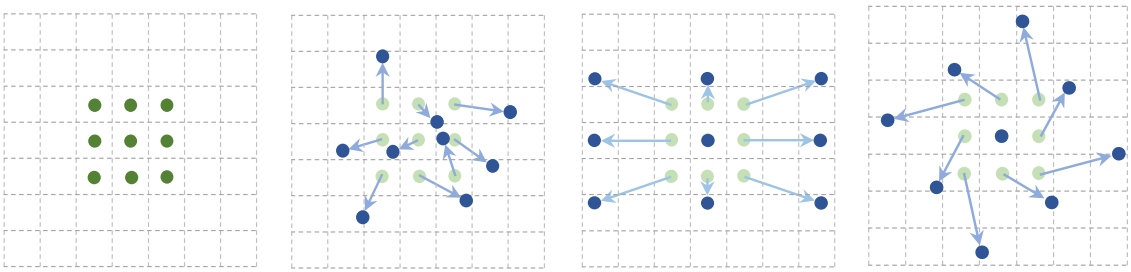
$$\mathcal{L}_{flow} = \sum_{j=t-J, j\neq t}^{t+J} \mathcal{L}_{flow_{j\to t}} = \sum_{j=t-J, j\neq t}^{t+J} ||F_{\hat{u}_{j\to t,LR}} y_j - y_t||_p^p + \varphi(\hat{u}_{j\to t}) \tag{2.12}$$

where $\hat{u}_{j\to t,LR}$ is the estimated optical flow projected in the LR space, $F_{\hat{u}_{j\to t,LR}}$ is the warping operator based on $\hat{u}_{j\to t,LR}$ and $\varphi(\hat{u}_{j\to t})$ is a regularizer. This explicit regularization term is mostly not used when a CNN estimates the optical flow field. In this case, the regularization is only implicit, based on data.

Once the flow fields are estimated, the motion compensation can occur either at image [110, 17] or feature [110, 96] spaces. However, optical flow estimation for explicit motion compensation encounters some issues. First, accurate flow is difficult to estimate if occlusion and large motion occur. Second, the authors of [146] showed that exact optical flow computation is intractable and could be suboptimal for a specific task. Incorrect optical flow estimation may corrupt original frames by provoking apparent artifacts, and decrease the performance of the restoration, even if the errors are minor. Third, the motion estimation module requires additional memory space. As an example, the FlowNet in FRVSR [110] accounts for around 40% of the total parameters of the VSR system.

Deformable convolution-based alignment

Deformable convolutions have been firstly introduced in [33]. A deformable convolution layer is a convolution layer with additionally learned spatial offsets. These offsets allow the deformable convolution to retrieve information away from the regular local neighborhood. The concept of deformable convolution is illustrated in Fig. 2.5. The technique has been used in various computer vision tasks, including object detection [10], action recognition [163], and semantic segmentation [33]. For VSR, the work [126] firstly used it to align the input frames at the feature level without explicit motion estimation or image warping. This alignment works as follows. Let F^a_{t+i} denotes the features aligned with respect to the reference features F_t at each pixel position $\mathbf{p}_0$. Moreover, let w_k and p_k denote the weight



(a) (b) (c) (d)

Figure 2.5: Illustration of the sampling locations in 3×3 standard and deformable convolutions, taken from [33]. (a) regular sampling grid (green points) of standard convolution. (b) deformed sampling locations (dark blue points) with augmented offsets (light blue arrows) in deformable convolution. (c)(d) are special cases of (b), showing that the deformable convolution generalizes various transformations for scale, (anisotropic) aspect ratio and rotation.

and the pre-specified offsets for the k-th location. For example, in the case of a 3×3 kernel, $K = 9$ and $p_k \in \{(1,1),(1,0),,(1,1)\}$. The deformable convolution-based alignment is expressed as:

$$F^a_{t+i}(\mathbf{p}_0) = \sum_{k=1}^{K} w_k \cdot F_{t+i}(\mathbf{p}_0 + \mathbf{p}_k + \Delta\mathbf{p}_k) \cdot \Delta m_k \tag{2.13}$$

where the offset $\Delta\mathbf{p}_k$ and the modulation scalar Δm_k are learnable and predicted from concatenated features of a neighboring frame and the reference one:

$$\Delta\mathbf{P}_{t+i} = f(\,[F_{t+i}, F_t]\,) \tag{2.14}$$

where $\Delta\mathbf{P}$=$\{\Delta\mathbf{p}\}$, f is a CNN module, and $[\cdot,\cdot]$ denotes the concatenation operation. The same logic applies to the modulation scalar Δm_k.

Inspired by the work [126], the authors of [134] used deformable convolutions in a pyramidal—*i.e.*, multi-scale—manner for feature alignments: they first align features in lower scales with coarse estimations and then propagate the offsets and aligned features to higher scales, similarly to previous studies on optical flow estimation [101, 120]. This pyramidal scheme enables to model large motion. In addition, the authors added the mechanism of cascading refinement, inspired again by previous works such as [60, 58, 59].

Authors of [25] showed that deformable convolution could be decomposed into a combination of spatial warping and convolution. This decomposition raises the common point of deformable and flow-based alignments in formulation but with a key difference in offset diversity. They showed that the increased diversity in deformable alignment yields better-aligned features, significantly improving video super-resolution performance. They further proposed an offset-fidelity loss that guides offset learning with optical flow. This loss prevents the overflow of offsets and alleviates the instability problem of deformable alignment.

VSR networks that operate deformable convolutions for feature alignment have yielded better results than the ones relying on optical-flow based explicit motion compensation. However, deformable convolutions highly consume memories and add a significant amount of parameters. For information, the authors of [134] used 8 NVIDIA Titan XP GPUs in parallel to train their Video Restoration framework with Enhanced Deformable convolutions (EDVR). Fig. 2.8 shows EDVR is highly memory-demanding.

Non-local block

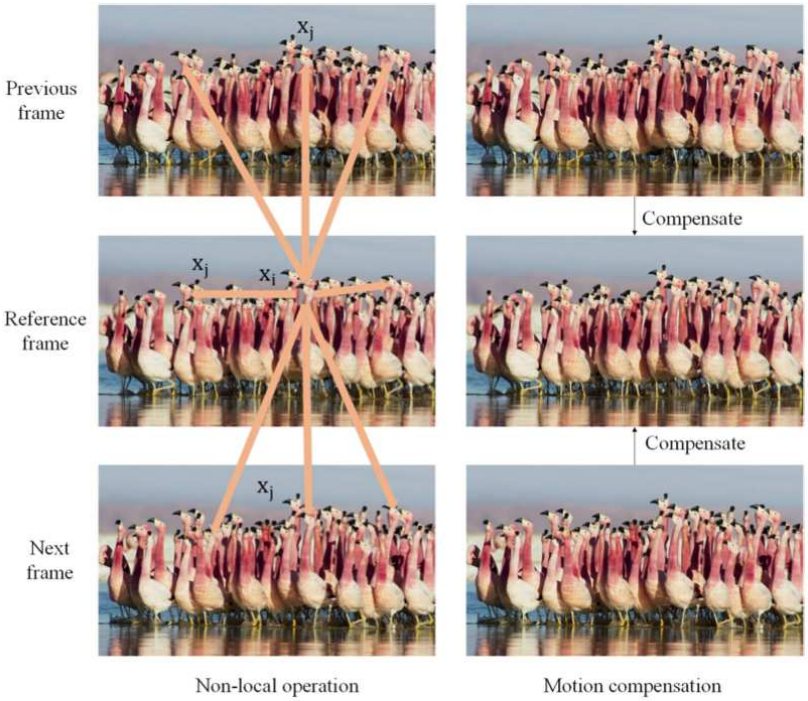


Figure 2.6: Differences of non-local operation and explicit motion compensation. Non-local operation tries to obtain the response at position x_i by computing the weighted average of relationships of all possible positions x_j [133]. Figure taken from [150].

In their Progressive Fusion Video Super-Resolution (PFNL) network, the authors in [150] proposed to do an implicit alignment based on their newly proposed Non-Local Residual Block (NLRB). This block captures long-range spatio-temporal correlations among pixels of the images in the input sliding-window. Fig. 2.6 illustrates the concept of non-local interaction modeling. Non-local operation tries to obtain the response at position x_i by computing the weighted average of correlations with all possible positions x_j. Mathematically, this operation is described as follows:

$$y_i = \frac{1}{C(x)} \sum_j f(x_i, x_j) g(x_j) \tag{2.15}$$

where x represents the input data, y denotes the output having the same size as x. i is

the index of an output position, and j is the index of all possible positions. The function f calculates a scalar representing the correlation between two inputs, while g gives a representation of the input. $C(x)$ is used for normalization. The authors considered the Gaussian function $f(x_i, x_j) = e^{x_i^T x_j}$, where $x_i^T x_j$ represents the dot-product similarity, and $C(x) = \sum_j f(x_i, x_j)$ is used for normalization.

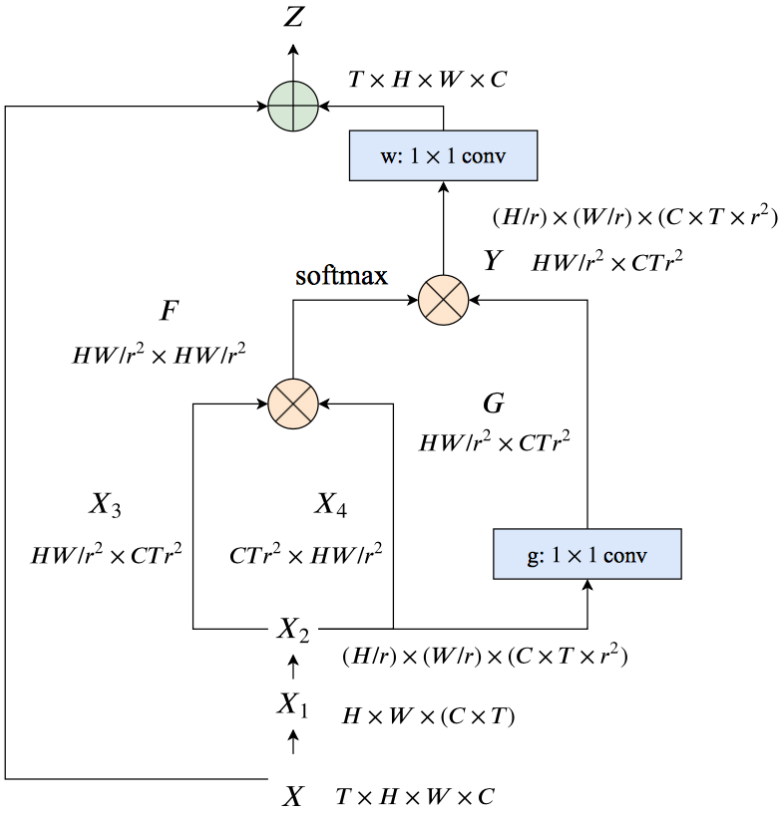


Figure 2.7: Structure of an NLRB. We show the feature maps as their shapes like $THWC$, which are reshaped if noted. The $\otimes$ represents matrix multiplication and $\oplus$ represents element-wise addition. Figure taken from [150].

Fig. 2.7 shows the structure of an NLRB. As one can see on it, the output of the block is $z_i = W_z y_i + xi$, where W_z is implemented by 1×1 convolution. The temporal dimension of the input is first transformed into the channel dimension. Indeed, the temporal correlations are captured through the channel correlations. Moreover, the feature map is secondly shuffled. These two steps allow to lower the required memory. Without them, a memory error probably occurs on most machines.

Dynamic upsampling filters

The work in [66] proposed Dynamic Upsampling Filters (DUF) for implicit motion compensation. First, these filters are constructed based on the input sliding-window, capturing motion information. Then, the HR frame is directly constructed by locally applying these filters to the center input frame. However, as DUF needs to estimate dynamic filters in each

location, the algorithm suffers from heavy computation, as illustrated in Fig. 2.8. Moreover, DUF generates a strong border effects.[1]

Recurrent Latent Space Propagation

In RLSP and RSDN, the recurrent convolutional layers simultaneously allow implicit alignment and feature fusion.

2.2.4 Feature fusion

Once the images or the features are explicitly or implicitly aligned, they must be fused to complete the super-resolution task. This section presents fusion techniques used in some SOTA networks, if the fusion occurs after alignment.

Direct fusion

The direct fusion strategy fuses multiple frames into one part from the beginning [110, 42]. They are concatenated in the channel dimension and given to a CNN. This strategy is the most simple and popular fusion method.

Progressive fusion

Authors of [150] introduced this fusion method. It consists in cascading several progressive fusion residual blocks. This block alternates individual feature extraction from each frame and mixed temporal information extraction. This way, both intra-frame spatial correlations and inter-frame temporal correlations are extracted progressively.

Temporal and spatial attention

The study from [134] used a temporal and spatial attention mechanism for fusion. This mechanism assigns pixel-level aggregation weights to each frame. The temporal attention computes frame similarity in an embedding space. Similar to the reference frame a neighboring one is, its attention coefficient is higher. Subsequently, spatial attention masks are then computed from the fused features. A pyramid design is employed to increase the attention receptive field. After that, the masks modulate the fused features through element-wise multiplication and addition, similar to [135].

[1] This border effect is visible on images available on https://github.com/yhjo09/VSR-DUF.

Temporal group attention

The work in [63] divided into several groups the input LR frames in the sliding-window. In the first place, each group undergoes an intra-group fusion. After that, the inter-group fusion module involves a temporal attention mechanism that computes a spatial mask for each group.

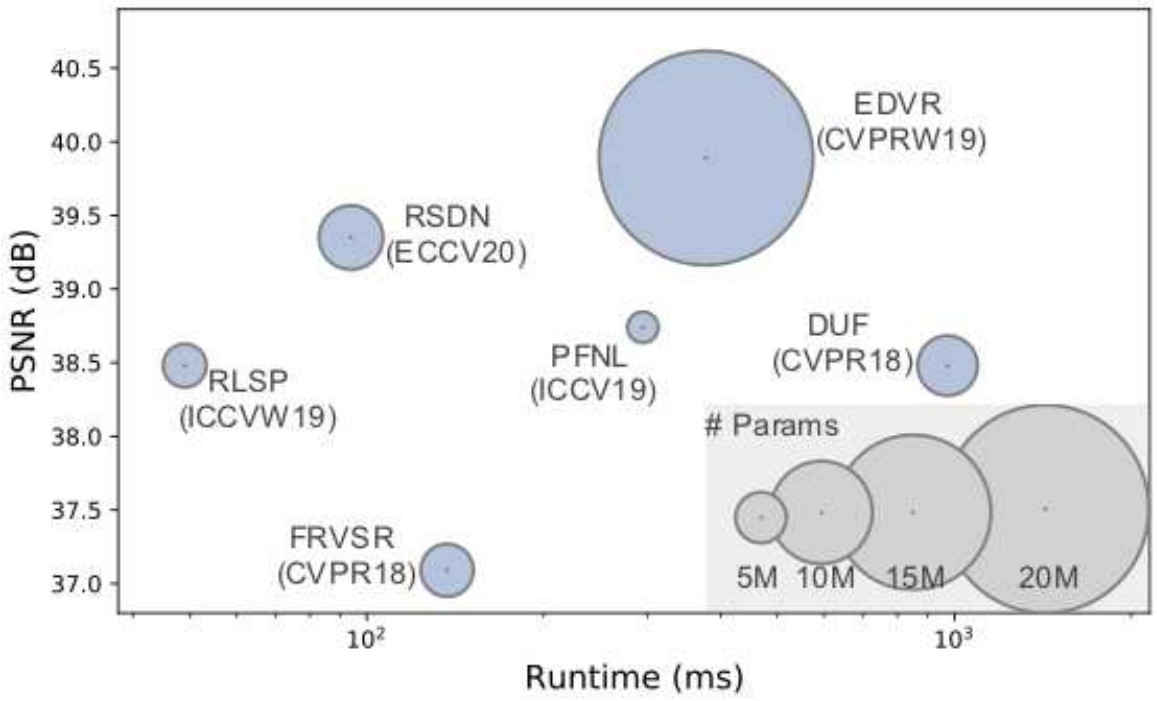


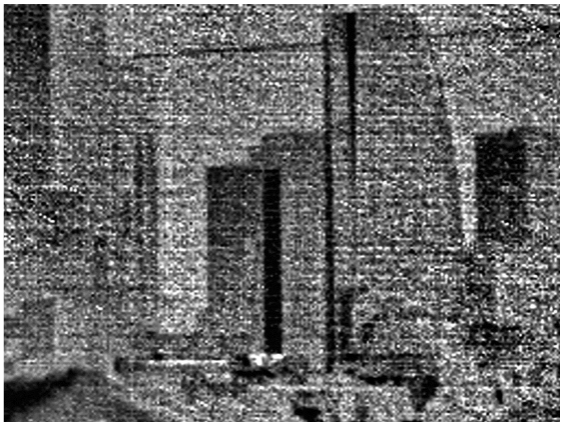
Figure 2.8: Speed and performance comparison, based on UDM10 dataset. [150]. Figure adapted from [24].

2.3 Comparison between classical and DL-based video restoration

This section compares DL-based and classical video restoration methods for restoration tasks that are not restricted to VSR. One should note that some restoration tasks are indeed special cases of VSR. For instance, in the case $s = 1$, Eq. (2.4) is a video deconvolution problem. If $s = 1$ and $H = I$ we have a video denoising problem. In contrast, for some problems such as atmospheric turbulence mitigation, the linear degradation is more complicated than $D_s H$.

In some cases of video restoration where the motions of the scene and objects are simple and well characterized, classical methods are adapted and provide excellent results. Fig. 2.9 and Fig. 2.10 illustrate these performances in static scenes, where DL-based approaches

do not perform as well. In the presence of moving objects, the classical methods can be applied on a local patch containing each object, where the motion model becomes simple and provides good performance, as shown in Fig. 2.11.



(a) Noisy

(b) Denoised

Figure 2.9: Classical static scene denoising (temporal and fixed pattern noise).

(a) Input degraded frame

(b) DL (EDVR [134])

(c) Classical (shift-and-add with iterative deconvolution)

Figure 2.10: Static scene video super-resolution.

When the motion is complex or difficult to estimate, neural networks can learn to compensate for this motion and take this information into account in the super-resolution process in a robust way. Fig. 2.12 shows that in this scene with complex motion, the deep-learning approach performs similarly as on the static scene, which is better than classical methods that are not well suited to this complex/dynamic setting.

Figure 2.11: Classical VSR with a mobile object. Left: input degraded frame. Right: the corresponding super-resolved frame.

In atmospheric turbulence, where the forward model (generating turbulence) is complex and highly ill-posed, it is challenging to design a classical algorithm to solve the problem. This algorithm would involve both luminance and dense motion field estimation with too many unknowns. Deep learning-based methods can provide reasonably good solutions, as highlighted in Fig. 2.13, needing only to simulate the forward model in the data generation process.

2.4 Deep Unrolled Network for VSR

So far, we have separately presented data-based and model-based VSR methods. In this section, we explore the technique that consists of blending these two paradigms to benefit from both of them in a complementary way. Indeed, we detail the designing of a new VSR network based on the technique of deep unrolling.

Previous works on unrolled optimization algorithms focus on single image restoration, such as image denoising, deblurring and SISR. To the best of our knowledge, unrolled optimization algorithm has never been explored to tackle VSR. On the other hand, most studies on VSR currently search for the best performing purely learning-based network under single degradation, without incorporating the image formation model in the network nor dealing with multiple degradations. In this context, we introduce a framework coined Unrolled Video Super-Resolution (UVSR) that resembles FRVSR [110] but is based on unrolled optimization. In other words, we first model an image sequence formation model, and UVSR then involves an unrolled network—more precisely unrolled gradient descent network—that is designed to solve the modeled problem. We empirically assess the UVSR

Figure 2.12: A frame with complex motion super-resolved with a classical method (left) and the neural network FRVSR [110] (right).

Figure 2.13: Deep atmospheric turbulence mitigation.

performance through implementation, training on the MM522 dataset [136, 150] and testing on the Vid-4 dataset [77]. PSNR and SSIM are used for performance evaluation. Two experimental configurations are set up: one that involves a single degradation and another that involves multiple degradations. Both of them are noise-free. In the first one, UVSR is compared to the resembling FRVSR and three SOTA networks. Qualitative evaluation is also done in this single degradation configuration. In the second one, UVSR is compared to Frame-Recurrent Video Super-Resolution for Multiple Degradations (FRVSR-MD), an improved version of FRVSR that we design so that the network can deal with multiple degradations.

2.4.1 Problem formulation

We rely on the forward video formation model of Eq. (2.4). We further assume that H is a Gaussian blur with standard deviation σ. For the adjoint operator of $D_s H$, we use the operator $H^T B U_s$ where U_s upsamples the input by the factor s by inserting zeros and B replaces these zeros by bilinear interpolation. H is a Gaussian blur, so $H = H^T$. The adjoint operator of $F_{u_{t\to j}}$ is $F^T_{u_{t\to j}} = F_{u_{j\to t}}$.

Supposing H is known, we saw in Sec. 2.1 that the inverse problem is classically solved by alternately solving the minimization problems from Eqs. (2.5) and (2.6), relying on handcrafted priors on the estimated frame and optical flow.

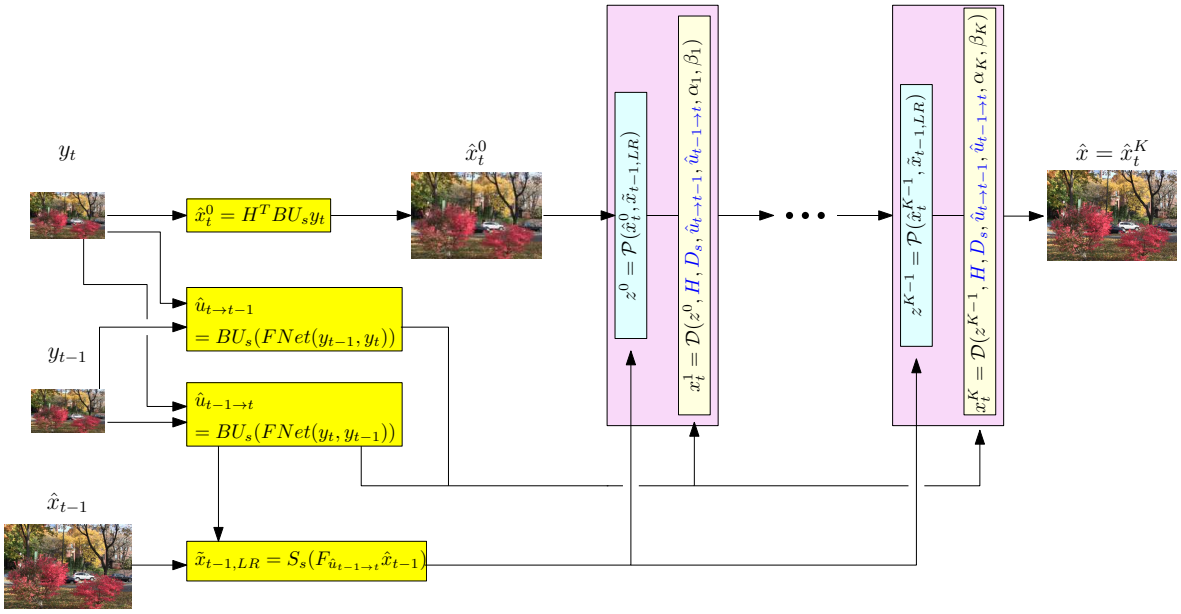


Figure 2.14: Illustration of UVSR. $\mathcal{P}$ and $\mathcal{D}$ respectively denote the prior and data steps.

2.4.2 UVSR Framework

Algorithm

If we want to recover the t-th HR frame by only using the t-th and $(t-1)$-th LR frames (taking example from FRVSR [110]), Eqs. (2.5) and (2.6) become:

$$\hat{x}_t = \arg\min_{x_t} ||D_s H x_t - y_t||_2^2 + ||D_s H F_{u_{t\to t-1}} x_t - y_{t-1}||_2^2 + \rho(x_t) \tag{2.16}$$

$$\hat{u}_{t\to t-1} = \arg\min_{u_{t\to t-1}} ||D_s H F_{u_{t\to t-1}} x_t - y_{t-1}||_2^2 + \varphi(u_{t\to t-1}) \tag{2.17}$$

The following outlines the idea of the framework we introduce: instead of alternately solving these problems, we propose first to compute $\hat{u}_{t\to t-1}$ and $\hat{u}_{t-1\to t}$ based on an optical flow estimation CNN called FNet, similarly to [110]. Then, to compute $\hat{x}_t$, we propose to unroll the gradient descent algorithm that solves Eq. (2.16) and replace the gradient of the prior by a CNN, similarly to [36]. We thus propose Algorithm 1, summarized in Fig. 2.14. In the algorithm, K denotes the number of unrolled iteration blocks, and α_k and β_k are trainable stepsize parameters of unrolled networks that are specific to each iteration block. We initialize them for the k-th iteration block with the following scheme: $(\alpha_k, \beta_k) = (\frac{1}{2^k}, \frac{1}{2^k})$. S_s and $\tilde{S}_s$ are respectively space-to-depth and depth-to-space [110, 117] transformations. They enable decreased computational cost by allowing convolution operations to be done in the LR image space. The notation $(\cdot, \cdot)$ denotes the concatenation operation.

The detailed workflow of Algorithm 1 is the following: the initial approximation of the super-resolved frame is obtained by backprojection (line 1 of Algorithm 1). In addition,

Algorithm 1: UVSR

Input: $y_t, y_{t-1}, \hat{x}_{t-1}, H, s$

1 *Initialization:* $\hat{x}_t^0 = H^T B U_s y_t$

2 $\hat{u}_{t\to t-1,LR} \leftarrow FNet(y_{t-1}, y_t)$

3 $\hat{u}_{t-1\to t,LR} \leftarrow FNet(y_t, y_{t-1})$

4 $\hat{u}_{t\to t-1} \leftarrow BU_s \hat{u}_{t\to t-1,LR}$

5 $\hat{u}_{t-1\to t} \leftarrow BU_s \hat{u}_{t-1\to t,LR}$

6 $\tilde{x}_{t-1} \leftarrow F_{\hat{u}_{t-1\to t}} \hat{x}_{t-1}$

7 $\tilde{x}_{t-1,LR} \leftarrow S_s(\tilde{x}_{t-1})$

8 **for** $k \leftarrow 0$ **to** $K-1$ **do**

Prior step:

9 $\hat{x}_{t,LR}^k \leftarrow S_s(\hat{x}_t^k)$

10 $z_{LR}^k \leftarrow N_{\theta_k}(\hat{x}_{t,LR}^k, \tilde{x}_{t-1,LR})$

11 $z^k \leftarrow \tilde{S}_s(z_{LR}^k)$

Data step:

12 $\hat{x}_t^{k+1} \leftarrow$
$\hat{x}_t^k + z^k - \alpha_k H^T B U_s (D_s H \hat{x}_t^k - y_t) - \beta_k F_{\hat{u}_{t-1\to t}} H^T B U_s (D_s H F_{\hat{u}_{t\to t-1}} \hat{x}_t^k - y_{t-1})$

13 **end**

14 $\hat{x}_t \leftarrow \hat{x}_t^K$

Output: $\hat{x}_t, \hat{u}_{t-1\to t,LR}, \hat{u}_{t\to t-1,LR}$

two inferences of FNet gives $\hat{u}_{t\to t-1,LR}$ and $\hat{u}_{t-1\to t,LR}$, the LR optical flow maps between the two consecutive LR frames y_{t-1} and y_t. These maps are then upsampled via bilinear interpolation which gives HR flow maps $\hat{u}_{t\to t-1}$ and $\hat{u}_{t-1\to t}$. The HR estimated of the previous frame x_{t-1} is then warped according to $\hat{u}_{t-1\to t}$ which produces $\tilde{x}_{t-1}$. The latter is then space-to-depth transformed which gives $\tilde{x}_{t-1,LR}$.

The next part of the algorithm constitutes the unrolling part. Each iteration step k outputs an estimation of the super-resolved current frame $\hat{x}_t^{k+1}$. This first involves the prior step (lines 9 to 11) in which the CNN $N_{\theta_k}(.)$ operates fusion and encodes statistical image prior simultaneously. The network takes as input the HR image estimated in the previous iteration step that is space-to-depth transformed $\hat{x}_{t,LR}^k$ and $\tilde{x}_{t-1,LR}$. Indeed, the prior image is drawn from a distribution with parameters θ conditioned by $\hat{x}_{t-1}$, which enforces temporal coherence. We also remark that our model is actually recurrent. After this prior step comes the data step (line 12). Solving Eq. (2.16) with the gradient descent unrolling algorithm involves two data consistency terms: one that is related to the current frame at t ($\alpha_k H^T B U_s (D_s H \hat{x}_t^k - y_t)$) and another term that is related to the previous frame at $t-1$ ($\beta_k F_{\hat{u}_{t-1\to t}} H^T B U_s (D_s H F_{\hat{u}_{t\to t-1}} \hat{x}_t^k - y_{t-1})$).

As in the data step z_k is reused, the network involves a "partial" residual connection, which facilitates the gradient flow. This skip connection is "partial" in the sense that the term $\hat{x}_{t-1}$ in the input that is concatenated is not involved in this connection. Space-to-depth and depth-to-space transformations are simple pixel rearrangement operations, so

this connection can be kept.

Architecture of the CNN

Considering Algorithm 1, for each of the iteration steps $k \in 0, ..., K-1$, with K being the number of total unrolled steps, the networks $N_{\theta_k}(.,.)$ has the same architecture. This architecture is similar to the one of VDSR [69], *i.e.*, is a feedforward CNN architecture with d layers, interspaced ReLU activations and the global skip connection. However, as written on Algorithm 1, here the input and output are in LR image space and there is the aforementioned partial skip connection. The choice of the two hyperparameters d and K, and the number of filters in each convolutional layer f significantly impact the inference speed and the number of parameters of UVSR. Our proposed method is similar to the FRVSR [110] that we consider as a baseline. For fair comparison and to demonstrate the benefit brought from the unrolled architecture, we choose d, K and f so that the number of parameters of UVSR becomes a bit comparable to the number of parameters of FRVSR. As we are dealing with a recurrent network, we also make sure these hyperparameters do not provoke any memory error regarding the backpropagation through time at training time. We choose $d = 7$, $K = 3$ and $f = 128$. Tab. 2.1 shows numbers of parameters of FRVSR and UVSR.

Architecture of FNet

We use the same architecture of FNet as in [110]. This contributes to a fair comparison between FRVSR and UVSR. Fig. A.1 in the Appendix. A.4 details this architecture.

Loss functions

We use the following loss function:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{sr} + \mathcal{L}_{flow_{t-1 \to t}} + \mathcal{L}_{flow_{t \to t-1}} \\ &= ||\hat{x}_t - x_t||_2^2 + ||F_{\hat{u}_{t-1 \to t, LR}} y_{t-1} - y_t||_2^2 + ||F_{\hat{u}_{t \to t-1, LR}} y_t - y_{t-1}||_2^2 \end{aligned} \tag{2.18}$$

The first term is related to the super-resolution task. The second and third terms account for the optical flow estimation from FNet. All losses are backpropagated through both FNet and the unrolled network as well as through time, and UVSR is end-to-end trained, as in [110].

2.4.3 Experiments

This section describes how we assess the UVSR performance via experiment.

Datasets

We use the MM522 dataset [136, 150] for training and the Vid-4 dataset [77] for testing. In the training phase, clips of 10 frames are randomly cropped with crop size 256×256 from the dataset. To generate LR frames, each HR video frame in the datasets is firstly blurred by a Gaussian kernel of standard deviation σ, then downsampled by sampling every 4-th pixel in each dimension (the sampling factor is $s = 4$), which generates LR video frames. No noise is added after. The variance of the Gaussian noise n_t in Eq. (2.1) is therefore zero. We set up two experimental configurations. In the first configuration, σ is fixed to 1.6 for both train and test sets, which enables to adopt similar experimental conditions as in [66, 150, 110]. We deal with a single degradation in this case. In the second configuration, for each sequence $\sigma = \sigma_{train}$ is uniformly sampled between 0.375 and 2.825 in the training phase. The value $\sigma = \sigma_{test} = 1.6$ is chosen for the test set. This allows us to assess a single UVSR under multiple degradations. Data augmentation by random flipping is operated during training.

Evaluation

We use a similar evaluation protocol as in [68, 66, 150] over the test set. PSNR and SSIM are computed over video pixels on the brightness channel of the ITU-R BT.601 YCbCr standard, excluding the first and last two frames and border pixels (8 pixels). We also compute the number of parameters and measure testing time cost as being the time needed to generate one 1920×1080 frame when the upscaling factor is 4. We perform experiments with an Intel I7-8700K CPU and one NVIDIA GTX 1080Ti GPU.

We compare our UVSR in the first single degradation configuration to FRVSR and the SOTA networks (DUF [66], PFNL [150], and RLSP [42]). In this experiment we do not consider EDVR [134] because of its too large size. For DUF and PFNL, we use values reported in [150] as performances of these networks were measured in the same setting. We implemented FRVSR and RLSP to measure their performances as the configurations in [110, 42] differ from ours.

For the second, multiple degradations configuration, we compare UVSR with an FRVSR variant that we improve in order to account for the multiple degradations configuration. We coin it Frame-Recurrent Video Super-Resolution for Multiple Degradations (FRVSR-MD). This model is derived from FRVSR, but the stretched feature maps that encode knowledge about σ are also concatenated at the input of SRNet, similarly to [157]. PCA is used to reduce the dimensionality of blur kernels to 10.

We also qualitatively evaluate UVSR based on generated predictions and temporal profiles.

Table 2.1: Number of parameters, testing time and PSNR(dB)/SSIM of different models on Vid4 test set with $\sigma = 1.6$. Values related to networks with '*' are taken from the referred publication. We implemented networks without '*'. Bold indicates the best performance.

Sequences	FRVSR	UVSR	*DUF [150]	*PFNL [150]	RLSP [42]
Calendar	23.90/0.8092	24.03/0.8102	23.85/0.8052	**24.37**/**0.8246**	24.08/0.8160
City	27.79/0.8220	27.59/0.8209	27.97/0.8253	28.09/**0.8385**	**28.10**/0.8278
Foliage	26.53/0.7806	**26.58**/**0.7819**	26.22/0.7646	26.51/0.7768	26.43/0.7766
Walk	29.98/0.9029	30.03/0.9035	30.47/0.9118	**30.65**/**0.9135**	30.36/0.9101
Average	27.05/0.8287	27.06/0.8291	27.13/0.8267	**27.40**/**0.8384**	27.24/0.8326
# param. (M)	5.055	4.624	5.829	**3.003**	4.225
Testing time (ms)	214	170	2754	741	**80**

2.4.4 Results

Tab. 2.1 summarizes comparisons between UVSR and other networks under single degradation. We observe that in terms of PSNR, on average UVSR does not significantly improve over FRVSR and performs worse than the SOTA networks. However, for the *foliage* sequence, UVSR performs the best. Regarding SSIM, on average UVSR performs better than DUF and FRVSR and worse than RLSP and PFNL. Here also, UVSR performs the best for the *foliage* sequence. We note that the PSNR and SSIM values of FRVSR that we implemented are higher than those reported in [150] that adopted a similar experimental configuration. Thus we make a fair comparison.

Then, UVSR presents faster inference than FRVSR and the SOTA networks except RLSP. Regarding model complexity, UVSR presents fewer parameters than FRVSR and DUF, a similar number of parameters as RLSP but more parameters than PFNL.

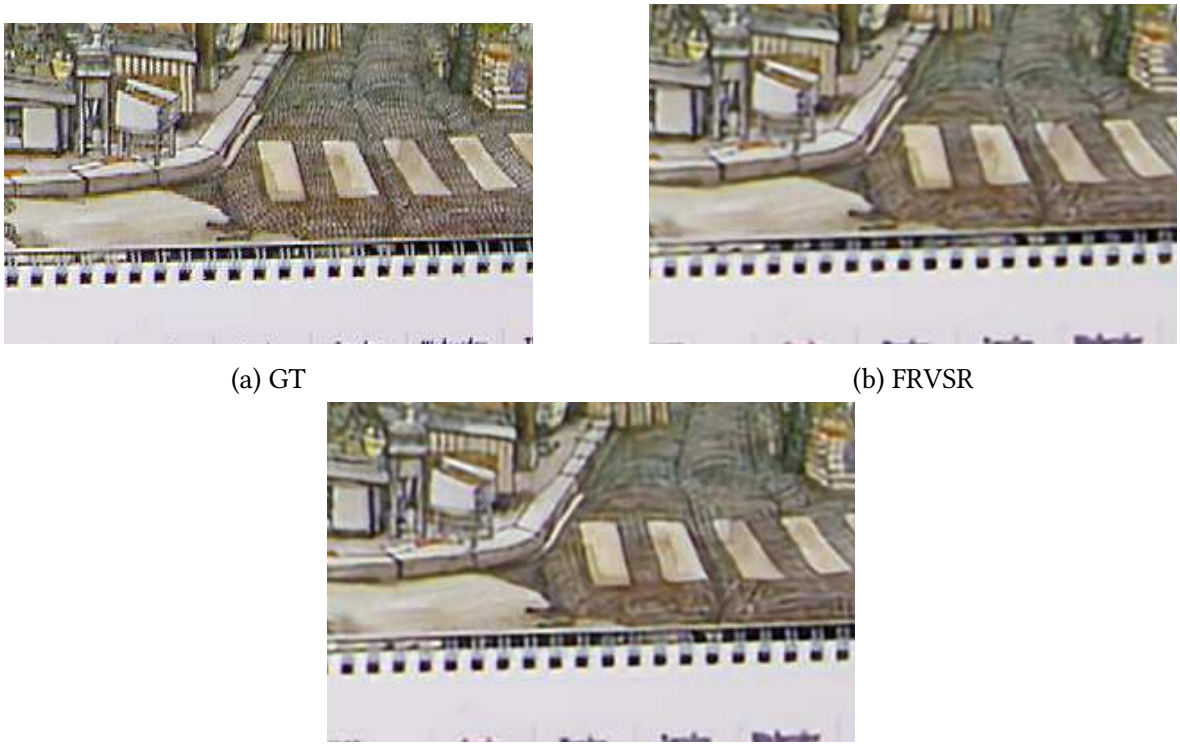
Tab. 2.2 compares UVSR and FRVSR-MD under multiple degradations. One can see that UVSR has fewer parameters than and is faster than FRVSR-MD. Concerning PSNR, UVSR does not improve over FRVSR-MD. But UVSR outperforms FRVSR-MD concerning SSIM.

From the above observations, the situation UVSR is the most adapted seems to be in the presence of multiple degradations with constraints on model size and inference latency. Moreover, UVSR rather presents more satisfactory performance with respect to SSIM than PSNR. We recall that SSIM is a better measure of perceived visual quality than PSNR [138]. The following observation is in the same vein. Indeed, Fig. 2.15 visually compares UVSR and FRVSR under single degradation. One can see that for the frame from the *calendar* sequence, UVSR more sharply estimates the clips and the road.

Fig. 2.16 shows temporal profiles for this *calendar* sequence. We observe that UVSR produces sharper results than FRVSR. This indicates that UVSR enables better temporal coherence. We think this is due to the fusion and the presence of a motion-related data consistency term at each unrolled iteration step.

Table 2.2: Number of parameters, testing time and PSNR(dB)/SSIM of FRVSR-MD and UVSR on Vid4 test set with $\sigma_{train} \in [0.375, 2.825]$ and $\sigma_{test} = 1.6$. Bold indicates the best performance.

Sequences	FRVSR-MD	UVSR
Calendar	23.52/0.7870	**23.62**/**0.7947**
City	**27.64**/0.8093	27.60/**0.8099**
Foliage	**26.17**/0.7596	26.05/**0.7614**
Walk	29.68/0.8971	**29.74**/**0.8982**
Average	**26.75**/0.8132	**26.75**/**0.8161**
# param. (M)	5.066	**4.624**
Testing time (ms)	216	**170**



(a) GT (b) FRVSR

(c) UVSR

Figure 2.15: Visual comparison on the *calendar* sequence from Vid4.

Figure 2.16: Temporal profiles on the *Calendar* sequence. (a-c) are respectively GT, FRVSR and UVSR.

Increasing the number of unrolled steps and/or the number of convolutional filters in each prior step should improve UVSR's performance. However, these increases are limited by the memory available at training time and the authorized inference speed.

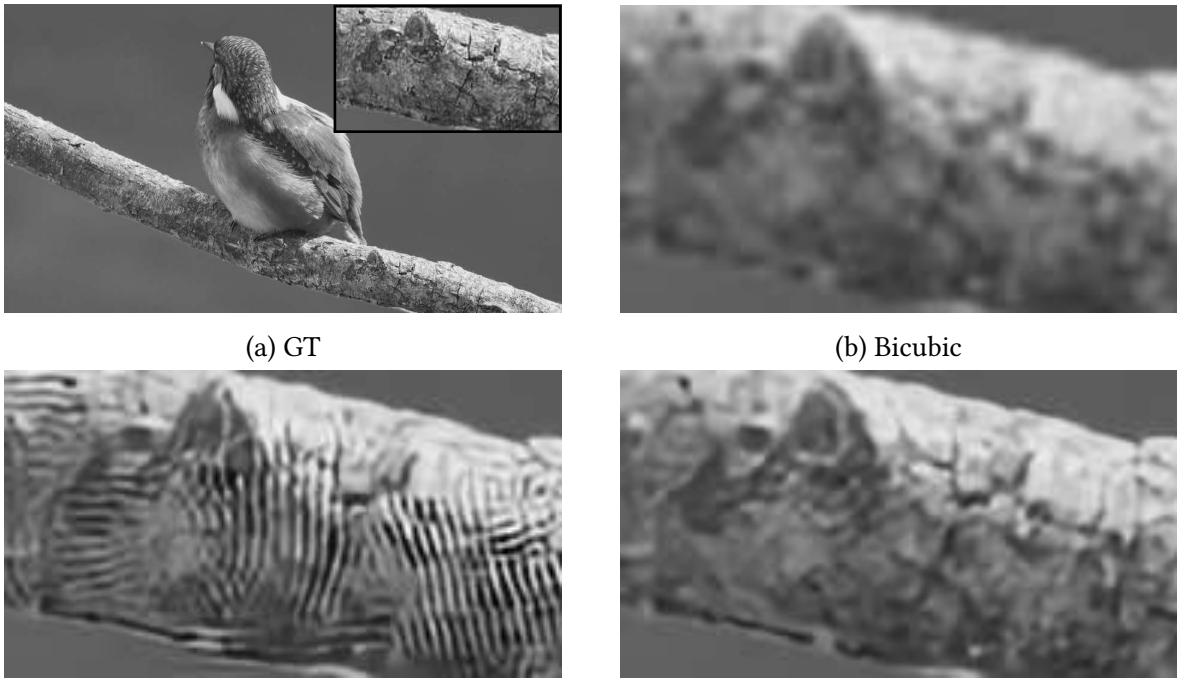
2.5 Stable Long-term Recurrent video super resolution

As stated in Sec. 2.2.1, recurrent models, like FRVSR, UVSR and RLSP, have gained popularity in deep VSR due to their increased computational efficiency, temporal receptive field and temporal consistency compared to sliding-window based models. However, in this section, we show that when inferring on long video sequences presenting low motion (*i.e.*, in which some parts of the scene barely move), recurrent models diverge through recurrent processing, generating high frequency artifacts. To the best of our knowledge, no study about VSR pointed out this instability problem, which can be critical for some real-world applications. Video surveillance is a typical example where such artifacts would occur, as both the camera and the scene stay static for a long time.

In the following, we expose the instabilities of existing recurrent VSR networks on long sequences with low motion. We demonstrate it on a new long sequence dataset Quasi-Static Video Set, that we have created. Finally, we introduce a new framework of recurrent VSR networks that is both stable and competitive, based on Lipschitz stability theory. We propose a new recurrent VSR network, coined Middle Recurrent Video Super-Resolution (MRVSR), based on this framework. We empirically show its competitive performance on long sequences with low motion.

2.5.1 Cause of the divergence

Because of computational and memory constraints, as well as vanishing and exploding gradients, recurrent VSR models are usually trained on sequences of 7 to 12 images. They are then deployed to super-resolve a sequence of any length. Some applications, such as video-surveillance, would require to super-resolve sequences of arbitrary length. However,



(a) GT (b) Bicubic

(c) RLSP (d) MRVSR (Ours)

Figure 2.17: A comparison between a state-of-the-art recurrent VSR network (RLSP) and our proposed network. The former generates high frequency artifacts on long sequences with low motion. The proposed network does not.

recurrent models are not trained on these long sequences. Hence, there is no guarantee that they optimally perform on long sequences. In this section (Sec. 2.5), we show that recurrent VSR networks generate high frequency artifacts when inferring on long video sequences presenting low motion. These are sequences that contain parts of the scene that barely move, for instance because the camera is quasi-static. The super-resolution process creates high-frequency information which is accumulated in the long-term recurrence, creating artifacts and causing divergence. Fig. 2.17 illustrates this phenomenon. To the best of our knowledge, our work is the first study about VSR that raises this instability issue. This unexpected behavior can be critical for some real-world applications, like video surveillance in which both the camera and the scene stay static for a long time.

The structure of the rest of this section is the following. First, we review studies related to VSR and instabilities of recurrent networks. Then, based on Lipschitz stability theory, we propose a new framework of recurrent VSR network that is both stable and competitive on long sequences with low motion. After this, we introduce a new recurrent VSR network MRVSR as an implementation of this framework. Finally, we empirically analyze instabilities of existing recurrent VSR models on long sequences with low motion and show the stability and superior performance of the proposed network. A new long sequence dataset has been created for our experiments. We make it publicly available.

2.5.2 Instabilities of recurrent neural networks

Recurrent Neural Networks (RNNs) are difficult to train [98]. First of all, they involve back-propagation through time (BPTT), *i.e.*, their unrolling through time, that is costly in terms of memory. Secondly, these architectures risk vanishing and exploding gradients issues. Related to this, RNNs are prone to divergence when inferring on long sequences. Authors of [87] showed, in the context of multi-layer and LSTM networks, that an RNN is stable if its Lipschitz constant is smaller than 1. To enforce this constraint, they proposed to clip singular values of the matrix associated with the recurrence map to 1. Several works circumvent vanishing and exploding gradients problems by setting all the singular values to 1 [6, 144, 86, 132, 67, 155].

Some studies are related to enforcing the Lipschitz constraint in the context of convolutional neural networks. Authors of [116] proposed to clip singular values of the block matrix of doubly block-circulant matrices associated with the convolutional layer. The work [89] explored *spectral normalization*, that relies on the power iteration to estimate maximal singular value of the reshaped kernel tensor of the convolutional layer. Authors of [131, 50] suggested not using this reshaping and instead proposed to directly use the kernel tensor in the power iteration. Finally, the work [111] proposed Spectral Rank Normalization (SRN), an algorithm that seeks to enforce either the Lipschitz constraint or its softer version.

In the context of recurrent video denoising, authors of [123] pointed out instabilities. They first brought out unforeseeable, colorful and black mask-like artifacts in long-term video denoising. Then, inspired by studies on adversarial examples [48], they proposed a diagnosis tool to check stability of a trained recurrent video processing network. Finally, they improved upon the SRN algorithm to propose *Spectral Rank Normalization for Convolutional layer* (SRN-C). While SRN reshapes the kernel tensor of the convolutional layer, SRN-C avoids this reshaping, similarly to [131, 50]. They applied this method on convolutional layers of their recurrent video denoising network and demonstrated its effectiveness.

To conclude this section, the following points summarize the limits of existing works regarding long-term recurrent VSR and our contributions:

- existing recurrent VSR networks have been only evaluated on relatively short generic sequences. Their performances have not been measured on long sequences. We demonstrate these networks perform poorly on such sequences when the motion amplitude is low, due to their recurrent structure. We create a novel dataset of long and low motion sequences, because existing datasets only contain sequences that either are too short or present fast scene motion;

- the relationship between instabilities and scene motion in video has not been investigated. We show that when inferring on long sequences presenting low motion, existing recurrent VSR models diverge;

- the Lipschitz constraint has not been applied on existing recurrent VSR networks. Indeed, in order to have a stable recurrent VSR network, we could first take one of these networks and directly apply a Lipschitz constraint to all convolutional layers

in the recurrent loop. We show that this strategy fails when super-resolving long sequences with low motion;

- we design a recurrent VSR framework that is stable on long sequences with low motion, while not being globally Lipschitz constrained. We demonstrate the superior performance of a network based on this framework.

2.5.3 Method

Stability of recurrent video processing models

A recurrent video processing model is determined by a *recurrence map* $\phi^L : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ and an *output map* $\psi : \mathbb{R}^n \to \mathbb{R}^c$. The recurrent information $h_t \in \mathbb{R}^n$ and the output image $\hat{x}_t \in \mathbb{R}^c$ are updated at each time step t as follows:

$$\begin{cases} h_t = \phi^L(h_{t-1}, y_t) \\ \hat{x}_t = \psi(h_t) \end{cases} \tag{2.19}$$

where $y_t \in [0,1]^d$ is an input image provided at time t.

The recurrent model is *Lipschitz stable* if ϕ^L is *contractive* in h, *i.e.*, if ϕ^L is L-Lipschitz in h with $L < 1$ (the superscript in ϕ^L highlights this Lipschitz continuity). L is the Lipschitz constant of ϕ^L. This stability ensures that the full recurrent system is globally stable when running the network an arbitrary number of times, avoiding any divergence. Assume that ϕ^L is composed of K convolutional layers interspaced with ReLU nonlinearities. Each convolutional layer can be encoded by a weight matrix, obtained from the layer's kernel tensor as a block matrix of doubly block-circulant matrices. Because Lipschitz constant of the ReLU activation is 1, L is upper-bounded by the product of the spectral norms of the weight matrices of the convolutional layers:

Proposition 1 *For a recurrent model ϕ^L constituted of K convolutional layers with weight matrices $W_1, ..., W_K \in \mathbb{R}^{n \times n}$ interspaced with ReLU nonlinearities, the Lipschitz constant L of ϕ^L verifies:*

$$L \leq \prod_{k=1}^{K} ||W_k||_{\text{sp}} \tag{2.20}$$

where $||.||_{\text{sp}}$ is the spectral norm.

Given this inequality, the Lipschitz stability can be ensured under the hard Lipschitz constraint:

Constraint 1 ***Hard Lipschitz constraint (HL)***

$\forall k \in [\![1, K]\!]$, we impose $||W_k||_{\text{sp}} \leq 1$.

However, the upper bound in Eq. (2.20) mostly overestimates L. As an illustration, if ϕ^L is constituted of 2 convolutional layers with weight matrices W_1 and W_2, the only case where $L = ||W_1||_{\text{sp}} \cdot ||W_2||_{\text{sp}}$ is when the first right singular vector of W_1 and the first left singular vector of W_2 are aligned. Hence, the constraint is overly restrictive. One can thus decide to relax it, leading to the soft Lipschitz constraint:

Constraint 2 ***Soft Lipschitz constraint (SL)***

$\forall k \in [\![1, K]\!]$, *we set* $||W_k||_{\text{sp}} = \alpha > 1$ *and minimize srank*(W_k) *based on training data, where srank is the Stable rank.*

Stable rank is an approximation of the rank operator that is stable under small perturbations of the matrix. This soft constraint does not theoretically guarantee the Lipschitz stability, so it is important to empirically verify the non divergence.

To enforce these constraints in the context of convolutional neural networks, *Stable Rank Normalization for Convolutional layers* (SRN-C) can be applied to a convolutional layer during the training stage. This sets the spectral norm of the matrix of this layer to a desired value α and minimizes the stable rank of the matrix during training, controlled by β. α and β are among hyperparameters of the algorithm. When $\beta = 1$, it is equivalent to performing spectral normalization on the matrix. After training, a normalization step is required just before test time, so the algorithm does not introduce any overhead in runtime and model size at inference time. Appendix B.2 details the SRN-C algorithm.

Unconstrained Stable Recurrent VSR framework

In approaches such as RLSP, FRVSR and RSDN, every convolutional layer of super-resolving networks is recurrent within feedback loops. This seeks to increase the depth and width of the recurrent connection by giving the hidden state and the previous output to the input of super-resolving networks. Therefore, these layers both incorporate past information and contribute to the deconvolution task. Adopting the notations from Eq. (2.19), in these networks ψ is reduced to the identity mapping (followed by pixel shuffling or transposed convolutions). In order to have a stable recurrent VSR network, a naive approach would be to directly apply SRN-C to one of these VSR networks. However, this approach presents some difficulties.

First, we applied SRN-C to RLSP with $(\alpha, \beta) = (2.0, 0.1)$ and empirically verified that SL was not capable of removing the artifacts on long sequences (Fig. 2.20d). Second, we did the same experiment with $(\alpha, \beta) = (1.0, 1.0)$ to enforce HL and this resulted in a stable network but with poor VSR performance (detailed in Sec. 2.5.5). This is because the resulting architecture has been constrained to be globally 1-Lipschitz, and a successful super-resolving function—that operates both upsampling and deconvolution—cannot be 1-Lipschitz; since some frequencies need to be boosted as the Wiener filter does in the optimal linear case. This is not the case for a denoising function, that can be 1-Lipschitz while correctly performing.

Considering these points, we define a new framework of recurrent VSR network that is stable and performs competitively on long sequences:

Definition 1 *An **Unconstrained Stable Recurrent VSR** network is defined by an input network $\xi : [0,1]^{d\times(2T+1)} \to \mathbb{R}^d$, a contractive recurrent network $\phi^L : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ and an output network $\psi : \mathbb{R}^n \to \mathbb{R}^c$. The features z_t, the hidden state h_t and the output image $\hat{x}_t$ are updated at each time step t as follows:*

$$\begin{cases} z_t = \xi(Y_t) \\ h_t = \phi^L(h_{t-1}, z_t) \\ \hat{x}_t = \psi(h_t) \end{cases} \tag{2.21}$$

where $Y_t = \{y_t\}_{t-T\leq t\leq t+T} \in [0,1]^{d\times(2T+1)}$ is an input batch of LR images provided to the network at t and $2T+1$ denotes the size of the batch.

Let ϕ^L be constituted of K convolutional layers with weight matrices $W_1, ..., W_K \in \mathbb{R}^{n\times n}$ interspaced with ReLU activations. ϕ^L is contractive in h based on the hard Lipschitz constraint: $\forall k \in [\![1, K]\!], ||W_k||_{\text{sp}} \leq 1$.

Stable: all the layers in the inner recurrent loop of such a network are contractive, which guarantees its stability over time.

Unconstrained: such a network is not globally constrained in terms of Lipschitz continuity, due to its non contractive input and output networks which can keep their full expressiveness.

Most of the deconvolution task is done by ξ and ψ. ϕ^L incorporates past information. When ξ and ψ are simultaneously identity mappings, the *unconstrained* property is lost, as the network becomes globally 1-Lipschitz. This is the case encountered when imposing HL on all convolutional layers of networks such as RLSP, FRVSR and RSDN.

Middle Recurrent Video Super-Resolution

As an implementation of the proposed framework, we design a new network coined **Middle Recurrent Video Super-Resolution** (MRVSR). Its architecture is illustrated in Fig. 2.18. The first part of the network, ξ, has a feed-forward architecture with n_ξ convolutional layers and interspaced ReLU activations. The second part ϕ^L is composed of $n_\phi + 1$ convolutional layers under HL and interspaced ReLU activations. The third part ψ has a feed-forward architecture with n_ψ convolutional layers interlaced with ReLU activations and followed by a pixel shuffling layer. This part takes as input the current hidden state h_t and the hidden state from the previous time step. This mecanism, called *feature-shifting*, is helpful to promote temporal consistency between two successively output frames.

Incorporating past information via the recurrent connection is a simpler task than deconvolution. This can be illustrated revisiting the traditional, non DL based Shift-and-Add

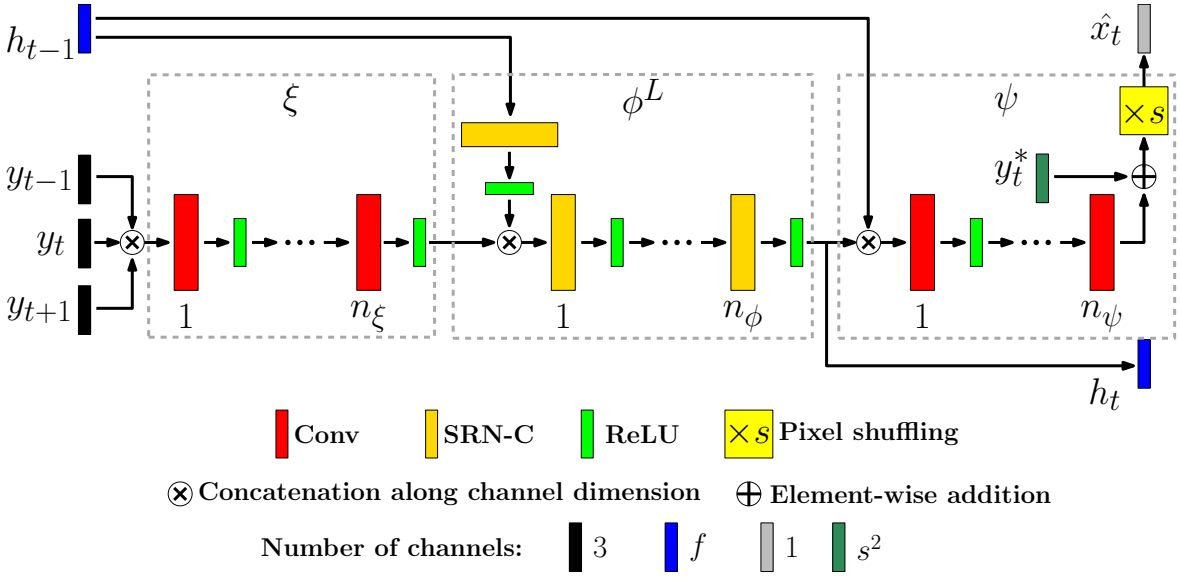


Figure 2.18: MRVSR architecture. SRN-C denotes convolutional layer under HL enforced by SRN-C. Each convolutional layer uses 3×3 kernel with stride 1 and outputs f feature maps ($f = 128$ in our study), except the last one which outputs $s^2 = 16$ feature maps, where s is the scaling factor. The network outputs the brightness channel Y of YCbCr color space. Cb and Cr channels are upsampled independently with bicubic interpolation. Input LR frames $\{y_i\}_{t-1 \le i \le t+1}$ are in RGB colorspace. Besides, y_t is converted from RGB to Y and replicated $s^2 = 16$ times in the channel dimension, which gives $x_t^\star$ for the residual connection. Pixel shuffling rearranges elements in a tensor of shape $(C \times s^2, H, W)$ to a tensor of shape $(C, H \times s, W \times s)$.

agorithm [40]. In this method, historical information is captured via averaging or median aggregating past frames after projection on a HR grid and motion compensation. Averaging or median aggregating are rather simple mathematical operations. Therefore, n_ϕ can be smaller than $n_\xi + n_\psi$. In practice, one can fix $n_\xi + n_\phi + n_\psi$ to satisfy some constraint on computational cost, set a small value for n_ϕ and then select n_ξ and n_ψ. In our setting, we have found that under the condition $n_\xi + n_\phi + n_\psi = 7$ (that enables both fast computations and good performance), the value $n_\phi = 1$ lead to the best performance among other values of n_ϕ on our validation set (described in Sec. 2.5.4).

2.5.4 Experiments

Networks

For comparison, we implement the following state-of-the-art recurrent VSR networks in Pytorch [99]: FRVSR 10-128 [110], RSDN 9-128 [62] and RLSP 7-128 [42]. The numbers after each network respectively indicate the number of repeated building blocks and the number of filters in each convolutional layer. These hyperparameters enable reasonably fast training and testing and satisfactory performance on short sequences. In the following, we omit these numbers for simplicity. For RSDN, our implementation is based on the official codes released by its authors.[2] Additionally, we implement modified RLSP where all its layers have been normalized by SRN-C with hyperparameter sets $(\alpha, \beta) = (2.0, 0.1)$ and $(\alpha, \beta) = (1.0, 1.0)$ to enforce the soft and hard Lipschitz constraints respectively. We call these networks RLSP-SL and RLSP-HL.

We compare these networks against the proposed MRVSR. We select (n_ξ, n_ϕ, n_ψ) so that $n_\xi + n_\phi + n_\psi = 7$ for the reason stated in Sec. 2.5.3. This number equals the number of convolutional layers in RLSP (excluding the layer that processes the hidden state), which yields fair comparison. Among MRVSR with different sets (n_ξ, n_ϕ, n_ψ), the network with $(n_\xi, n_\phi, n_\psi) = (3, 1, 3)$ was the best performing model on our validation set. Therefore, in Sec. 2.5.5 we only report performances recorded by MRVSR with this hyperparameter set. We use SRN-C with $(\alpha, \beta) = (1.0, 1.0)$ to impose the HL.

In order to measure the benefit from constrained recurrence map, we also implement MRVSR without its recurrence and feature-shifting, which coincides with RLSP without its recurrence. This can be seen as an extension of SISR that takes 3 consecutive LR frames as an input at each time step. Its architecture is feed-forward with 7 convolutional layers with interlaced ReLU activations. We call this network RFS3 for **R**esidual **F**usion **S**huffle network with **3** input frames. This network will serve as baseline against recurrent models. In addition, we also implement RFS with an input batch of 7 LR frames, that we call RFS7. This serves as a representative sliding-window based model to compare against MRVSR, because most of sliding-window based VSR models take a batch of 5 to 7 LR frames.

[2] https://github.com/junpan19/RSDN

Datasets

We prepare the training dataset in a similar way as in [42]. From the 37 high resolution Vimeo videos that were used in this study, after downsampling them by a factor of 2 we extract 40,000 random cropped sequences of size $I \times 256 \times 256 \times 3$, where $I \geq 12$. The delimiting keyframes are excluded from the sequence. At training time, we sample random sub-sequences of these crops with length 12. By excluding the first and the last frames, we obtain GT sequences with length 10. The first and last frames of the sampled sequences are used to produce y_{-1} at the beginning and y_{10} at the end. Data augmentation (random flip/transposition) is also employed.

We also prepare a validation set of 4 sequences. They come from videos with no constraints on motions of objects and count between 30 and 50 frames each.

We introduce a new test set of long sequences in which the camera is quasi-static and foreground objects move. This dataset will be complementary to the existing datasets (Vid4 [77], REDS [91] and Vimeo-90K [146]) which contain only videos that either are short, or present fast scene motion. To generate this new dataset, we download videos from `vimeo.com` and `youtube.com` and extract 4 sequences with quasi-static scene and moving objects inside. The first two of them are respectively Full HD and HD Ready and the two others are 4K. The HD and 4K sequences are downsampled respectively by a factor of 2 and 4. These 4 sequences respectively have the following lengths in number of frames: 379, 379, 379 and 172. They constitute the test dataset we call **Quasi-Static Video Set**. We limited the lengths of the sequences to 379 to ensure dataset homogeneity, but the video containing the first sequence contains a much larger number of frames. Therefore, we have also prepared a longer version of the first sequence called *Sequence 1-XL*. This sequence contains 8782 frames. All of these sequences are available on https://github.com/bjmch/MRVSR.

The train and validation sets contain standard, relatively short sequences with no constraints on motion, whereas the test set contains long sequences with low motion. It aims at testing the capability of networks trained on short sequences to work on real-life long sequences that may have low-motion periods. We remind the reader that training recurrent networks on such long sequences is not realistic for reasons explained in Sec. 2.5.1, so the generalization gap between short and long sequences cannot be addressed with training data.

We additionally compare the reconstruction performances on the standard Vid4 dataset.

From each of the training, validation and test sequences in HR space, the corresponding LR sequence is generated by applying gaussian blur with σ and sampling every $s = 4$ pixel in both spatial dimensions. We set $\sigma = 1.5$, except when testing RSDN. In the case of this network, we use the pre-trained weights available on its official github repository. We thus adapted the codes of the corresponding degradations that are available on this repository to generate the LR sequence and the value of $\sigma = 1.6$ was used.

Training procedure and evaluation

All of the networks we prepare are trained from scratch after the Xavier initialization [46], except RSDN. The loss function is pixel-wise mean-squared-error between pixels in the brightness channel Y of YCbCr color space of GT frames and the network's output. The networks are trained with Adam optimizer [70] and a batch size of 4. The learning rate starts at 10^{-4} and is divided by 10 after the 200th and 400th epochs. RFS3, RFS7 and MRVSR are trained for 600 epochs. Other models except RSDN are trained between 400 and 600 epochs until convergence, based on train and validation losses.

We numerically evaluate the networks based on frame PSNR and SSIM. Qualitative evaluation that checks the presence of artifacts is of equal importance. We also assess the temporal consistency by examining temporal profiles from output sequences.

Moreover, the diagnosis tool from [123] can be used in order to visualize Spatio-Temporal Receptive Field (STRF) of a recurrent network. This tool, that is inspired by studies on adversarial examples [48], works as follows: given a trained recurrent video processing network, it looks for an input sequence $Y = (y_{-\tau}, ..., y_{\tau})$ that is optimized to maximize the response at the center pixel in the output sequence $X = (x_{-\tau+1}, ..., x_{\tau-1})$. To do so, the L1 norm of the center pixel $|p|$ in x_0 is maximized. This optimization only affects pixels in Y that have an effect on p. Therefore, the optimized sequence Y can be interpreted as a visualization of the STRF for the pixel p. τ is typically set to 40, values of pixels in Y are randomly initialized between 0 and 1 and images in Y have dimensions $64 \times 64 \times 3$. In our experiment, the optimization is solved using gradient descent and Adam optimizer for 1500 iterations. The learning rate starts at 1 and is divided by 10 after 750 and 1250 iterations.

2.5.5 Results

Performance of existing recurrent networks

Fig. 2.19 shows the evolution of the PSNR per frame for some of the networks, averaged over the first three sequences of Quasi-Static Video Set. The curve of RFS3 is taken as a baseline and subtracted to the other ones, and the resulting curves are displayed. We see that until a relatively small number of processed frames, existing recurrent networks (RLSP, RSDN and FRVSR) perform optimally and remain better than the baseline model. But at a certain point their performance drop and they become worse than the baseline model, indicating that the recursion integrates harmful information at each new frame. This can be seen as divergence.

Tab. 2.3 summarizes the performances of the networks on the Quasi-Static Video Set. It summarizes the performances of the methods at the beginning of the sequences, through the entire sequences, and at the end of the sequences. The table conforms with the curves shown on Fig. 2.19. Based on reported performances, at the beginning of the sequences RLSP and RSDN perform better than the baseline RFS3. However, at the end of the sequences these networks and FRVSR have diverged and perform worse than RFS3. The differences in performance on the last 50 reconstructed frames between RFS3 and respec-

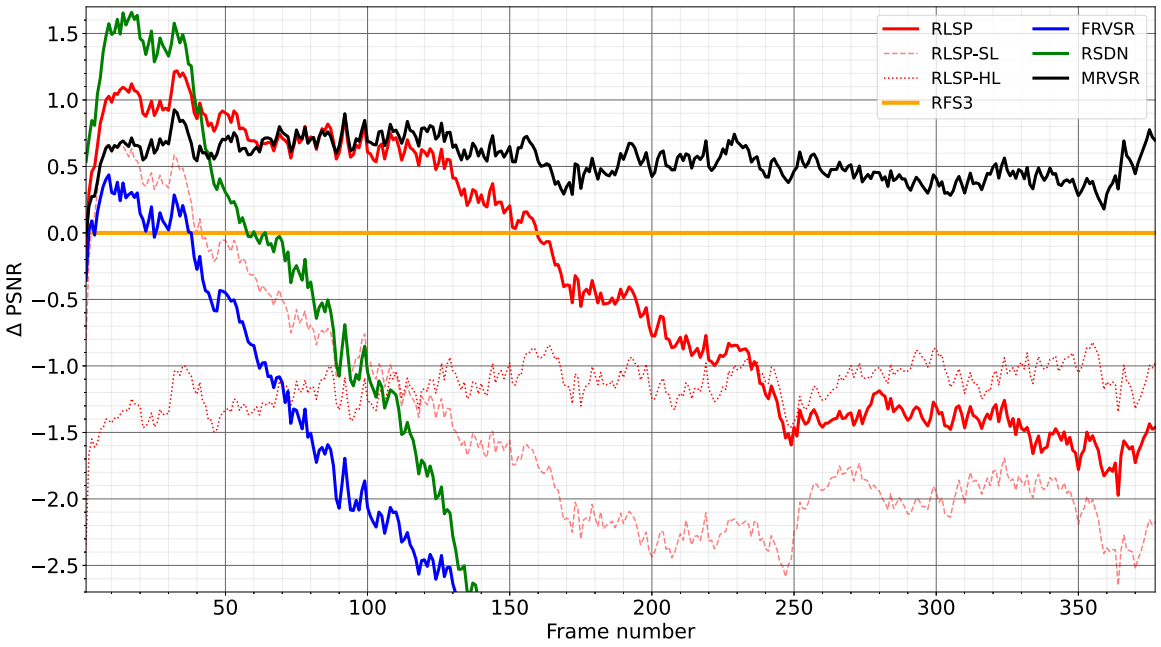
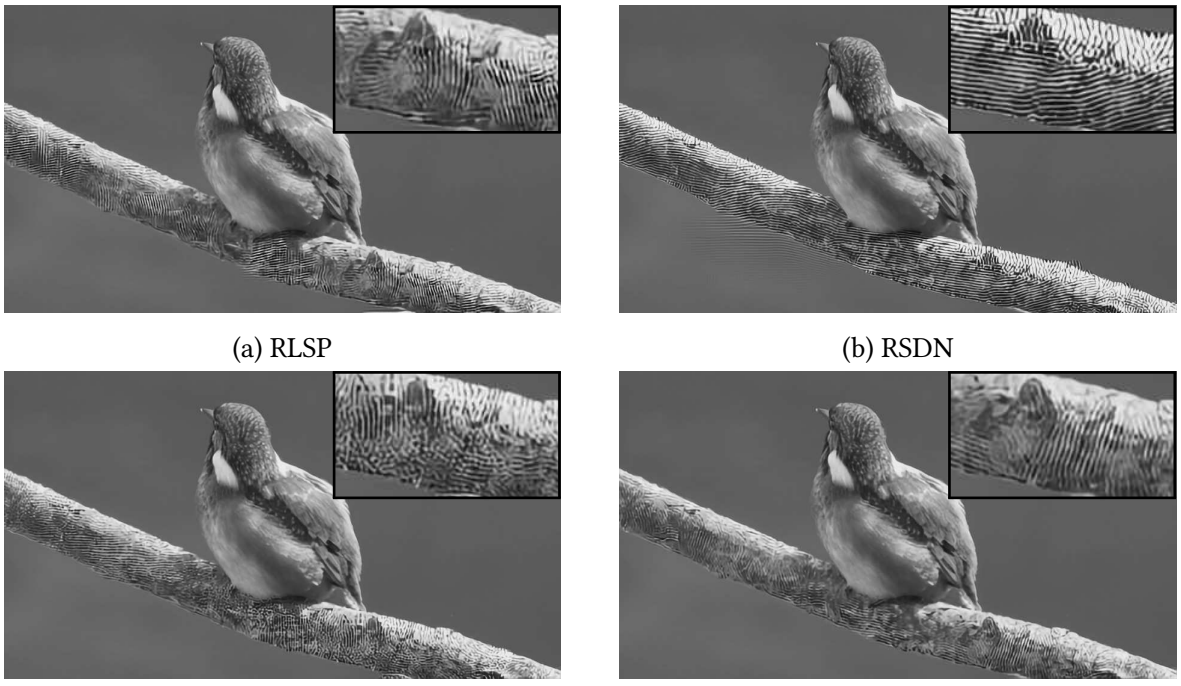


Figure 2.19: Evolution of PSNR on the brightness channel per frame averaged over the first three sequences of the Quasi-Static Video Set. We substract the curve of the RFS3 baseline and the graph shows these differences.



(a) RLSP

(b) RSDN

(c) FRVSR

(d) RLSP-SL

Figure 2.20: A frame near the end of the first sequence of Quasi-Static Video Set (the 376th frame) reconstructed from state-of-the art recurrent networks, and RLSP-SL. The brightness channel is visualized. The networks generate high frequency artifacts on the branch, which is a quasi-static object.

Model	First 50	All	Last 50
Bicubic	30.08 / 0.8362	30.05 / 0.8356	30.11 / 0.8387
RFS3	32.20 / 0.8911	32.04 / 0.8886	32.07 / 0.8911
RFS7	32.38 / 0.8945	32.23 / 0.8921	32.26 / 0.8943
FRVSR	32.15 / 0.8947	29.16 / 0.8442	27.68 / 0.8121
RSDN	33.46 / 0.9181	29.82 / 0.8788	27.98 / 0.8549
RLSP	33.08 / 0.9099	31.67 / 0.8964	30.57 / 0.8882
RLSP-SL	32.45 / 0.8991	30.62 / 0.8708	29.98 / 0.8627
RLSP-HL	30.98 / 0.8618	30.91 / 0.8608	30.95 / 0.8630
MRVSR	32.80 / 0.9030	32.62 / 0.9007	32.62 / 0.9026

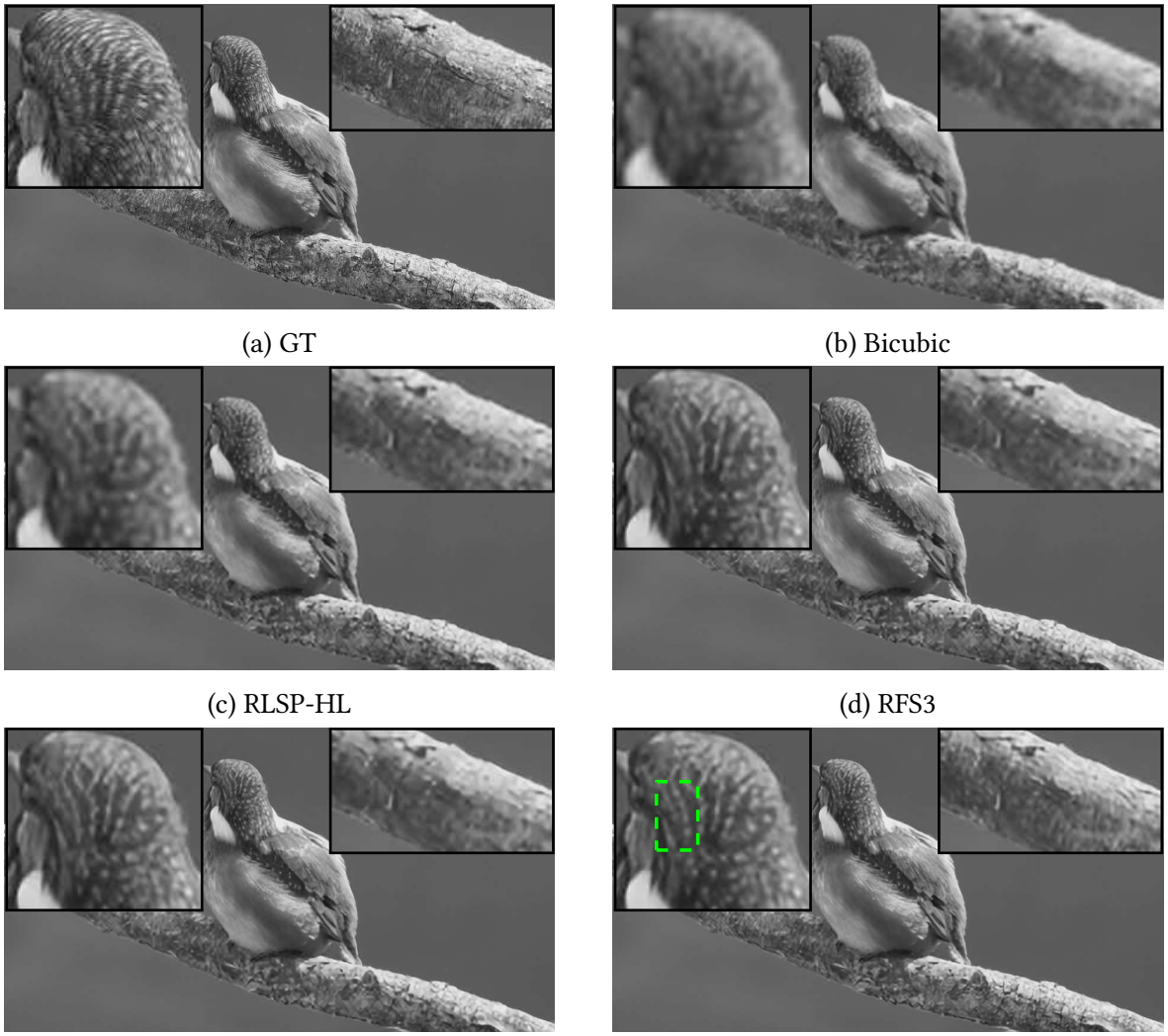
Table 2.3: Mean PSNR / SSIM on the brightness channel of Quasi-Static Video Set. The metrics are measured excluding the first 3 and last 3 GT frames. 'First 50' means the metrics are computed at the beginning of the sequences, *i.e.*, on the first 50 reconstructed frames. 'All' means the metrics are computed through the entire sequences, *i.e.*, on all reconstructed frames. 'Last 50' means the metrics are computed at the end of the sequences, *i.e.*, on the last 50 reconstructed frames. Red: the best result. Blue: the second best result.

tively RLSP, FRVSR and RSDN are -1.50, -4.39 and -4.09 in PSNR and -0.0029, -0.0790 and -0.0362 in SSIM. They represent in average -3.33dB in PSNR and -0.0394 in SSIM. This performance drop is due to the generation and accumulation of high frequency artifacts. These artifacts appear on objects that barely move. Example artifacts are shown on Figs. 2.20a to 2.20c which show a frame near the end of the first sequence of Quasi-Static Video Set (the 376th frame) reconstructed by each network.

Behavior analysis: These existing recurrent networks are trained to optimize their performance on a very low number of frames (at most 10). In this setting, it is beneficial to the network to produce rapidly a huge amount of details in the output sequence. These high frequency details grow in strength with time, but they are not fed back into the network more than 10 times, so the optimization process is not trained to manage their increase after this period. When inferring on long sequences, these details keep accumulating long after the short-term network's training regime, which produces visible artifacts that diverge over time. In the presence of strong motion, even with short-term training, the network learns to forget the past information (this forgetting capability of recurrent VSR networks is empirically demonstrated in Appendix B.1), which is inconsistent with the new one. The newly created high frequency content is forgotten at the same time, preventing divergence on scenes with enough motion. In the first sequence of the Quasi-Static Video Set, the bird moves regularly, which is why artifacts do not have time to appear on the bird itself, as can be seen on Fig. 2.20.

Constraining existing recurrent networks

SL: RLSP-SL faces the same issues as existing recurrent networks. After being better than the baseline RFS3 at the beginning of the sequences, it diverges (Fig. 2.19). It generates high



(a) GT (b) Bicubic

(c) RLSP-HL (d) RFS3

(e) RFS7 (f) MRVSR

Figure 2.21: The 376th frame of the first sequence of Quasi-Static Video Set, reconstructed from methods that are stable by design (non recurrent or under HL). MRVSR presents the best quality.

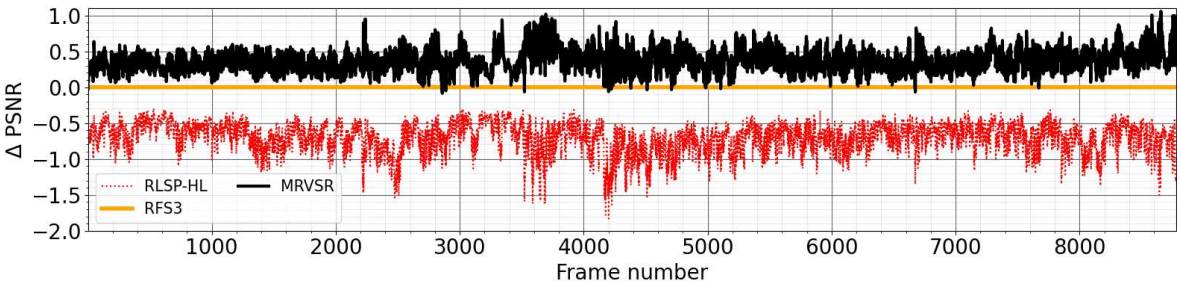


Figure 2.22: Evolution of PSNR on the brightness channel per frame on *Sequence 1-XL*. We substract the curve of the RFS3 baseline and the graph shows these differences.

frequency artifacts (Fig. 2.20d) and its performance at the end of the sequences is poor, as shown in Tab. 2.3 (-2.09dB in mean PSNR and -0.0284 in mean SSIM compared to RFS3 on the last 50 reconstructions). This proves that SL is not enough to prevent the divergence.

HL: RLSP-HL also obtains an overall poor performance (-1.13dB in average PSNR and -0.0278 in average SSIM compared to RFS3 based on all reconstructed frames, according to Tab. 2.3). Its reconstruction performance is stable on a long sequence (Figs. 2.19 and 2.22), but the reconstructed image is blurred (Fig. 2.21c). This is because RLSP-HL is globally constrained to be 1-Lipschitz. Thus, as stated in Sec. 2.5.3, it is poorly suited to the deconvolution task.

Performance of the proposed network

Model	RFS3	FRVSR	RSDN	RLSP	MRVSR
PSNR	26.43	26.69	27.92	27.46	26.90
# Param. (M)	0.77	5.05	6.18	1.08	1.21
Runtime (ms)	9	55	56	11	12

Table 2.4: Mean PSNR on the brightness channel of Vid4, model size and runtime. PSNR values for FRVSR, RLSP and RSDN are taken from their papers. Runtime is measured on an LR size of 180×320, an Intel I9-10940X CPU and one NVIDIA TITAN RTX GPU.

At the beginning of the quasi static sequences (Fig. 2.19 and Tab. 2.3) MRVSR cannot match RLSP and RSDN, but performs better than the baseline RFS3 and FRVSR. This performance is compatible with the results on Vid4 (Tab. 2.4), where MRVSR is 0.56dB behind the unconstrained similar network RLSP. This is due to the Lipschitz constraint on MRVSR, built to ensure its long-term stability at the price of a lower short-term performance.

When considering long-term performance on sequences with low motion, MRVSR gives the best results. Figs. 2.19, 2.22 and 2.21f show that MRVSR does not diverge and does not generate any artifact. According to Tab. 2.3, MRVSR achieves the best mean performance on the test set, based on all reconstructed frames as well as focusing on the last 50 reconstructed frames. Because MRVSR and RFS3 take the same number of input frames—namely three—the differences of $+0.58$ dB in average PSNR and $+0.0121$ in average SSIM computed on all reconstructed frames represent the benefit brought by the contractive recurrence map of MRVSR. Moreover, considering that RFS7 takes an input batch of 7 frames, the fact that MRVSR outperforms RFS7 ($+0.39$dB in average PSNR and $+0.0086$ in average SSIM) shows that the temporal receptive field enabled by its contractive recurrence accounts for more than 7 frames. This is confirmed in Fig. 2.23, where the temporal receptive field of MRVSR spans around 28 frames, which is much larger than the usual length (*i.e.*, 7) of temporal receptive field of sliding-window based models. Moreover, temporal profiles produced by MRVSR are less noisy and sharper than the ones produced by RFS3 and RFS7. This shows the contractive recurrence map of MRVSR additionally enables increased temporal consistency. Visually speaking, sequences generated by MRVSR present less flickering artifacts than sequences produced by RFS7 and RFS3. Fig. 2.24 displays examples of temporal profiles

for the first sequence of Quasi-Static Video Set. Finally, MRVSR presents the best long-term reconstruction in terms of visual quality. Some examples can be observed in Fig. 2.21.

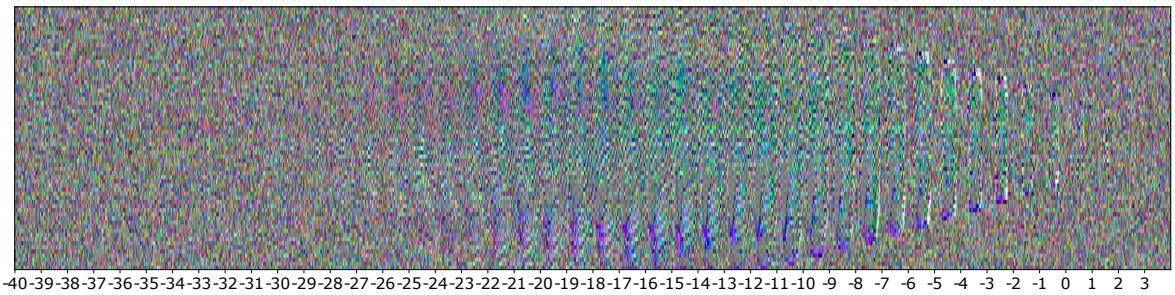
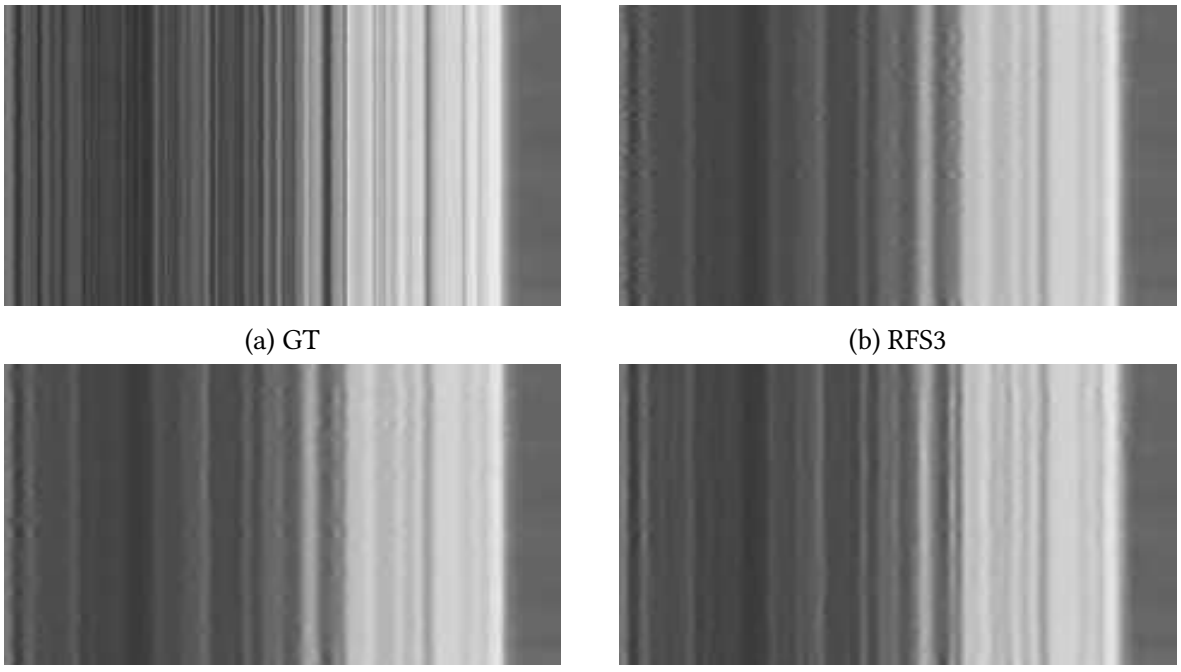


Figure 2.23: Spatio-temporal receptive fields of MRVSR (vizualization of juxtaposed images in the input sequence $Y = (y_{-\tau}, ..., y_{\tau})$ optimized to maximize the L1 norm of the center pixel in the output image x_0). The horizontal axis accounts for the time index t of y_t. The figure is stretched in vertical direction.



(a) GT

(b) RFS3

(c) RFS7

(d) MRVSR

Figure 2.24: Temporal profiles from the brightness channel of the first sequence of Quasi-Static Video Set. We take the 256th horizontal row of all images and stack them vertically.

As one could expect, MRVSR has practically the same computational complexity compared to RLSP (similar runtime and slight overhead in number of parameters, according to Tab. 2.4). As we stated in Sec. 2.2.2, RLSP is known to be the fastest VSR network so far. Therefore, MRVSR presents state-of-the-art runtime and compact model size.

Fig. 2.25 shows the SVD spectrum of our MRVSR model. We see that SRN-C successfully works in constraining the spectral norm of only recurrent layers of ϕ^L to 1. Other layers

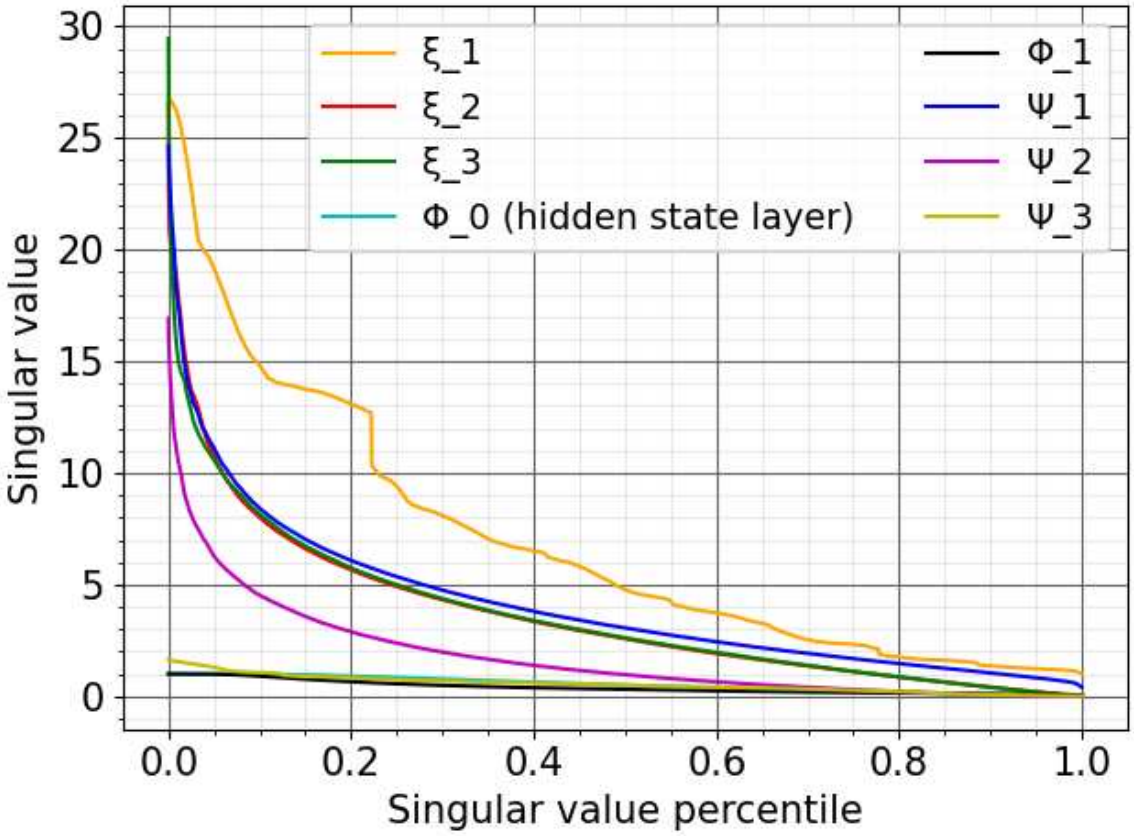


Figure 2.25: SVD spectrum of MRVSR, based on the code from [116]. Each label in the legend indicates the n-th layer in one of the sub-networks ξ, ϕ^L or ψ. We see that SRN-C successfully works in constraining the spectral norm of only recurrent layers of ψ^L to 1.

have spectral norms that are higher than 1 and can fully contribute to the deconvolution task.

Finally, in Appendix. B.3 we propose a simple way to diagnose whether a trained recurrent VSR model is stable on sequences with low motion or not. We also present additional experimental results.

2.5.6 Discussion

Given an existing pre-trained recurrent model, one way to operate it in a stable manner would be the following: chunking the long sequence into shorter, either overlapping or not overlapping sequences and using the model for each chunk. The recurrent features are reset to zero between each chunk. In our study, we do not consider this method for the following reasons. If the chunks do not overlap, this approach results in a severe visual flickering artifact between each pair of chunks, due to the reinitialization of recurrent features that makes the model go through a new burn-in period. If the chunks overlap, on the one hand, this still gives visual flickering artifacts due to discontinuities of recurrent features. On the other hand, the method becomes computationally redundant and inefficient, suppressing one of the main advantages of using a recurrent model. Computation time is doubled at overlapped regions and memory consumption is doubled.

2.6 Conclusion

In the first part of this chapter, we explored VSR methods. The first paradigm regroups model-based methods that rely on hand-crafted regularization and iterative algorithms. They are generally slow but under some assumptions on motion they can be made fast. The second paradigm regroups DL-based methods. They can learn complex spatio-temporal statistics of natural videos based on supervised training. Their success is at the mercy of effective feature/image implicit or explicit alignment and feature fusion. We compared these paradigms based on examples of more general video restoration tasks. When the motion is simple and known, it is easy to come up with a forward video formation model that correctly describes the encountered situation and in this case model-based methods perform very well. In other, more general cases, data-based methods are better suited as they can manage complex motion to some extent.

Then, we introduced UVSR, a VSR framework that blends classical and DL-based approaches, based on deep unrolling of gradient descent algorithm. UVSR, in contrast to purely DL-based VSR methods, can incorporate prior knowledge about image degradation model and enables a better interpretability of the role of CNNs based on its correspondence to classical iterative algorithms. We compared UVSR with FRVSR, three SOTA networks and FRVSR-MD under single/multiple degradation configurations, considering PSNR and SSIM over the test set, number of parameters, inference speed, visual evaluation and temporal coherence. Empirical evaluation confirms that the situation UVSR is the most adapted is when there are multiple degradations with constraints on inference speed and number of parameters.

Finally, we have pointed out the divergence problem of recurrent VSR when facing long sequences with low motion. Existing recurrent VSR networks generate high-frequency artifacts on such sequences. To solve this issue, we defined a new framework of recurrent VSR model, based on Lipschitz stability theory. This method is more adapted for long sequences with low motion compared to existing recurrent VSR networks. As an implementation of this framework, we proposed a new recurrent VSR network coined MRVSR. We experimentally verified its stability and state-of-the-art performance on long sequences with low motion. As part of our experiments, we introduced a new test dataset of such sequences, namely Quasi-Static Video Set.

Chapter 3

Deconvolution for interferometric Radio Transient Reconstruction

In this chapter, we explain how we can apply the techniques described in Chap. 1 to solve the following inverse problem in astrophysics: reconstruction of astronomical sources that evolve with time, in the context of radio interferometry. In particular, we deal with transient sources. These sources appear and disappear over time and are associated with high-energy physical phenomena. While the VSR problem investigated in Chap. 2 models the PSF induced by the camera lens, the radio interferometry inverse problem involves the PSF of the observing instrument, *i.e.*, the radio interferometer. In VSR, we supposed the camera

lens PSF was constant over time. Due to the Earth's rotation, in radio interferometry, an observer on the ground will experience the rotation of the apparent sky over the instrument. Consequently, an interferometer tracking a source during the observation will have a time-dependent PSF due to line of sight projection.

3.1 Introduction

Next-generation radio facilities like LOFAR [52], MeerKAT/SKA [14], ASKAP/SKA [35] and SKA-LOW [112] allow for high spectral, high rate, improved angular resolutions, and high instantaneous sensitivity. This is a notable improvement for studying transient radio sources via aperture synthesis using radio interferometers. These sources appear and disappear over time and can be randomly distributed in the sky. They are associated with high-energy physical phenomena (*e.g.*, pulsars, rotating radio transients (RRATs), solar-system magnetized objects, and Lorimer-type bursts [80]) and, more generally, "fast radio bursts" (FRB). Searching for such sources in large datasets produced by these instruments is a new challenge that requires competitive and efficient signal reconstruction algorithms.

Radio interferometers enable imaging via aperture synthesis based on processing the correlations between each pair of antenna signals. In the first approximation, a radio interferometer samples noisy Fourier components of the sky (associated with its spatial frequencies, *i.e.*, the visibilities [143]) inside the main field of view of the instrument. Under the small field approximation assumption, the sky can be approximated by computing the inverse Fourier Transform of those Fourier samples. The number of baselines is limited, so the Fourier map is incomplete. Therefore, one needs to solve an "inpainting" [43] inverse problem, *i.e.*, estimate lacking information in the Fourier plane. Another option is to switch from the inpainting problem in the visibility space to its equivalent deconvolution problem in the image space. Deconvolution of radio images from a "static" sky in the context of radio interferometry has been subject to studies for several decades. Notably, the authors of [56] designed the original CLEAN algorithm, which is still the most widespread basis for newer deconvolution algorithms in the community. Several variants of this algorithm have been subsequently proposed [115, 30]. Improvements taking into account source morphology [1], spectral dependencies [102] and sparse representations [43, 92, 45, 32, 20, 21, 141] have also been investigated in recent years.

When the sky contains transient sources, classical detection methods rely on frame-by-frame image analysis (*e.g.*, with the LOFAR Transient Pipeline [122]). However, frame-by-frame transient detection is subject to two observing biases: i) a detection issue when the frames are derived from too short time integration data displaying a high noise level and, conversely, ii) a "dilution" problem when time integration is too long to resolve the transient in time, resulting in a time smearing of the transient. Therefore, to account for these biases and sources that possess coherent structure in time, one has to design methods that directly account for the time-coherent structure (*i.e.*, the source light curve) hidden in the signal. This also occurs in large radio surveys with interferometers. Mapping the visible radio sky requires an optimal use of the observing time and pointing location to reach a target angular resolution and sensitivity. While surveys mainly address the distribution of static sources (astrometry and flux density) transient radio sources might also occur during

the short exposure (*e.g.*, ∼15 min pointing of MeerKAT) and be missed due to observing biases (detection and dilution). Therefore, finding a robust deconvolution method is key to optimize both telescope time and transient detectability.

3.2 Deconvolution in radio interferometry

Two approaches have been classically used in radio interferometry deconvolution: the Variational Maximum Entropy method [41, 94, 140] and the iterative CLEAN algorithm [56]. Both methods generally perform well when dealing solely with point sources, but CLEAN is the most widespread technique in the radio astronomy community. This algorithm supposes a finite number of point sources. It restores them based on Matching Pursuit [9] using a single basis vector and the impulse response (PSF) of the telescope that made the observation. Authors of [30] proposed a variant of CLEAN by optimizing the algorithm with Fast Fourier Transform (FFT) and structuring the algorithm computations between "major" and "minor" cycles. Minor cycles are carried out in (gridded) image space, whereas major cycles befall in the ungridded visibility space. Going back and forth between these two spaces led to improvements in both fidelity and accuracy. This strategy was further developed in [115]. Authors of [1] and [102] brought in further improvements by respectively taking into account the morphological and spectral behavior of the sources. More recently, several teams have addressed the deconvolution problem within the compressed sensing framework [43, 45, 32, 20, 21, 141] then DL [125] ([113] also used DL but solved the equivalent inpainting problem in the Fourier space). However, few methods take into account temporal structures of time-evolving sources in the context of image time series deconvolution.

3.3 Interferometry imaging problem

3.3.1 Single image deconvolution problem

This study deals with imaging by aperture synthesis from interferometric data. The limited number of antennas and observing baselines, time, and frequencies restrict the amount of accessible samples of the sky visibility function. In addition, these visibilities are subject to a noise that can be modeled as an additive Gaussian noise in the first approximation. In a limited field of view and ignoring direction-independent or direction-dependent effects and calibration issues, this ill-posed inverse problem can be expressed in the Fourier space (*i.e.*, the measurement space) as follows:

$$V_y = M(V + \epsilon) \tag{3.1}$$

where V_y is the collection of observed visibilities, V is the true visibility function, ϵ is approximated as an additive white gaussian noise, and M is a sampling mask representing the limited access of an interferometer to the measurement (depending on the antennas' configuration and observational parameters).

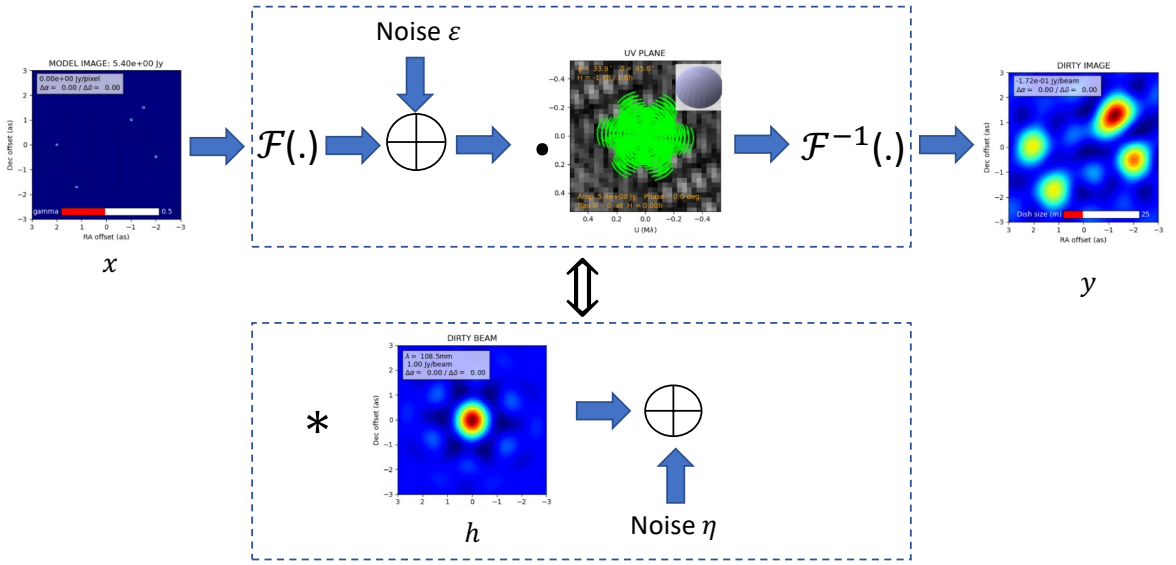


Figure 3.1: Principle of radio interferometry by aperture synthesis. Images have been generated thanks to APSYNSIM[82]. We simulated a VLA array configuration and 4 gaussian sources, with a total observation duration of 4h.

Eq. (3.1) can be rewritten as a deconvolution problem formulated in the image or direct space. This problem links the observed degraded sky image to the corresponding true sky image:

$$y = h * x + \eta \tag{3.2}$$

with $\mathcal{F}^{-1}(V_y) = y$, $\mathcal{F}^{-1}(M) = h$, $\mathcal{F}^{-1}(V) = x$ and $\mathcal{F}^{-1}(M) * \mathcal{F}^{-1}(\epsilon) = \eta$. $*$ denotes the convolution operation; y is the observed image, called *dirty image*; h is the PSF, also called *dirty beam*, which represents the sampling operation in Fourier space by the interferometer; and x is the GT image. We note σ_ϵ the noise level of ϵ. Thus, the corresponding variance for η can be obtained from $\sigma = ||M||_2 \cdot \sigma_\epsilon = ||\mathcal{F}(h)||_2 \cdot \sigma_\epsilon$. Fig. 3.1 illustrates the inverse problem expressed in both Fourier and direct domains.

3.3.2 Extension to transient imaging

To enable robust imaging of transient sources, instead of solving frame by frame— which can be subject to observation bias—we extend the problem in (3.2) to a deconvolution problem accounting for the temporal dependency of the different terms (*i.e.*, sky, noise, and instrument sampling):

$$y_t = h_t * x_t + \eta_t, \qquad t \in I = \{t_0, \dots, t_{T-1}\} \tag{3.3}$$

with $\mathcal{F}^{-1}(V_{y,t}) = y_t$, $\mathcal{F}^{-1}(M_t) = h_t$, $\mathcal{F}^{-1}(V_t) = x_t$ and $\mathcal{F}^{-1}(M_t) * \mathcal{F}^{-1}(\epsilon_t) = \eta_t$. The noise level of η_t is $\sigma_t = ||M_t||_2 \cdot \sigma_\epsilon = ||\mathcal{F}(h_t)||_2 \cdot \sigma_\epsilon$. By stacking $\{y_t\}_{t\in I}$, $\{x_t\}_{t\in I}$ and

$\{h_t\}_{t\in I}$ in the temporal dimension, we respectively obtain a *dirty cube*, a GT cube and a PSF cube. These cubes are 3-dimensional data structures denoted Y, X and H respectively. I is the set of T time steps ordered following the observation intervals. In the single image problem (3.2), h depends on the total time and frequency integration of observation, while the sky x is supposed to be static. As the sky naturally rotates over the instrument during the observation (because of the Earth's rotation), the morphology of h depends on the interferometer location on Earth, declination of the source, and observing dates. In the dynamic imaging problem extension (3.3), the h_t operator has a time dependency, *i.e.*, the instrumental response varies for consecutive single observing time intervals. As a result, both the sky and the interferometer responses vary over time. The associated mask M_t samples the Fourier transform of the sky at different time dates, enabling the possibility of capturing the temporal evolution of the observed sky.

We assumed that the datacube X only contains point-like sources for simplification. Indeed, we assume each source has an angular scale much smaller than the angular resolution brought by the PSF. In addition, we assume that the cube contains mixtures of sources with constant and varying flux densities over time. Their locations in the sky will be random but constant during the observation.

Because a radio interferometric dataset provides exact information on the baseline length and orientation for all samples, the morphology and time dependency of h_t are known for all t. Finally, we control the noise level σ_ϵ to mimic the various quality of observations. Thus, following the terminology defined in Sec. 1.3, we are in a multiple degradation scenario.

In this study, for the sake of simplification, we focus on the monochromatic case where the observing frequency is fixed. We therefore do not deal with the dependency in frequency of the imaging problem.

3.4 Method

3.4.1 CLEAN

CLEAN supposes the sky is constituted of a finite set of point sources. This is indeed the image prior. The algorithm restores these points and their intensities based on Matching Pursuit [9]. The set of these points is convolved with a smooth kernel called CLEAN/restoring beam constructed from the PSF. Generally the CLEAN beam corresponds to a 2D Gaussian function adjusted to the PSF. The form of the solution is then:

$$\hat{x} = G * a + r \tag{3.4}$$

where G is the CLEAN beam, a is the reconstructed sky with points restored by CLEAN, also called CLEAN components, and r is the residual. The original CLEAN algorithm is described in Algorithm 2.

Algorithm 2: CLEAN algorithm [56]. $h_{p,q}$ is the image of the PSF h centered on the position (p, q). $\delta_{p,q}$ is a sky with the point source of amplitude 1 at the position (p, q).

Input: y, h
Parameter: Gain $g \in]0, 1]$, N_σ, N_{max}
Output: The solution $\hat{x}$ and the sky model a
Initialization: $r^0 \leftarrow y$, $a^0 \leftarrow 0$, $k \leftarrow 0$, $\sigma =$ noise level estimated from y, $G =$ 2D Gaussian function adjusted to h.

1: **while** $max(r^k) > N_\sigma \sigma$ or $k < N_{max}$ **do**
2: $\quad f_m \leftarrow max(r^k)$
3: $\quad (p, q) \leftarrow \arg\max(r^k)$
4: $\quad r^{k+1} \leftarrow r^k - h_{p,q} \cdot f_m \cdot g$
5: $\quad a^{k+1} \leftarrow a^k + \delta_{p,q} \cdot f_m \cdot g$
6: $\quad k \leftarrow k + 1$
7: **end while**
8: **return** $\hat{x} = G * a^k + r^k, a = a^k$

3.4.2 Proposed DL-based methods

In the context of the time series image deconvolution problem (3.3), one can independently apply a single image deconvolution method, such as CLEAN, in a frame-by-frame approach. However, this method does not capture the temporal structure of the sky. A method capable of dealing with the temporal evolution of both telescope and the sky is required. Given recent successes of neural networks in various restoration tasks, we propose to solve the problem (3.3) based on DL. In our setting, a network realizes the following mapping: $\hat{X} = N(Y, H, \sigma_\epsilon; \theta)$. Its input contains a degraded cube and information about the degradation (H, σ_ϵ) in a non-blind manner. θ denotes parameters of the network that are learned from training data $\{Y_i, X_i, H_i, \sigma_{\epsilon,i}\}$. We propose two implementations of the network N, coined 2D-1D Net and Deflation Net, respectively.

Multiple degradations

We adopt the following scheme to incorporate knowledge about the image formation model and handle multiple degradations. Each h_t in H is originally of size $l \times l$ (with $l = 256$ in our study; see Sec. 3.5) but is firstly center-cropped with crop size $r \times r$ ($r = 96$ in our study) and projected onto a b-dimensional linear space by a PCA projection matrix $P \in \mathbb{R}^{b \times r^2}$. P is learned from all PSFs that constitute the training PSF cubes, and with a value of $b = 50$ we can explain 90 % of the total variance. We note $\vec{h}_t^b$ this projected PSF. $\vec{h}_t^b$ is then concatenated with $\sigma_t = ||M_t||_2 \cdot \sigma_\epsilon = ||\mathcal{F}(h_t)||_2 \cdot \sigma_\epsilon$. We denote this vector $\vec{h}_t$.

2D-1D Net

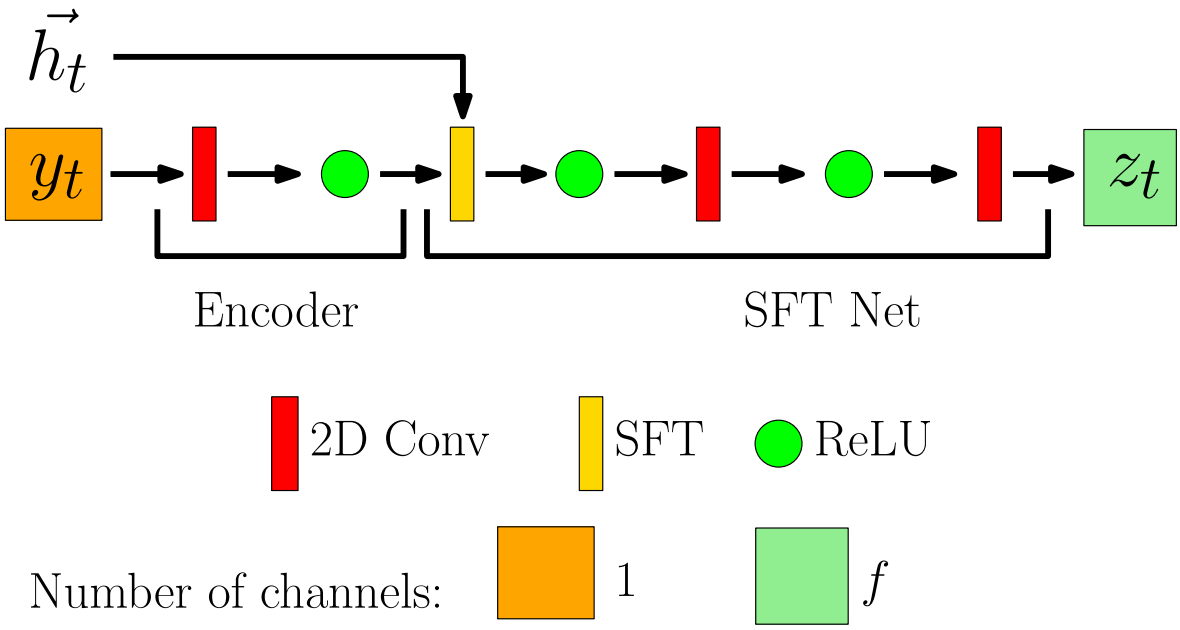


Figure 3.2: 2D Net. Each convolutional layer outputs f feature maps. $f = 32$ in our study. The kernel size of each convolutional layer is set to 3×3. The vector entering the SFT layer indicates $\vec{h}_t$.

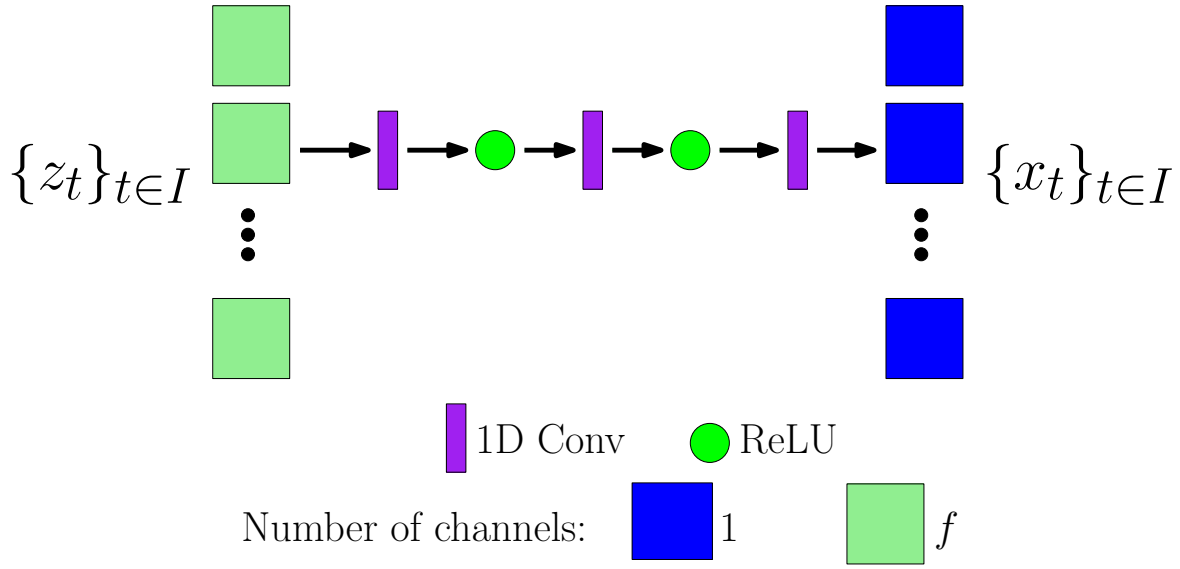


Figure 3.3: 1D Net. $\{z_t\}_{t\in I}$ has dimensions $C \times T \times H \times W$. Each convolutional layer outputs f feature maps, except the last one which outputs images with 1 channel each. Each 1D convolutional layer has the kernel size 5.

We first propose a new network coined 2D-1D Net to solve the transient imaging problem of Eq. (3.3). The following details the idea behind this model. We decouple the network N into two modules that sequentially process the input data. The first module is a network

Algorithm 3: 2D-1D Net. C_r denotes the operation that center-crops a 2D structure with crop size r.

Input: $(Y, H, \sigma_\epsilon) = (\{y_t\}_{t\in I}, \{h_t\}_{t\in I}, \sigma_\epsilon)$
Output: $\hat{X}$
1: $S \leftarrow$ an empty list
2: **for all** $t \in [|0, T-1|]$ **do**
3: $\quad \vec{h}_t^b \leftarrow PC_r h_t$
4: $\quad \sigma_t = ||M_t||_2 \cdot \sigma_\epsilon = ||\mathcal{F}(h_t)||_2 \cdot \sigma_\epsilon$
5: $\quad \vec{h}_t \leftarrow \text{Concat}(\vec{h}_t^b, \sigma_t)$
6: $\quad f_t \leftarrow \text{Encoder}(y_t)$
7: $\quad z_t \leftarrow \text{SFT Net}(f_t, \vec{h}_t)$
8: $\quad S.\text{append}(z_t)$
9: **end for**
10: $Z \leftarrow \text{StackAlongTimeAxis}(S)$
11: $\hat{X} \leftarrow \text{1D Net}(Z)$
12: **return** $\hat{X}$

with 2D convolutional layers that successively and independently encode each image in the degraded cube into feature maps. Each of these encodings considers the PSF and the noise level used in the degradation. This transformation performs a deconvolution of the degraded image. This module is referred to as 2D Net. After these independent deconvolutions, the produced feature maps are stacked in an extra temporal dimension and given to the second module. This module captures temporal structures within the stacked maps and estimates the GT cube. The temporal profile[1] of a source is continuous in time, which justifies this architectural choice. Because point sources do not spatially move in our study, the temporal structure is extracted based on 1D convolutional layers along the time dimension. We call this module 1D Net. By unifying the two modules, we build the entire network 2D-1D Net.

Algorithm 3 summarizes the 2D-1D Net. In the first place, each pair $(y_t, \vec{h}_t)$ is given to 2D Net to produce an intermediate feature map z_t (lines 2 to 8). Fig. 3.2 describes the structure of this subnetwork. This network is composed of an encoder that extracts input features and SFT Net that can manage multiple degradations. In this step, 2D Net deconvolves y_t by using $\vec{h}_t$. The SFT layer uses this vector to modulate feature maps. This layer has been introduced in [135] and used in [51] for the first time to handle multiple degradations in inverse problems. This layer applies an affine transformation to the feature maps F_{in} conditioned on the degradation maps $F_t^{(h)}$, which is obtained by stretching $\vec{h}_t$ into size $(b+1) \times H \times W$, where all the elements of the i-th map equal the i-th element of $\vec{h}_t$. The affine transformation involves scaling and shifting operations:

$$\text{SFT}(F_{in}, F_t^{(h)}) = \gamma \odot F_{in} + \beta \tag{3.5}$$

[1]This means the evolution through time of the amplitude of a source and has nothing to do with the "temporal profile" used in the previous chapter regarding the VSR problem.

where γ and β are estimated by additional convolutional layers and $\odot$ is the Hadamard product.

Next, $\{z_t\}_{t\in I}$ are stacked in the temporal dimension. This gives Z, a tensor of dimensions $C \times T \times H \times W$ (line 10). The output is passed to 1D Net, composed of three 1D convolutional layers working along the temporal dimension, interlaced with ReLU activations (line 11). The kernel size of each 1D convolutional layer is set to 5. With three such layers, the temporal receptive field is $5+4+4=13$, which is enough considering the temporal extension of the experimental transient events in this work (see Sec. 3.5.1). Fig. 3.3 details the 1D Net architecture.

Deflation Net

One can observe that, on the one hand, averaging the input images over the temporal dimension produces an average input sky with a reduced noise level. This average sky keeps constant sources. On the other hand, averaging the PSFs over the temporal dimension produces an average PSF that is better conditioned than each individual PSF. These observations will be illustrated with figures in Secs. 3.5.1 and 3.6.1. Considering these points, one can expect that deconvolving the average input sky based on the average PSF is easier and better reconstructs constant sources than deconvolving each individual sky based on the corresponding PSF. Therefore, to use these averaged representations, we design another new network coined Deflation Net and summarized in Algorithm 4. This network is based on the same subnetworks as 2D-1D Net but involves a different computation flow. Specifically, it decouples the reconstruction of constant and transient sources. In the first place, both input images $\{y_t\}_{t\in I}$ and PSFs $\{h_t\}_{t\in I}$ are averaged over the temporal dimension (lines 1 and 2). The reduced noise level of the average input image can be analytically computed (line 4). Then, the average image is deconvolved in the feature space based on the average PSF to give the average sky features (line 7). Next, this average sky is reconvolved by the corresponding PSF (line 14) and subtracted to the individual degraded sky in the feature space at each time step. Each resulting image only contains transient sources. This sky is then deconvolved based on the individual PSF to reconstruct transient sources (line 15). This individual deconvolved image and the average deconvolved sky are finally summed in the feature level via skip connections (line 16). This processing is done for each time step, and all outputs are then sent to the final 1D Net (lines 19 and 20).

Remarks on the incorporation of knowledge about the degradations

In our work, we do not rely on any iterative proximal algorithm with the prior replaced by one or more DNNs, nor on deep unrolling. Indeed, the former requires a significant number of iterations to deconvolve each frame, resulting in an overall slow reconstruction of an entire cube. The latter needs unrolled steps for each deconvolved frame, therefore, presents a memory issue during the backpropagation step when dealing with the entire cube. To mitigate this problem, the crop size of the input image has to be small, which can limit the corresponding size of the sky and the PSF. Moreover, the gradient step in an unfolding approach requires convolving at each step the input signal by the PSF and its

Algorithm 4: Deflation Net. C_r denotes the operation that center-crops a 2D structure with crop size r.

Input: $(Y, H, \sigma_\epsilon) = (\{y_t\}_{t\in I}, \{h_t\}_{t\in I}, \sigma_\epsilon)$
Output: $\hat{X}$

$\underline{y} \leftarrow \frac{1}{T}\sum_{t=0}^{T-1} y_t$
$\underline{h} \leftarrow \frac{1}{T}\sum_{t=0}^{T-1} C_r h_t$
$\vec{\underline{h}}^b \leftarrow P\underline{h}$
$\underline{\sigma} \leftarrow \frac{1}{T}(\sum_{t=0}^{T-1} ||M_t||_2^2\sigma_\epsilon^2)^{1/2} = \frac{1}{T}(\sum_{t=0}^{T-1} ||\mathcal{F}(h_t)||_2^2\sigma_\epsilon^2)^{1/2}$
$\vec{\underline{h}} \leftarrow \text{Concat}(\underline{h}^b, \underline{\sigma})$
$\underline{f} \leftarrow \text{Encoder}(\underline{y})$
$\underline{z} \leftarrow \text{SFT Net}(\underline{f}, \vec{\underline{h}})$
$S \leftarrow$ an empty list
for all $t \in [|0, T-1|]$ **do**
  $\vec{h}_t^b \leftarrow PC_r h_t$
  $\sigma_t = ||M_t||_2 \cdot \sigma_\epsilon = ||\mathcal{F}(h_t)||_2 \cdot \sigma_\epsilon$
  $\vec{h}_t \leftarrow \text{Concat}(\vec{h}_t^b, \sigma_t)$
  $f_t \leftarrow \text{Encoder}(y_t)$
  $\Delta_t \leftarrow f_t - h_t * \underline{z}$
  $d_t \leftarrow \text{SFT Net}(\Delta_t, \vec{h}_t)$
  $z_t \leftarrow d_t + \underline{z}$
  $S.\text{append}(z_t)$
end for
$Z \leftarrow \text{StackAlongTimeAxis}(S)$
$\hat{X} \leftarrow \text{1D Net}(Z)$
return $\hat{X}$

adjoint, which dilutes the signal too much because the PSF is strongly ill-conditioned in this radio interferometry problem. Therefore, unfolding approaches result in poor performance.

3.5 Experiment

3.5.1 Datasets

We generate disjoint training, validation, and test sets of GT and PSF cubes at various noise levels. The corresponding dirty cubes are generated based on Eq. (3.3).

PSF cubes

We simulate the interferometric response of MeerKAT in the L-band, using its current 64 antennas distribution. The observing frequency is fixed to 1420 MHz. The location of

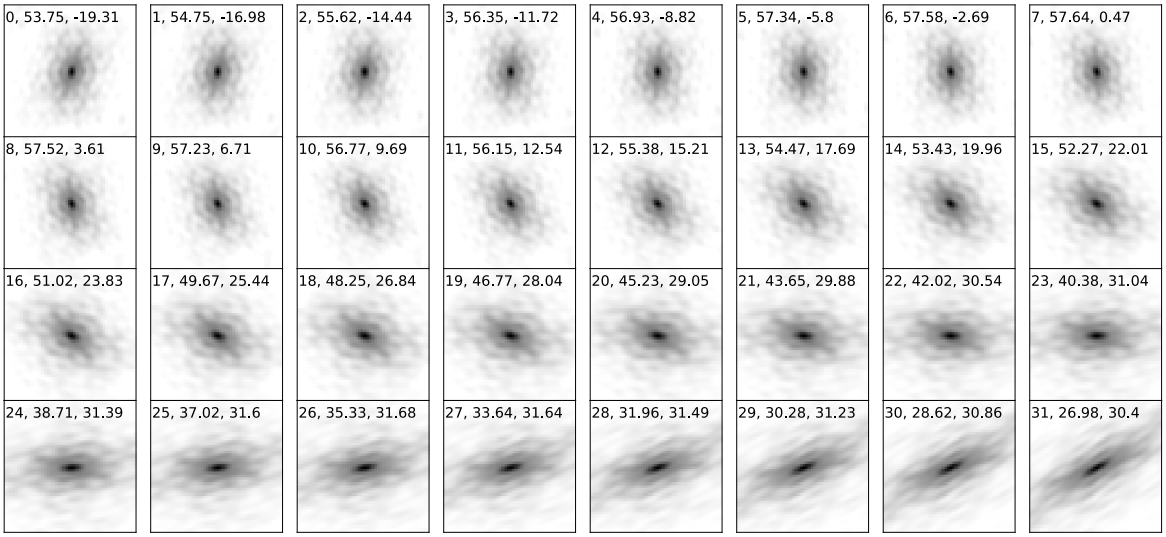


Figure 3.4: The PSF test cube that has been picked up for evaluation. Text on each image reports: time step, elevation in degrees, azimuth in degrees. Each PSF is normalized to 1 (black). The grey color scale was reversed for clarity.

MeerKAT on Earth is: $Long = 21.33°$, $Lat = -30.83°$, $h = 1195m$. The minimum pointing elevation is set to 20 degrees. Based on these parameters, we randomly pick PSF cubes by sampling the local visible sky for a given observing time and observing rate. Each cube accounts for the typical 8-hour tracking observation of a randomly selected source in the J2000 reference frame that is visible in the local sky during the 8 hours. This period is divided into successive intervals of 15-minute scans. They produce a 15-minute integrated PSF associated with the (u,v) coverage computed towards the source's current location. Each PSF is projected into a spatial grid of 256×256 pixels. Hence, the cube has the dimensions $32 \times 256 \times 256$. Adopting the notations from Sec. 3.3.2, we have $T = 32$ and $t_{i+1} - t_i = 15$ minutes. We generated 435 training, 50 validation, and 50 test PSF cubes samples. For all of them, the source elevations at the beginning of the observation are above 20 degrees as seen from MeerKAT. Fig. 3.4 depicts the 32 (15 min) frames of a PSF cube. As the source appears to be rotating above the interferometer during the 8-hour observation, the projected baselines vary, leading to a continuous rotation and warping of the PSF.

Fig. 3.5 depicts the effective distribution of elevation in the training PSFs set. This distribution is skewed toward 30° elevation in the case of the telescope at hand, MeerKAT. This is expected because the PSF simulation accounts for the visibility of sources in the southern sky, as seen from MeerKAT. For any given observation duration (*e.g.*, 8h max) and integration (*e.g.*, 15 min/image), we computed the telescope source tracking and its aperture projection, both required to derive the effective (u,v) coverage and therefore the PSF. An astrophysical source is associated to a unique pair of coordinates (declination δ and right ascension α) on the celestial sphere (see Appendix A.5 for definitions). Depending on the source declination (δ, in the equatorial frame), hour angle (associated with Right Ascension α and time of observation), and observation duration, not all (α,δ) directions are accessible. Therefore, a uniform random distribution of directions, drawn from the "accessible" direction window in the sky, will appear skewed around the local elevation of the South Celestial Pole, located around 30° for MeerKAT. Circumpolar sources close to the

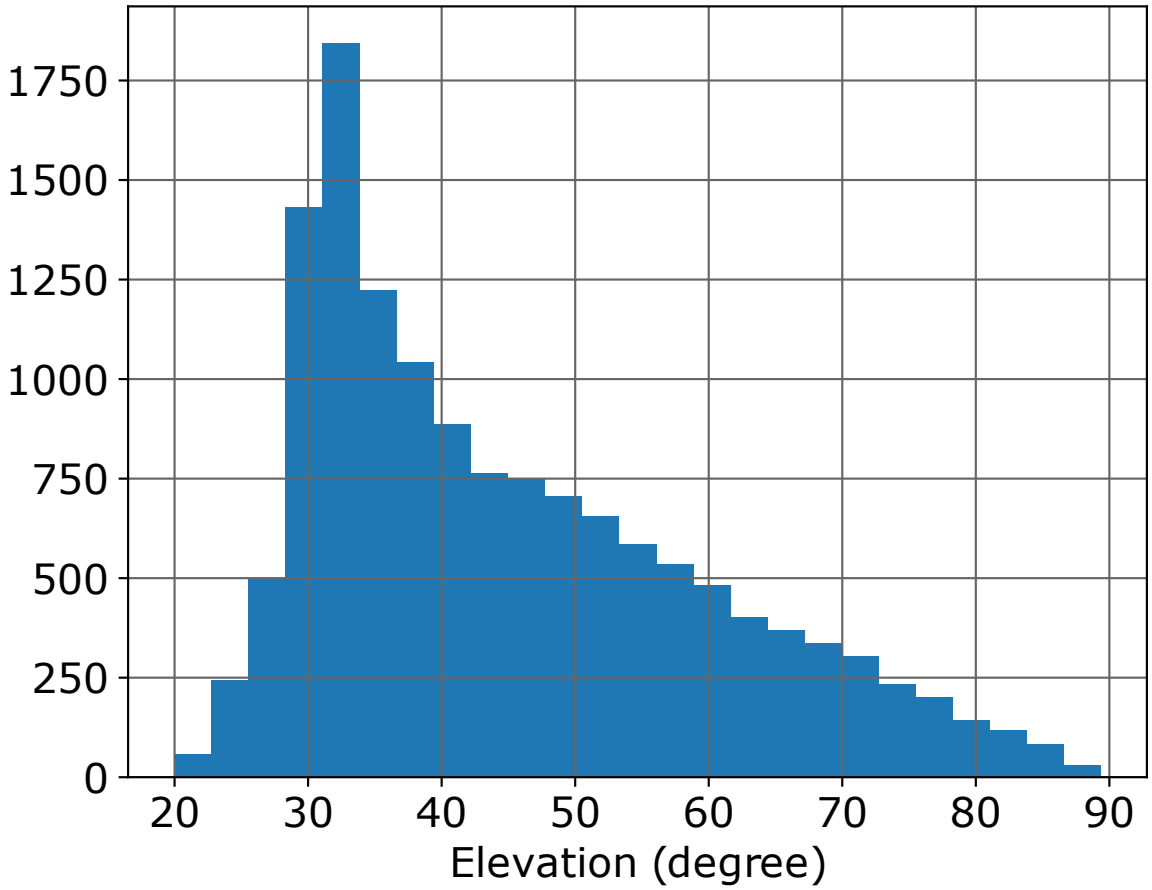


Figure 3.5: Histogram of distribution of elevations in the training PSF set. The distribution is skewed toward the South Celestial Pole where circumpolar sources can be observed from MeerKAT.

South Celestial Pole are always accessible.

The PCA projection matrix P that encodes knowledge about the PSF is learned from all PSFs of the train PSF set, *i.e.*, from the $435 \times 32 = 13920$ PSFs. The value of PCA components $b = 50$ explains 90 % of the total variance. Fig. 3.6 shows the 10 main PCA eigenvectors which composed the PSF of the whole set.

GT cubes

We suppose the sources to be unresolved points sources in the sky image. The size of the pixel on the sky is fixed to $1.5''$. We assume that within a sky image of size $30' \times 30'$ (*i.e.*, 1200×1200 pixels), at most 30 constant and 2 transient point sources can be placed. This distribution of sources can be considered compatible with shallow imaging like MeerKAT. Following this distribution, we generate 39000 training sky cubes with dimensions $32 \times 256 \times 256$. These data are divided into three equal parts, containing zero, one, or two transient sources per field. Each validation and test set contains 66 cubes following the same distribution of sources, with half of them containing a transient source and the other

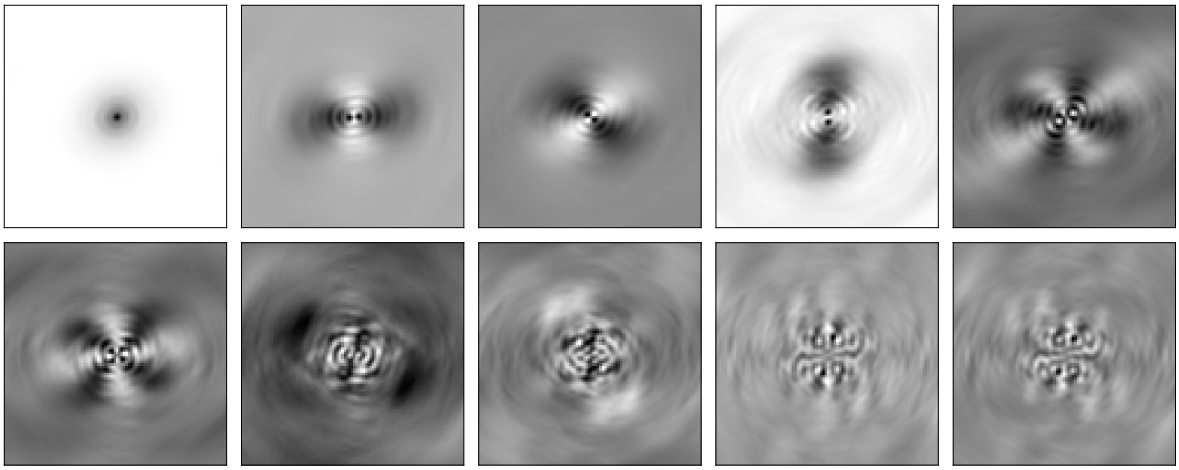


Figure 3.6: PCA eigenvectors for the first 10 largest eigenvalues computed over the whole PSF set.

half containing two. The source peak flux density, *i.e.*, the value of the pixel of a constant source in the GT, is randomly sampled between 1 and 100. Regarding a transient source, its amplitude is a discrete function $A(t_i)$ sampled from the continuous functions $A(t)$. For each source, this profile is randomly chosen between the following models: *gate*, *Gaussian* and *skew-normal*. Their parameters are randomly chosen, and the transient source's maximum amplitude is randomly chosen between $[50, 100]$. Examples of temporal profiles are shown in Sec. 3.6, Fig. 3.7.

3.5.2 Training procedure

From each GT cube in the training, validation, and test sets, the corresponding *dirty cube* is generated based on Eq. (3.3). Mini-batch gradient descent with a batch size of 4 is used for training. For each example in the batch, a train PSF cube is randomly picked, and the noise level σ_ϵ is randomly sampled from $[0, 6]$. Data augmentation with random flipping/transposition was performed. The learning rate is set to 10^{-4}, and we train our models for 100 epochs each. The loss function is the pixel-wise mean-squared error (MSE) between GT and estimated cubes. Authors of [129] claimed that if GT images are skies with point sources, then the MSE loss function would be suboptimal and instead proposed to smooth the point sources. However, our study obtained satisfactory empirical results with this pixel-wise MSE between GT and estimated cubes.

3.5.3 Evaluation metrics

Given a constant or transient source at a certain location in a GT cube X, we can first define a subcube obtained by locally cropping the cube around the source with a path size $p \times p$ ($p = 3$ in our study, defining the region $\mathcal{D}_s$). We can also extract a subcube at the same location for the estimated cube. We can then compute the root MSE (RMSE) between the two subcubes. This error quantifies the fidelity of the restored temporal profile. The GT

subcube norm can normalize this value. We note $\mathrm{NRMSE_s}$ this error, for the source s.

To measure the input signal quality, for each source s in the *dirty cube* $\{y_t\}_{t\in I}$ with location (i_s, j_s) we evaluate its signal-to-noise ratio $\mathrm{SNR_s}$. We adopt the following definition:

$$\mathrm{SNR_s} = \sum_{i,j\in\mathcal{D}_s} \frac{y_\tau[i,j]}{\sigma_b} \tag{3.6}$$

where τ refers to the temporal localization of the transient. It corresponds to the time step when the amplitude of the transient source is maximum in the GT cube. For a constant source, we set $\tau = 15$. $\mathcal{D}_s$ is the local patch of size 3×3 centered on the true source location (i_s, j_s). σ_b denotes the background noise estimated on y_τ after excluding $\mathcal{D}_s$. In radio images, Peak SNR is the ratio of peak flux density (usually a single pixel) of the source and the local noise root mean square. Here, to absorb pixel gridding bias, the source flux is accounted over the region $\mathcal{D}_s$. This will ensure that all the recovered flux density of the source is properly accounted for, in presence of noise. This error could lead to detrimental and unfair representations of light curve reconstruction with different methods.

Moreover, to measure the performance of background denoising, *i.e.*, restoration of the empty region of the sky, we exclude all $\mathcal{D}_s$ subcubes for a GT cube. We operate the same procedure on the corresponding estimated cube and compute the RMSE between the two. We note this metric $\mathrm{RMSE_{noise}}$.

We compare the proposed algorithms for the cube restoration against frame-by-frame CLEAN deconvolution of the *dirty cubes*. To avoid biasing the final result during the CLEAN final restoring beam step, we considered only the detected CLEAN components associated with the detected sources. PSF cube and noise level are provided to CLEAN in a non-blind manner. They intervene in defining a threshold from which the algorithm iteration stops. Furthermore, to prevent bias on the use of CLEAN due to a high noise environment of the *dirty cube*, we stopped CLEAN based on a maximum number of iterations and, as for other reconstruction methods, also considered only counting the flux density around the source location in $\mathcal{D}_s$.

3.6 Results

3.6.1 Fixed test PSF cube and varying noise

We pick a PSF test cube and evaluate the methods on the test sky cubes with the following noise levels: $\sigma_\epsilon \in \{0, 1, 2, 3, 4, 5, 6\}$. The injected noise levels were selected to range from low noise level cases, where the constant and transient sources are readily detectable in the *dirty cubes*, to high noise level cases where no sources can be seen (such a high noise level is displayed on the first line of Fig. 3.8). Fig. 3.4 shows the PSF test cube that has been picked. We see that the PSF rotates with time.

Fig. 3.7 compares the reconstruction of temporal profiles related to several transient sources in the test cubes. The figure compares methods at different noise levels σ_ϵ. We ob-

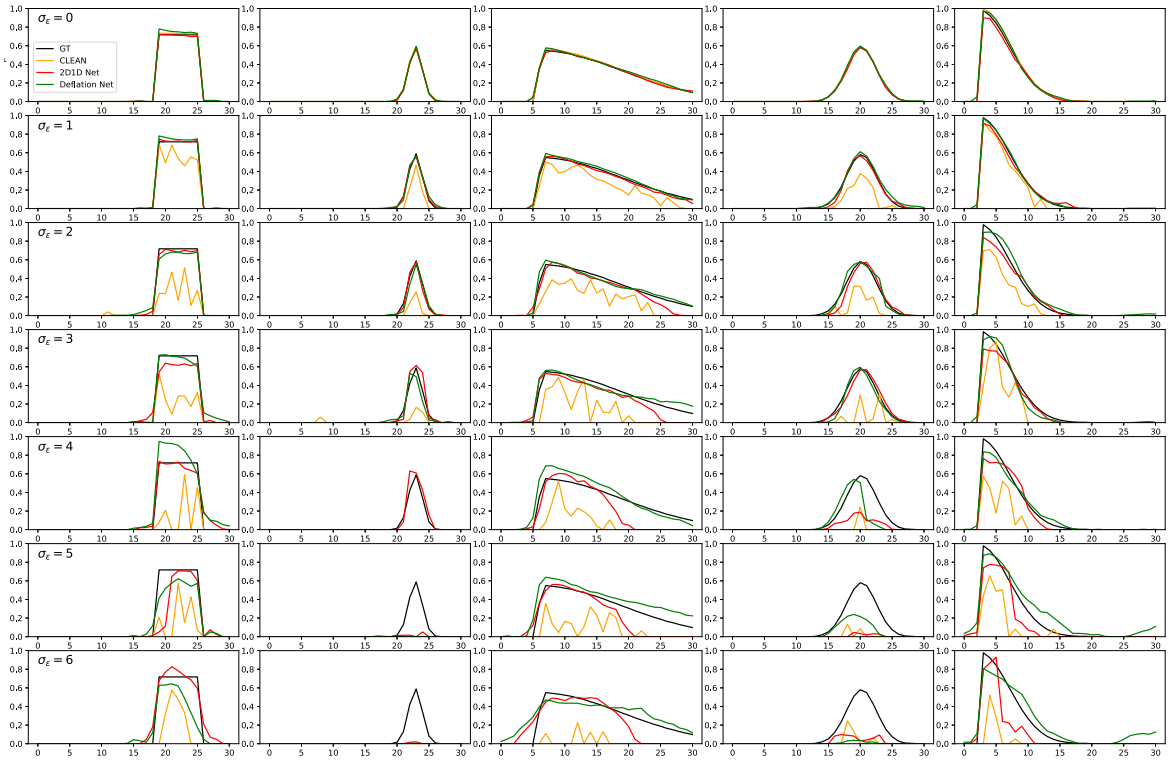


Figure 3.7: Reconstructions of temporal profiles related to some transient sources in the test cubes. The horizontal and vertical axes indicate the time step and amplitude for each subfigure. The figure compares methods at different noise levels σ_ϵ. The names of the sources are, from left to right: *source 1*, *source 2*, *source 3*, *source 4*, and *source 5*.

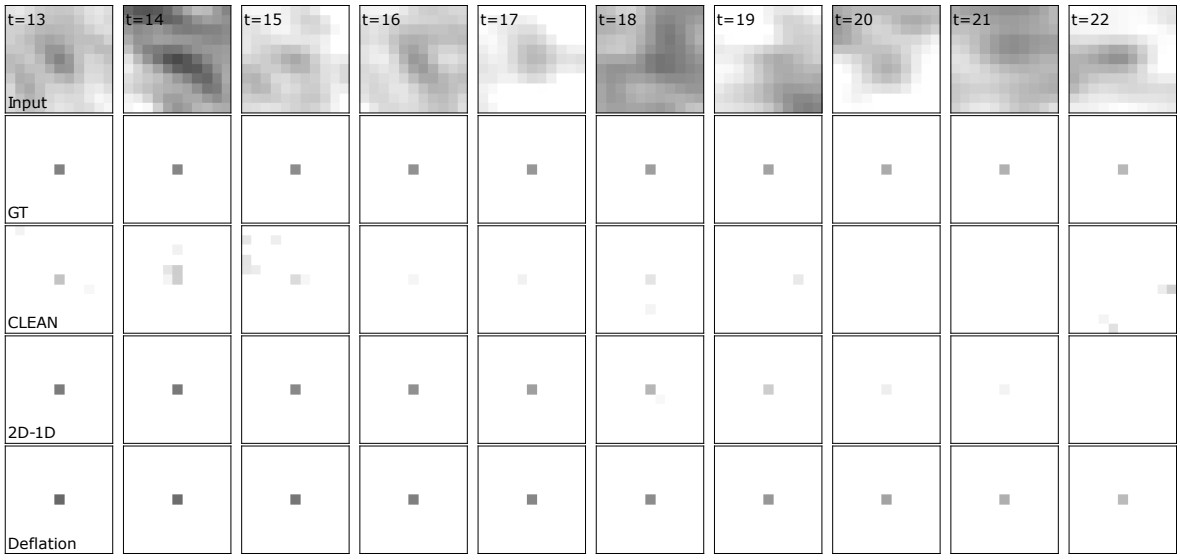


Figure 3.8: Visualization of *source 3* in Fig. 3.7 reconstructed from different methods, at noise level $\sigma_\epsilon = 4$.

serve that DL-based methods produce better reconstructions than CLEAN. With increasing noise level σ_ϵ, the performance of CLEAN rapidly degrades, whereas that of DL-based meth-

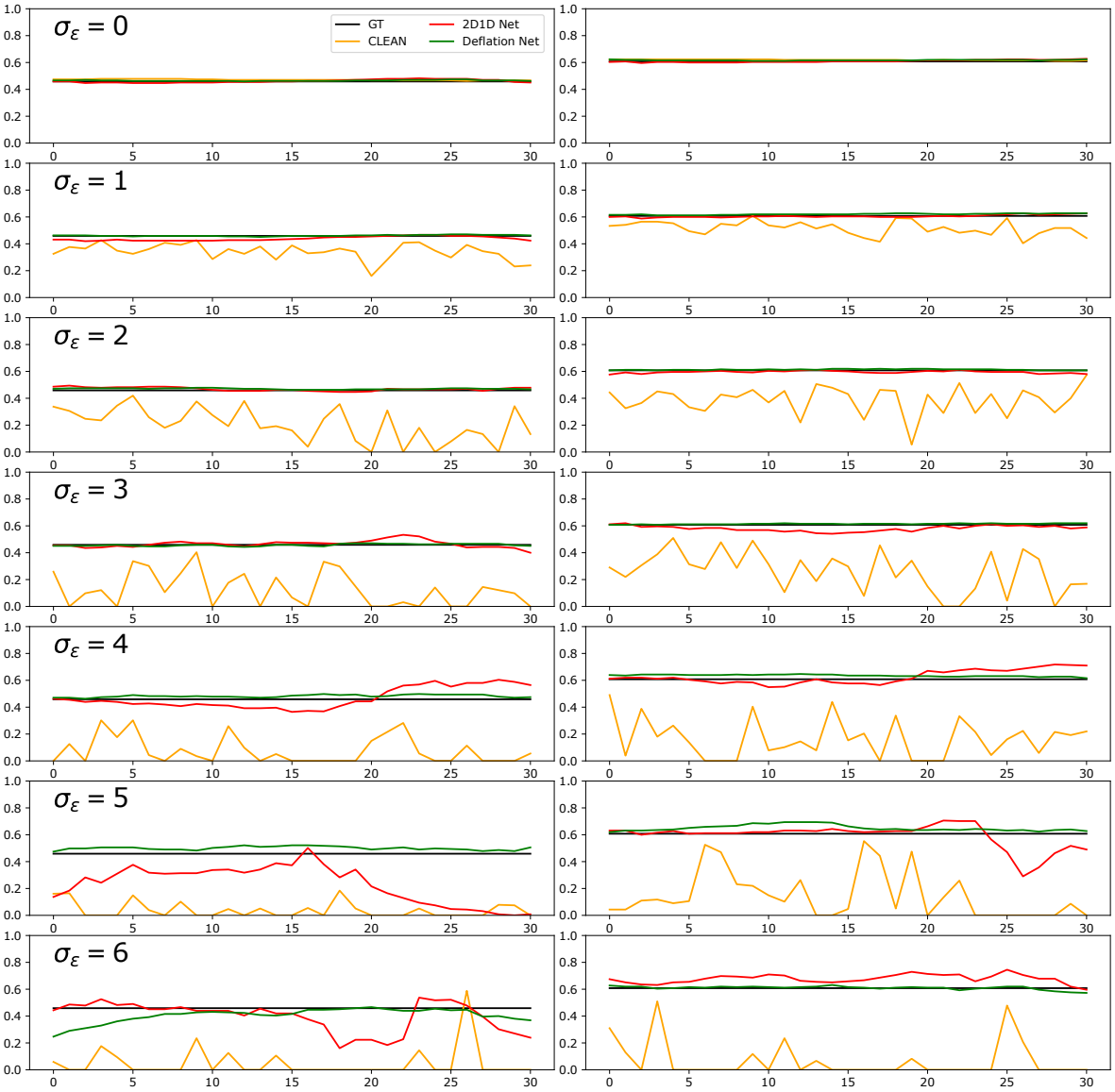


Figure 3.9: Reconstructions of temporal profiles of constant sources. The horizontal and vertical axes indicate the time step and amplitude for each subfigure.

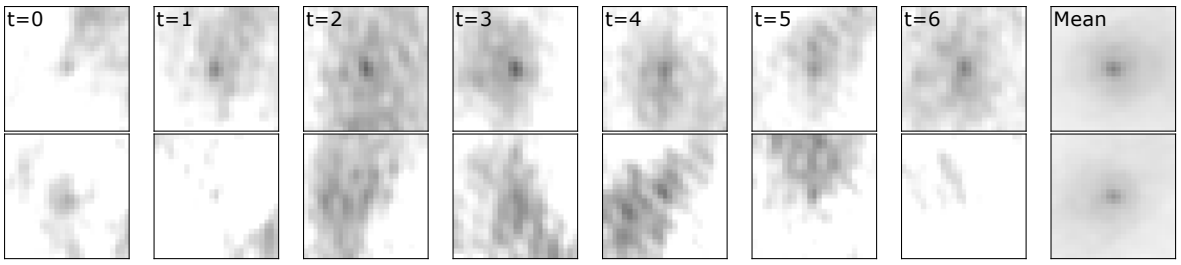


Figure 3.10: A constant source in the input degraded sky between time steps 0 and 6, and the average of the input skies over all the time steps. First row: noise level $\sigma_\epsilon = 2$. Second row: $\sigma_\epsilon = 3$. The noise level is reduced in the averaged sky.

ods remains good. The plots of Fig. 3.7 related to *source 1* illustrate this. Moreover, even in the presence of noise, DL-based methods provide a better restoration and preservation of the start and end dates of the transient signal. This is granted by the temporal modeling capability of these methods. Conversely, in the case of CLEAN, the restored start and end dates of the transient is distorted because of the noise in single frames. The plots related to *source 5* at noise level $\sigma_\epsilon = 3$ and $\sigma_\epsilon = 4$ illustrate these statements regarding the end of the transient signal. Among DL-based methods, Deflation Net seems better for restoring transient signals at a high noise regime. The plots of *source 3* depict this. This suggests that the architectural choice of Deflation Net decoupling constant and transient source reconstructions is appropriate with respect to the inverse problem. Even at the highest noise level $\sigma_\epsilon = 6$, Deflation Net can restore the end of the transient signal. Fig. 3.8 visualizes *source 3* reconstructed from different methods, at noise level $\sigma_\epsilon = 4$. We see that DL-based methods more efficiently reconstruct the transient source than CLEAN. Moreover, Deflation Net performs the best in reconstructing the end of the transient source. These results are particularly important whenever one telescope wants to react quickly upon detecting a potential transient. A transient could be detected on a nearly real-time imaging pipeline that integrates our trained network as soon as its light curve rises. Alerts could therefore be created and distributed earlier and with more confidence concerning false detections.

As a matter of comparison and robustness, Fig. 3.9 compares reconstructions of temporal profiles for constant sources. As expected, DL-based methods systematically present reconstructions with higher fidelity. Moreover, they produce light curves with more stable amplitude variations than the CLEAN-based method. Indeed, we must ensure that the trained network correctly reconstructs the light curves of constant sources without making them look like flickering transient sources. Moreover, Deflation Net presents less varying reconstructed constant sources than 2D-1D Net. Indeed, Deflation Net is understood to provide a better reconstruction of constant sources for the following reasons. Firstly, the noise level is reduced in the average input sky, and the average PSF is better conditioned than the individual PSF frame. Deconvolution of an average sky by the average PSF is thus easier for constant sources. Figs. 3.10 and 3.11 illustrate this by respectively displaying the average input sky for various noise levels and the average PSF. Secondly, the skip connections of Deflation Net (line 16 of Algorithm 4) allow reconstructions at different time steps to be distributed near the average deconvolved sky. Fig. 3.12 compares the average variance of reconstructed constant sources over the test cubes at different values of σ_ϵ. The figure confirms that DL-based methods reconstruct constant sources with better fidelity than CLEAN and that Deflation Net produces the lowest variances concerning constant sources. On average, constant sources reconstructed by 2D-1D Net present a variance $\sim$3.0 times smaller than CLEAN. Deflation Net's variance is $\sim$8.6 times smaller than CLEAN.

After aggregating results of all inferences with all of the noise levels $\sigma_\epsilon \in \{0, 1, 2, 3, 4, 5, 6\}$, Fig. 3.13 shows $\mathrm{NRMSE_s}$ averaged over sources belonging to the same SNR interval delimited by deciles in $\mathrm{SNR_s}$. We observe that DL-based methods generally perform better than CLEAN, except when the $\mathrm{SNR_s}$ is very high ($\mathrm{SNR_s}197$). This case indeed corresponds to when $\sigma_\epsilon = 0$ for which CLEAN is optimal. In other cases, DL-based methods, on average, perform better than CLEAN. By comparing positions of error bars indicating standard deviations, we observe that in many cases ($\mathrm{SNR_s}$ between 40.5 and 128.6), DL-based methods present inhomogeneous performances over different sources, but their worst performances

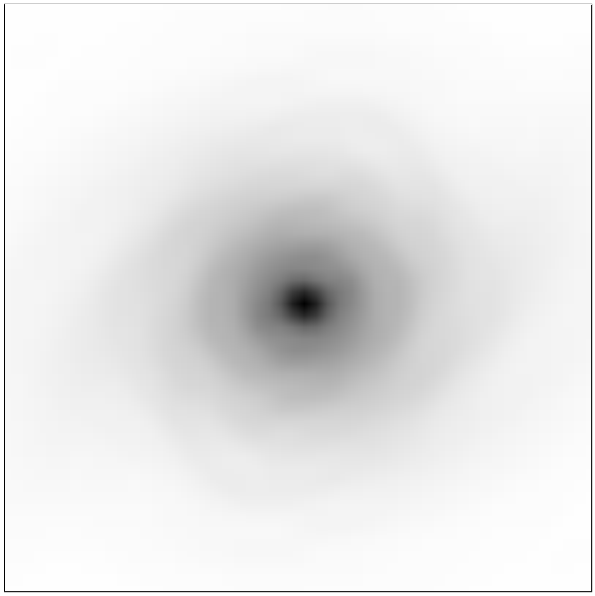


Figure 3.11: Average PSF over the PSF cube of Fig. 3.4.

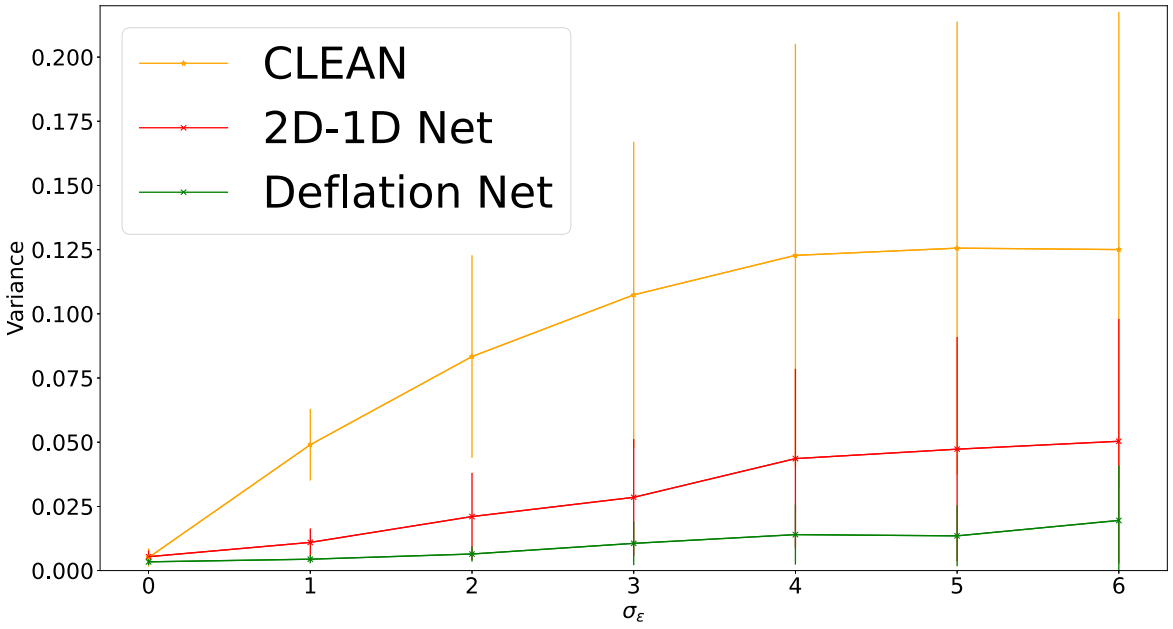


Figure 3.12: The average variance of reconstructed light curves of constant sources over the test cubes at different values of the noise level σ_ϵ. The variances are computed on cubes normalized between 0 and 1. Vertical bars show standard deviation.

are statistically still better than the best performances of CLEAN. At a high noise regime ($\mathrm{SNR_s}$ below 40.5), DL-based methods perform better than the CLEAN-based method on average. Within the DL-based methods, Deflation Net performs better than 2D-1D Net on average in most noisy cases (SNR between 0 and 86). On average, CLEAN presents $\mathrm{NRMSE_s}$ that are $\sim$2.4 times higher than 2D-1D Net and $\sim$2.7 times higher than Deflation Net.

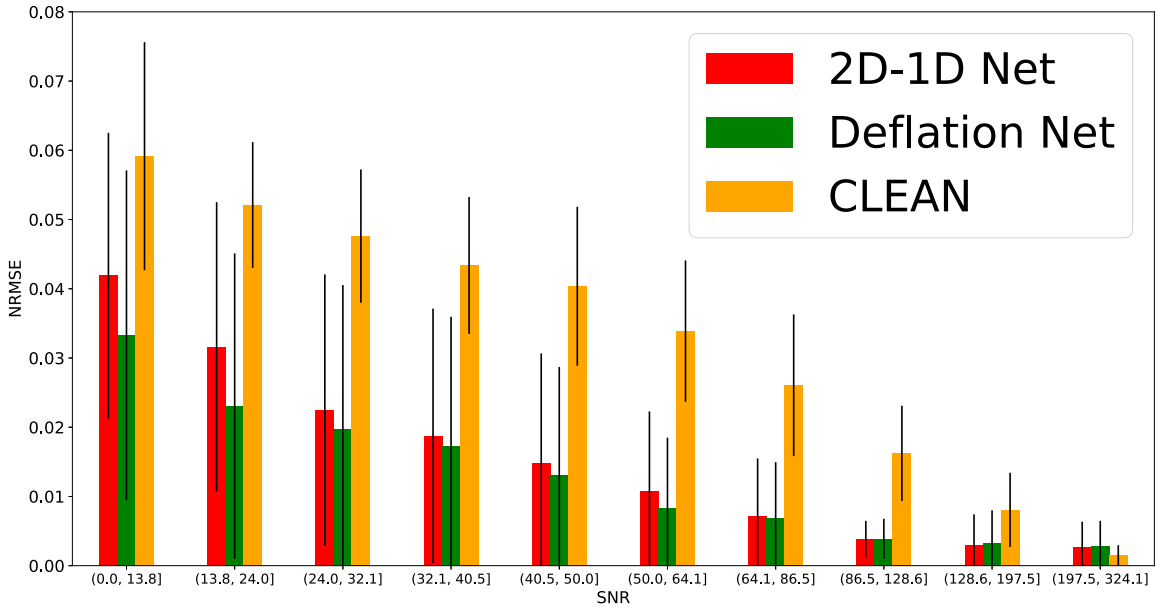


Figure 3.13: Mean $\mathrm{NRMSE_s}$ for sources in the test set for each decile of $\mathrm{SNR_s}$. Vertical bars represent standard deviations.

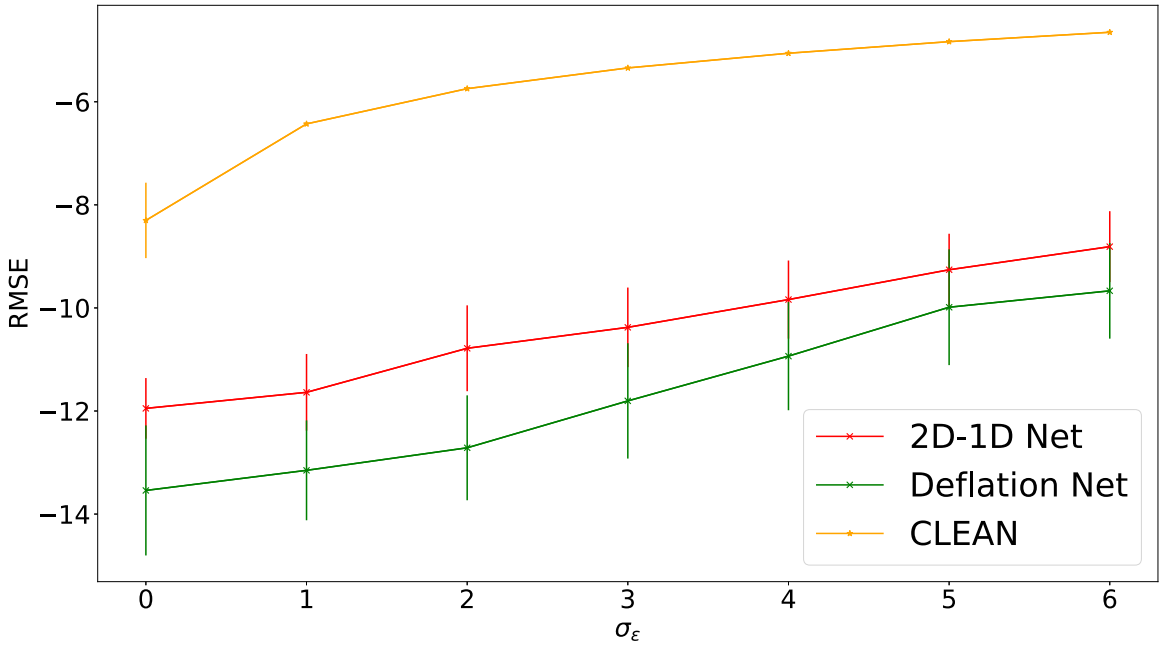


Figure 3.14: $\log(\mathrm{RMSE_{noise}})$ averaged over the test cubes at different values of σ_ϵ. Vertical bars show standard deviation. Because of the log scale, the standard deviation is higher when $\mathrm{RMSE_{noise}}$ is small. This is the case for the DL-based methods.

Fig. 3.8 illustrates the high performance of DL-based methods in background denoising. We observe that they effectively restore the empty sky around the sources. This is less the case for the frame-by-frame CLEAN method: the latter captures noise and generates residual noisy pixel distributions around the true source. Fig. 3.14 compares $\mathrm{RMSE_{noise}}$,

averaged over the test cubes at different values of σ_ϵ. We observe that for all values of σ_ϵ, DL-based methods better suppress background noise than CLEAN. Within the DL-based methods, Deflation Net better restores the background sky. This is because the noise level is reduced and the PSF is better conditioned in averaged input sky. On average, CLEAN presents $\mathrm{RMSE}_{\mathrm{noise}}$ values that are $\sim$1.8 times bigger than the 2D-1D Net ones and $\sim$2 times bigger than the Deflation Net ones.

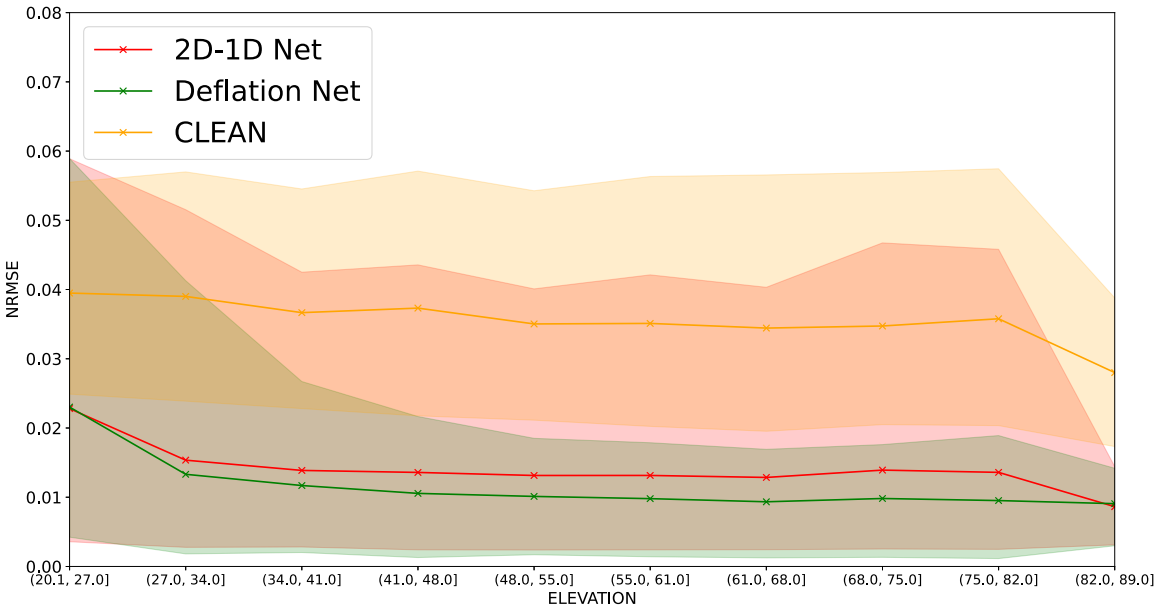


Figure 3.15: Mean $\mathrm{NRMSE}_{\mathrm{s}}$ for sources in the test set for 10 equally spaced bins of PSF pointing elevations. The envelope for each method delimits the first and the last 10 percentiles.

3.6.2 Varying test PSF cube and fixed noise

In this section, we set $\sigma_\epsilon = 3$ and aggregate results over all of the PSF test cubes.

Fig. 3.15 shows average $\mathrm{NRMSE}_{\mathrm{s}}$ for sources in the test set for 10 equally spaced bins of PSF elevations. The apparent envelope for each method delimits the first and the last 10 percentiles. We see that for all bins, our DL-based methods, on average, outperform CLEAN in large margins. This is especially true for Deflation Net, which presents the last 10 percentile near the first 10 percentile of CLEAN in most cases (elevations>41 degrees). Among DL-based methods, here again, in most cases, Deflation Net performs better on average than 2D-1D Net (elevations between 27 and 82 degrees). This confirms the Deflation Net architecture efficiently tackles the inverse problem. On average, CLEAN presents NRMSE that are $\sim$2.6 times bigger than 2D-1D Net and $\sim$3.2 times bigger than Deflation Net. Furthermore, in most cases (elevations $>$ 34 degrees), the envelope of Deflation Net is smaller than the ones of 2D-1D Net and CLEAN. Therefore, Deflation Net brings less error dispersion in source light curve reconstruction than other methods. This can be explained by its skip connections that set reconstructions at different time steps near the average de-

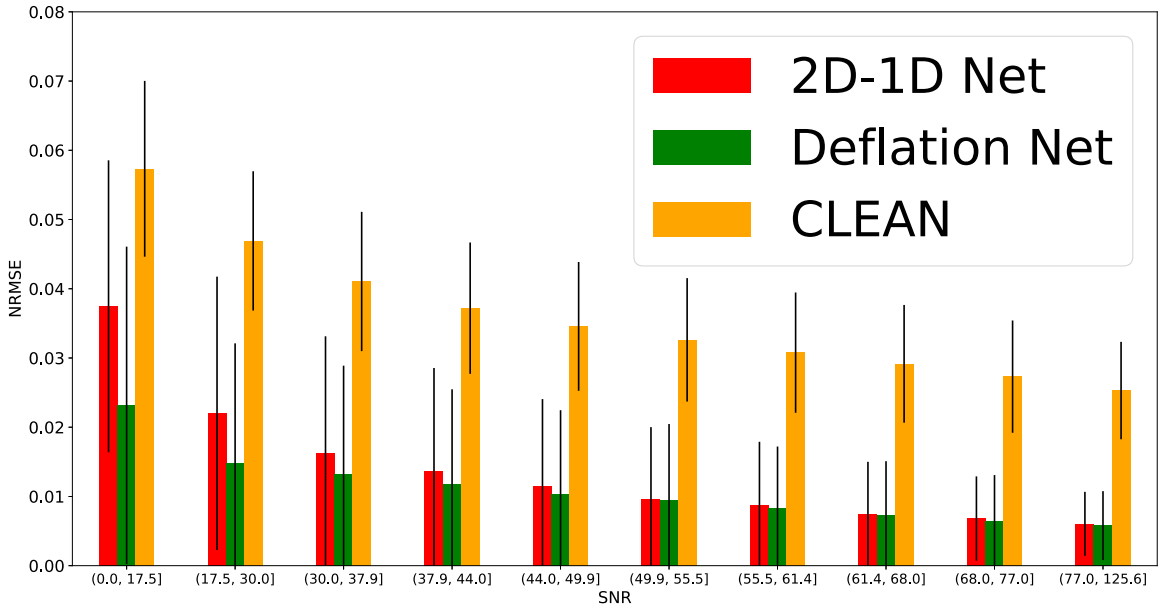


Figure 3.16: Mean $\mathrm{NRMSE_s}$ for sources in the test set for each decile of $\mathrm{SNR_s}$. Vertical bars represent standard deviations.

convolved sky. Our methods perform worse at elevations between 20 and 27 degrees and better at elevations between 82 and 89 degrees. Between these bounds, their performance seems to increase with increased elevation. This is coherent: the closer to 90 degrees the elevation, the better conditioned the PSF is, therefore the easier it is to deconvolve. Conversely, lower elevation means that the PSF is worse conditioned and undergoes a strong projection effect, making the deconvolution task harder.

Second, after aggregating the results of all inferences with all the test PSF cubes over the test sky cubes, Fig. 3.16 shows $\mathrm{NRMSE_s}$ averaged over sources belonging to the same SNR interval delimited by deciles in $\mathrm{SNR_s}$. We observe that DL-based methods perform better on average than CLEAN for all intervals of $\mathrm{SNR_s}$. We can state the following by observing vertical bars indicating standard deviations around mean values: in most cases, DL-based methods present inhomogeneous performance, but their worst performances are statistically still better than the best performances of CLEAN. Within the DL-based methods, Deflation Net performs slightly better than 2D-1D Net when $\mathrm{SNR_s}$ is higher than 44 and outperforms 2D-1D Net when $\mathrm{SNR_s}$ is below 44. This confirms again that the Deflation Net architecture is appropriate regarding the inverse problem. On average, 2D-1D Net presents $\mathrm{NRMSE_s}$ that is 3.10 times smaller than CLEAN. Deflation Net shows $\mathrm{NRMSE_s}$ values that are 3.50 times smaller than that of CLEAN.

3.6.3 Importance of temporal modeling

In this part, we evaluate the benefit brought by the 1D Net in 2D-1D Net. This measures to which extent capturing the temporal structure of a signal increases the signal reconstruction performance. To do so, we compare the performances of CLEAN, 2D-1D Net,

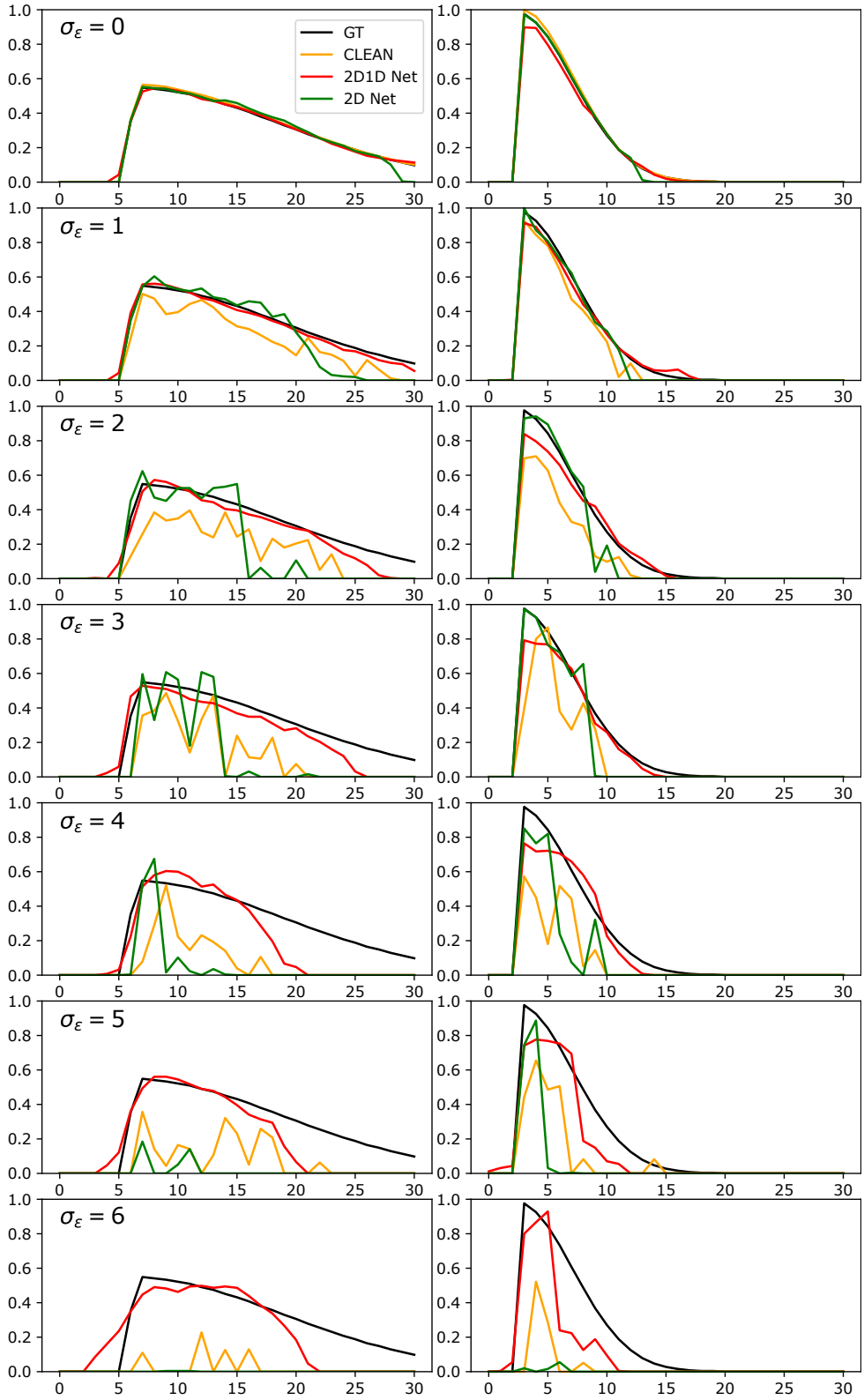


Figure 3.17: Reconstructions of temporal profiles of *source 3* and *source 5* from the test cubes (that were also reported in Fig. 3.7). The figure compares methods at different noise levels σ_ϵ.

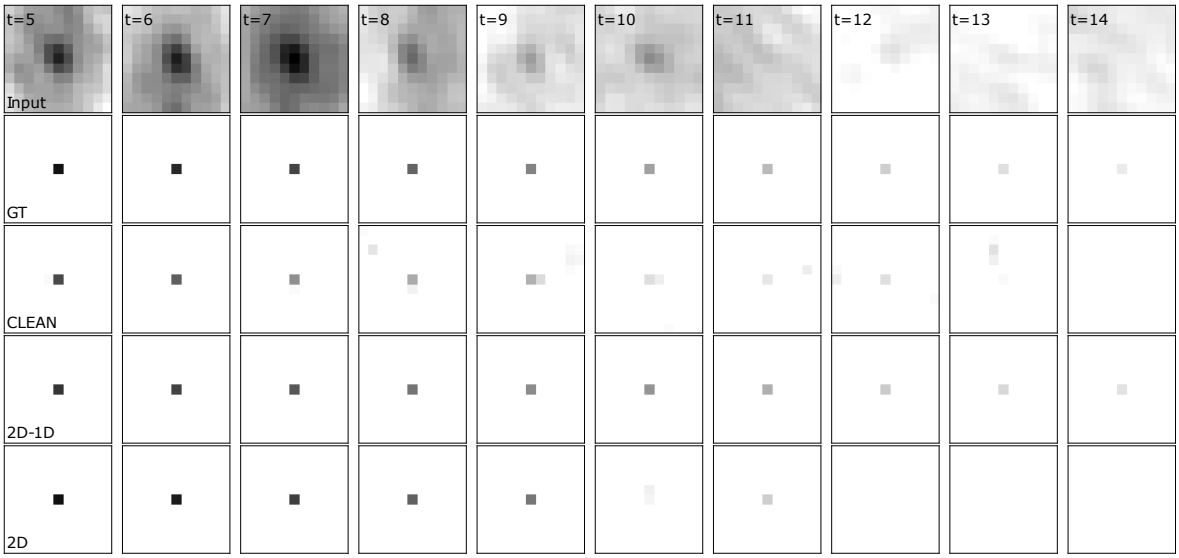


Figure 3.18: Visualization of the source 5 in Fig. 3.17 reconstructed from different methods, at noise level $\sigma_\epsilon = 2$.

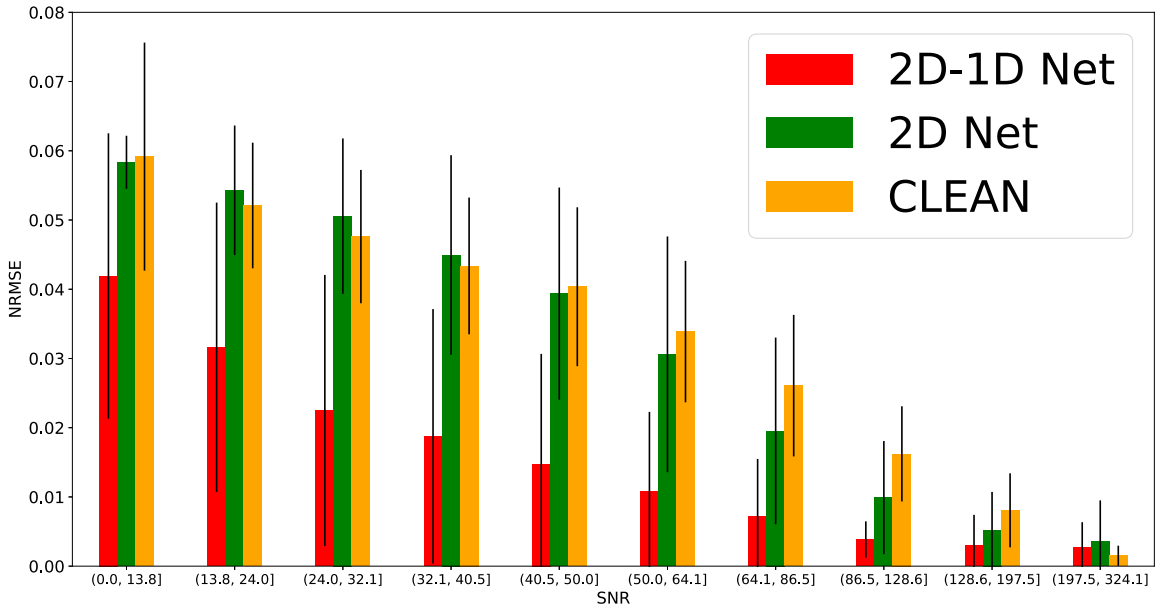


Figure 3.19: Mean $\mathrm{NRMSE_s}$ for sources in the test sky cubes for each decile quantile of $\mathrm{SNR_s}$. Vertical bars represent standard deviations.

and 2D Net. The latter is a variant of 2D-1D Net from which we exclude the 1D Net part, and we make the last 2D convolutional layer of 2D Net output a 1-channel image at each time step. This mode reduces our method to a classic imager that forms a single image on time-integrated data. Fig. 3.17 compares the reconstructions of temporal profiles of some transient sources in the test cubes with the fixed PSF used in Sec. 3.6.1 and illustrated in Fig. 3.4. As expected, we observe that with increasing noise levels, 2D Net rapidly loses the capability to reconstruct the ends of the light curve, which lodge at the noise level. This drawback is shared with CLEAN. On the contrary, 2D-1D Net succeeds in restoring them

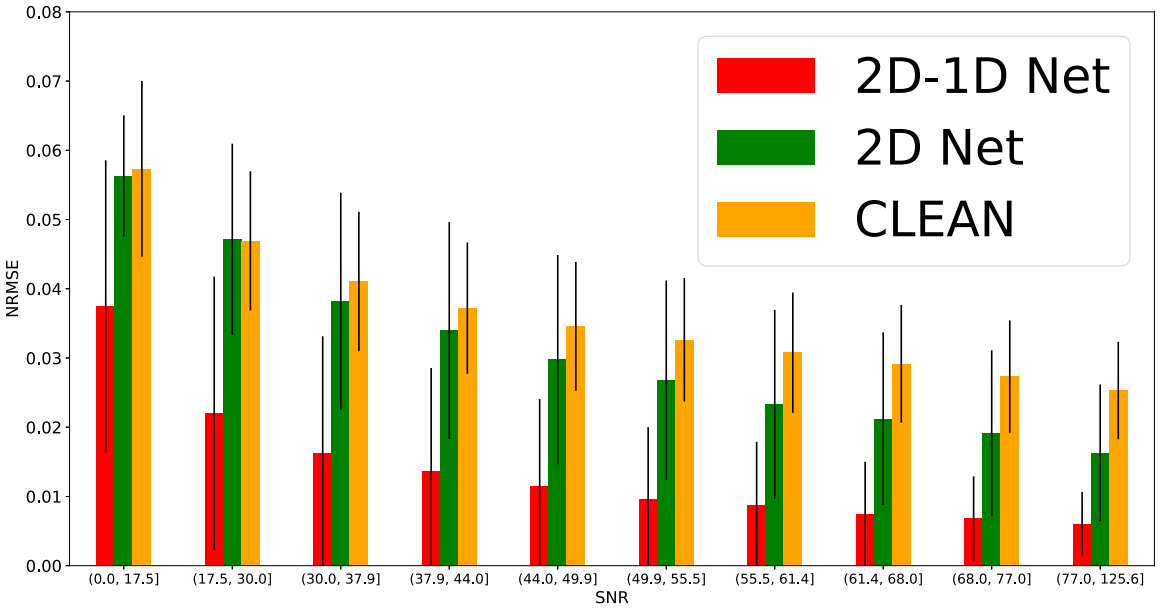


Figure 3.20: Mean $\mathrm{NRMSE_s}$ for sources in the test sky cubes for each decile quantile of $\mathrm{SNR_s}$. Vertical bars represent standard deviations.

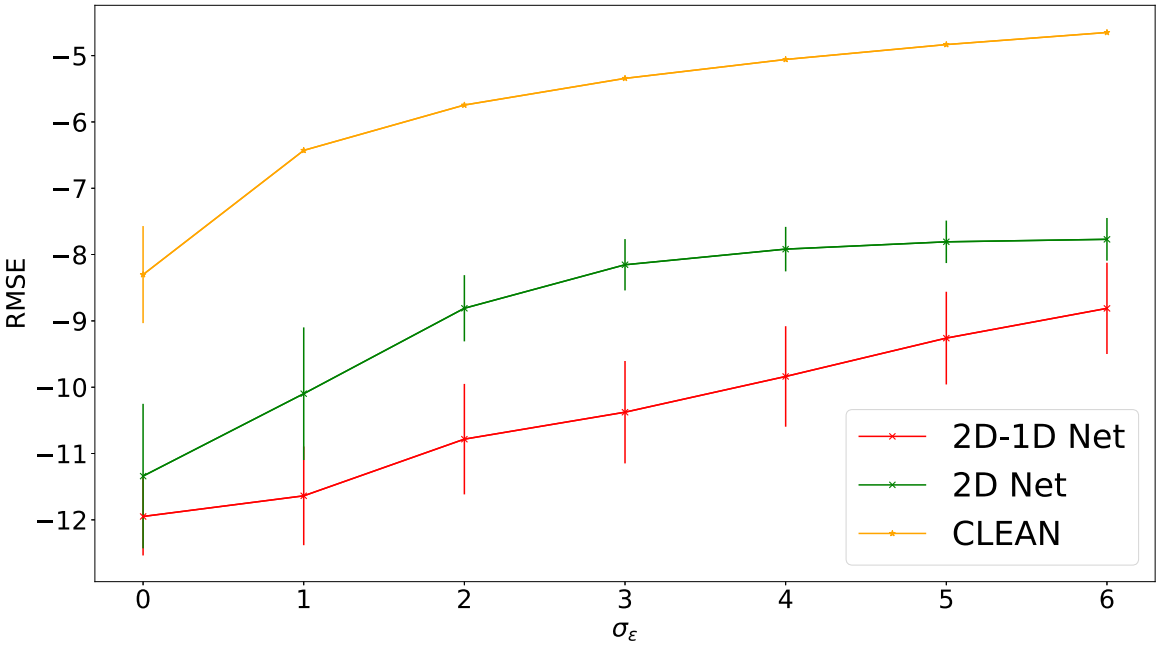


Figure 3.21: $\log(\mathrm{RMSE_{noise}})$ averaged over the test cubes at different values of σ_ϵ. Vertical bars show standard deviation. Because of the log scale, the standard deviation is higher when $\mathrm{RMSE_{noise}}$ is small. This is the case for the DL-based methods.

properly. Fig. 3.18 visualizes the *source 5* reconstructed from different methods, at noise level $\sigma_\epsilon = 3$. We see that 2D-1D Net reconstructs the transient profile more efficiently down to the disappearance. 2D Net fails to reconstruct it, similarly to CLEAN. These figures illustrate that, despite the noise, 1D Net allows correctly reconstructing the beginning

and the end of a transient event by enabling temporal modeling.

Fig. 3.19 aggregates inference results on the test cubes with the fixed PSF cube illustrated in Fig. 3.4 and the noise levels $\sigma_\epsilon \in \{0, 1, 2, 3, 4, 5, 6\}$. Fig. 3.20 aggregates inference results on the test cubes with the fixed noise level $\sigma_\epsilon = 3$ and the 45 PSF test cubes used in Sec. 3.6.2. Both figures compare $\mathrm{NRMSE_s}$ averaged over sources of the same interval delimited by deciles in $\mathrm{SNR_s}$. On average, we observe that 2D Net performs slightly better than CLEAN in terms of $\mathrm{NRMSE_s}$. 2D-1D Net significantly outperforms both methods and presents on average ~ 2.3 times smaller $\mathrm{NRMSE_s}$ than 2D-1D Net and CLEAN. This exemplifies the benefit brought by the temporal modeling capability of 1D Net.

Even if 2D Net performs similarly to CLEAN based on $\mathrm{NRMSE_s}$, the former performs better regarding background denoising. This is shown in Fig. 3.21, which compares $\mathrm{RMSE_{noise}}$, averaged over the test cubes at different values of σ_ϵ and the PSF illustrated in Fig. 3.4. For all values of σ_ϵ, 2D Net better suppresses background noise than CLEAN by large margins. This shows that even if 2D Net does not extract temporal features, its convolutional layers enable high-performing denoising. 2D-1D Net performs even better than 2D Net regarding this background denoising task. This shows that the temporal modeling capability of 1D Net also contributes to increasing the background denoising performance. On average, 2D Net presents $\mathrm{RMSE_{noise}}$ that are ~ 1.5 times smaller than that of CLEAN. 2D-1D Net presents $\mathrm{RMSE_{noise}}$ that are ~ 1.2 times smaller than 2D Net. Fig. 3.18 illustrates these observations. Especially at the end of the transient event, frame-by-frame CLEAN captures noise and generates residual noisy pixels around the true source. 2D Net generates less noisy pixels (it only generates a noisy pixel at time step $t = 10$). 2D-1D Net does not generate any obvious noisy pixel.

3.7 Discussion and limitations

- **Support size of the frame**: in our inverse problem, we approximate the aperture synthesis method as linear degradation embodied by a multiplication of the true data with a mask in the Fourier domain. Moreover, we carried out our study in the gridded space where all quantities are discrete set with constant frame support size. The mask, therefore, has the same support size as the sky. All frame support sizes between the sky, the mask, and the PSF cubes are tied together. Therefore, if one wants to change the support size, new training of the networks is required. This is contrary to some other DL-based image/video restoration problems where the linear degradation is defined by a convolution rather than by a mask. In this case, a trained neural network can deconvolve an image of any size. Regarding our networks, as only the 2D Net module operates deconvolution, we should only retrain this module, and 1D Net can be frozen. From a dataset of ungridded visibilities, it is up to the scientist to decide which support size fits the scientific objective of the data. It should be trained on support size that corresponds to the final products used (*e.g.*, image catalog of a survey, transient detection pipeline, etc.).

- **Total duration of transient events**: the total observation period and the number of temporal frames (which defined a slice PSF of the PSF cube) determine the upper

limits on the entire duration and the shortest timescale of transient events we deal with. As an illustration, for a fixed number of PSF frames over the total observation period, an increase in the latter will increase the integration time of each PSF frame. Thus, if the total observation period is more extended, with a fixed number of PSFs, each PSF will appear smoother and presents fewer secondary lobes. Therefore, each time the total observation period changes, our networks should be theoretically retrained with new PSF cubes. We can suppose that the profiles of transient events in the train sky cube are diverse enough to be representative of real transients that can be encountered. In this case, we can keep the same train sky set. Conversely, suppose the length of the observation increases for a fixed PSF time integration. In that case, the third dimension of the cube increases but each PSF will share the same characteristics as before. In that case, we should only retrain 1D Net, and 2D Net can be frozen as its role is to deconvolve PSFs similar to before.

- **Computational complexity**: once trained, our networks present faster reconstruction than frame-by-frame CLEAN. This is because contrary to CLEAN, they are not iterative algorithms and can benefit from efficient GPU-based architectures. This aspect makes them attractive to the radio astronomy community. On our Intel I9-10940X CPU and NVIDIA TITAN RTX GPU, 2D-1D Net reconstructs the sky cube of 32 frames in ~ 65 ms. This speed is ~ 75 ms for Deflation Net. Both of them contain 0.1 M parameters.

- **Theoretical guarantee**: one of the drawbacks of our DL-based methods is that they lack mathematical frameworks allowing us to derive theoretical error bound or uncertainty beyond empirical statistics on the results.

- **Extension to the polychromatic case**: it is possible to extend our proposed methods to the polychromatic case, where the inverse problem also has a frequency dependency. In this case, the dirty, GT and PSF cubes are 4-dimensional data structures, adding the frequency dimension. A simple way to realize this extension is to i) stack several skies at different frequencies in the channel dimension at the input of the 2D Net module and ii) stack several PSFs at different frequencies in the channel dimension at the input of the SFT layer. The computation of the average sky and PSF, and the subtraction step in the Deflation Net should be done for each frequency.

3.8 Conclusion

In this chapter, we dealt with transient source reconstruction in the context of radio interferometric imaging. We formulated this problem as a deconvolution of image time series. We proposed two neural networks to address this task, namely 2D-1D Net and Deflation Net. Thanks to the SFT layer, they can handle multiple PSFs that vary depending on the observed sky positions. They involve the same sub-modules: 2D Net and 1D Net. The former deconvolves individual frames, and the latter enables temporal sky modeling. 2D-1D Net is based on a simple feedforward inference, whereas Deflation Net involves different computational flows. The latter restores the average sky and uses it to isolate transient sources in individual frames. Experiments based on simulated data and metrics measuring temporal

profile reconstruction and background denoising demonstrated superior performances of these DL-based methods over CLEAN in the presence of noise. Deflation Net performs the best, excelling in reconstructing constant sources and background denoising. The ablation study confirms the temporal modeling enabled by 1D Net significantly increases the sky cube reconstruction performance.

Learning methods adapted to deconvolution, temporal structures of transient sources, and specificities of instrumental response are key elements to efficiently analyzing the images obtained via radio interferometers in the SKA era. For instance, the raw sensitivity of MeerKAT enables deep imaging (*i.e.*, a typical $\sigma \sim 10$ μJy in 15 min) as well as high cadence imaging capabilities in a large field of view. Being able to deconvolve the data in high noise regimes efficiently will maximize the chance to discover new transients and overcome the limitations imposed by current deconvolution methods. Missing transients in the image plane can be due to the lack of sensitivity (*i.e.*, detection problem) or the lack of sufficient temporal sampling (*i.e.*, dilution problem) that averages out short-scale transients. The robust 2D and 1D image reconstruction brought by the trained networks introduced in this chapter has significantly improved the 3D estimation of the sky. Replacing classical frame-by-frame deconvolution methods with the DL-based reconstruction methods can lead to better use of telescope time. This will drastically reduce the required exposure time while enabling a faster temporal sky sampling. Our vizualisations of transient source reconstructions are available at the following GitHub repository: https://github.com/bjmch/DL-RadioTransient.

Part III

Conclusion

Chapter 4

Conclusion and perspective

4.1 Conclusion

This thesis exposed how we can empower a DL-based image/video restoration model by giving it a priori knowledge adopted to the inverse problem, similarly to classical restoration methods. To begin with, a purely data-based restoration CNN learns to map the input degraded image (video) to its restored version based on training pairs of degraded and original images (videos). These pairs are generated based on the forward model on the inverse problem. This operating mode is different from the one of classical restoration methods. A classical method derives from the forward model a minimization problem regularized in a hand-crafted manner. This problem is mostly solved based on an iterative algorithm. DL-based methods present advantages and disadvantages compared to classical ones. First, the regularization they learn based on data better captures natural images (videos) statistics than the hand-crafted ones. They do not require expert knowledge and time, which are necessary when hand-crafting the regularizer and setting the regularization parameter. Second, DL-based methods are faster than the classical ones, as they do not rely on iterative algorithms and can benefit from efficient GPU architectures. However, their black box nature prevents them from being interpretable, which differs from classical methods that are explainable by construction. Furthermore, they lack flexibility in dealing with heterogenous degradations and cannot be suitable for some applications. More recent methods breach the gap between the two paradigms.

Regarding our contributions, we firstly focused on the problem of VSR. We presented how existing classical and DL-based approaches have solved this inverse problem and compared them in various situations. As our first contribution, we proposed UVSR, a new VSR neural network based on deep unrolling, a technique consisting of designing CNN architectures based on classical iterative algorithms. We measured and compared its performance to existing networks in some experimental settings and concluded that our newly introduced network is suitable when there are multiple degradations with constraints on inference speed and the number of parameters. As our second contribution, we pointed out for the first time the instability problem of recurrent VSR when facing long sequences with low motion. Existing recurrent VSR networks generate high-frequency artifacts on such sequences. As part of the proposed solution, we defined a new framework for recurrent VSR models based on Lipschitz stability theory. As an implementation of this framework, we proposed a new recurrent VSR network coined MRVSR that is more suited for long sequences with low motion than existing recurrent VSR models. We experimentally showed its stability and SOTA performance on long sequences with low motion. As part of our experiments, we introduced a new test dataset of such sequences: Quasi-Static Video Set.

As our third contribution, we designed the first NNs that deconvolve the time series of radio interferometry images to help reconstruct transient astronomical sources. They are sources that appear and disappear over time and are interesting for astrophysicists because they can be related to high-energy physical phenomena such as pulsars, rotating radio transients (RRATs), solar-system magnetized objects, and "fast radio bursts" (FRB). We proposed two new NNs that can manage the multiple PSFs of the instrument and operate spatial and temporal modelings. We showed their superior performance on our simulated data over CLEAN, the most used classical algorithm in the radio interferometry community.

4.2 Perspectives

4.2.1 VSR

In VSR, maximizing the temporal receptive field, *i.e.*, the number of input LR frames used to super-resolve an HR frame, can significantly increase the performance. However, we saw that this number is limited by the memory we can allocate at training time for sliding-based VSR methods. This number is increased for recurrent methods, but they, in turn, present the divergence issue on long sequences with low motion. Imposing the Lipschitz constraint solves this problem but reduces the temporal receptive field. We need a method to stock an arbitrary number of LR frames in memory without presenting any divergence issue.

In the restrictive case where all motions are translational, the first part of the shift-and-add algorithm, *i.e.*, the accumulation that estimates the blurred version of the HR frame, can fulfill this role. If the L2 distance is used for the data fidelity term, the pixelwise average of measurements after image registration can be computed recursively, enabling online processing of the input LR frame. [96] proposed to use a CNN for the subsequent deblurring stage. However, this approach's temporal receptive field is still limited by the memory allocated at training time. In case the number of accumulated data goes above the one used

at training time, the blur corrected in the subsequent stage differs from the ones observed during training time. There is still a domain adaptation problem.

To conclude, a new paradigm, different from RNN and sliding-window methods, should be proposed to stock useful information from an arbitrary number of LR frames in memory without presenting any divergence issue.

4.2.2 Unrolled VSR

Starting from the forward model of the VSR models that involves the warping operator based on the optical flow between two frames, we have derived an unrolled network. Instead of relying on optical flow, we can use alignment based on deformable convolution in its data step. As this alignment is more diverse than the optical flow-based one in terms of offset, this approach should give better results. However, the alignment occurs in the HR space. Therefore, we can encounter a memory issue with the deformable convolution at training time. Parallel computing could be necessary in this case.

4.2.3 Radio interferometry

Our proposed networks take as input dirty images and output restored sky models. They, therefore, exclusively work in the image domain. Another approach worth trying is designing CNNs that work in the Fourier domain, taking sampled amplitudes and phase distributions as inputs and applying inpainting on them. Restored skies can then be computed by inverse Fourier transform. Studies like [53] operated DL-based reconstructions in the Fourier domain in their MRI reconstruction problem. Regarding single radio image deconvolution, [113] recently proposed to do the inpainting in the Fourier domain. This idea should be extended to the inpainting of the input time series of Fourier plans.

4.2.4 General image and video restoration problems

Recently, transformers have been pushing the limit of image and video restoration performance [74, 19, 73, 75, 137, 153]. However, these transformer-based models are again used in an ideal single degradation scenario and given only degraded data at their input. They lack flexibility in dealing with multiple degradations. Enabling this flexibility by non-blindly giving the knowledge about degradation to the input would be an important future work. In this case, one should design how the knowledge about degradation can enable spatial transformation within these transformer-based models. Moreover, combining a transformer-based architecture with frameworks such as deep unfolding, plug-and-play, RED or DIP would be an interesting approach. As transformer-based architectures are highly memory-demanding, they would benefit from parallel computing. Moreover, progress in GPU, parallel, and high-performance computing would undoubtedly help the image and video restoration community.

Appendix A

Code, data and other materials

A.1 Code, data and other materials related to MRVSR

Please find our code and instructions in the following GitHub repository: https://github.com/bjmch/MRVSR. The repository also contains links to download the proposed dataset, network weights and videos reconstructed from different networks.

A.2 Vizualisations of transient source reconstructions

Please find our vizualisations of transient source reconstructions in the following GitHub repository: https://github.com/bjmch/DL-RadioTransient.

A.3 Training settings of SOTA VSR networks

Tab. A.1 summarizes training settings of SOTA VSR networks. All of them rely on Adam momentum optimization.

Model	Machine(s)	LR size	Batch size	Initial lr	BPTT length	loss
RLSP	NVIDIA Titan XP (12 GB)	64	4	10^{-4}	10	MSE
RSDN	8 Nvidia Tesla V100 GPUs (16 GB or 32 GB)	64	16	10^{-4}	7	Charbonnier
FRVSR	NVIDIA P100	64	4	10^{-4}	10	L2
EDVR	8 NVIDIA Titan Xp 12 GB	64	32	10^{-4}	-	Charbonnier
PFNL	NVIDIA GeForce GTX 1080Ti 11 GB	32	16	10^{-3}	-	Charbonnier
DUF	NVidia GeForce GTX 1080 Ti 11 GB	32	16	10^{-4}	-	Huber

Table A.1: Training settings of SOTA VSR networks. "lr" means learning rate. All of them used Adam optimization.

A.4 Architecture of FNet

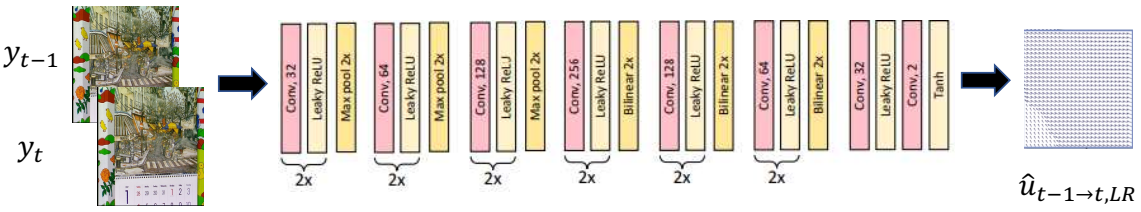


Figure A.1: Architecture of FNet. The input LR frames y_t and y_{t-1} are concatenated in the channel dimension and given to the encoder/decoder style architecture. 2x indicates the corresponding block is duplicated. All convolutions use 3×3 kernels with stride 1. Figure adapted from [110].

Fig. A.1 illustrates the architecture of FNet.

A.5 Coordinates of an astrophysical source

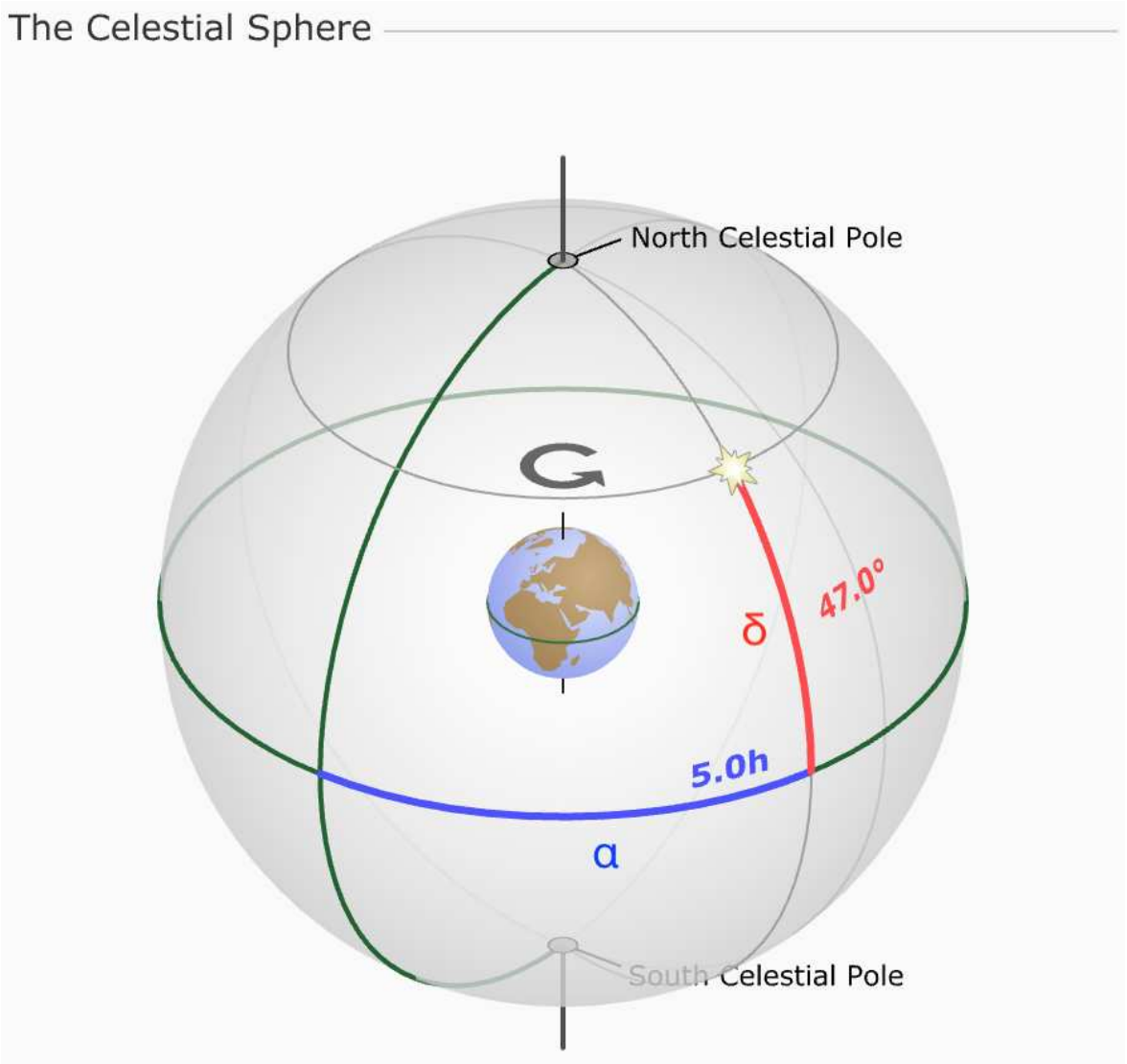


Figure A.2: An astrophysical source is associated to a unique pair of coordinates (declination δ and right ascension α) on the celestial sphere. Figure adapted from [7].

Fig. A.2 illustrates the unique pair of coordinates (declination δ, right ascension α) that is associated with an astrophysical source.

Appendix B

Other experimental results on recurrent VSR

B.1 Forgetting capability of recurrent VSR networks

As stated in Sec. 2.5.5, in the presence of strong motion, even with short-term training, a recurrent VSR network learns to forget the past information, which is inconsistent with the new one. The newly created high-frequency content is forgotten at the same time, preventing divergence on scenes with enough motion. Apart from the example of the frequently-moving bird in the first sequence of the Quasi-Static Video Set in Sec. 2.5.5, the following experimentally illustrate this forgetting capability.

REDS4 is a public test dataset of 4 sequences of 100 frames used in [134, 97]. These sequences present fast scene motion with a rather largely moving camera and a frame rate that is not so high (120fps) [91]. We take our RLSP implementation detailed in Sec. 2.5.4 and run it several times on REDS4. Each run starts at different offsets, *i.e.*, time step. Fig. B.1 numerically compares these runs. We observe that delayed runs catch up with the performance of the baseline one (the one from the first frame) rather rapidly (after around 5 to 15 frames). This means that about 5 to 15 frames after each of these offsets, the recurrent model that started the inference from the beginning uses no more information older than this offset. This information has been forgotten.

As part of another experiment, we center-crop with a crop size of 256 all of the four

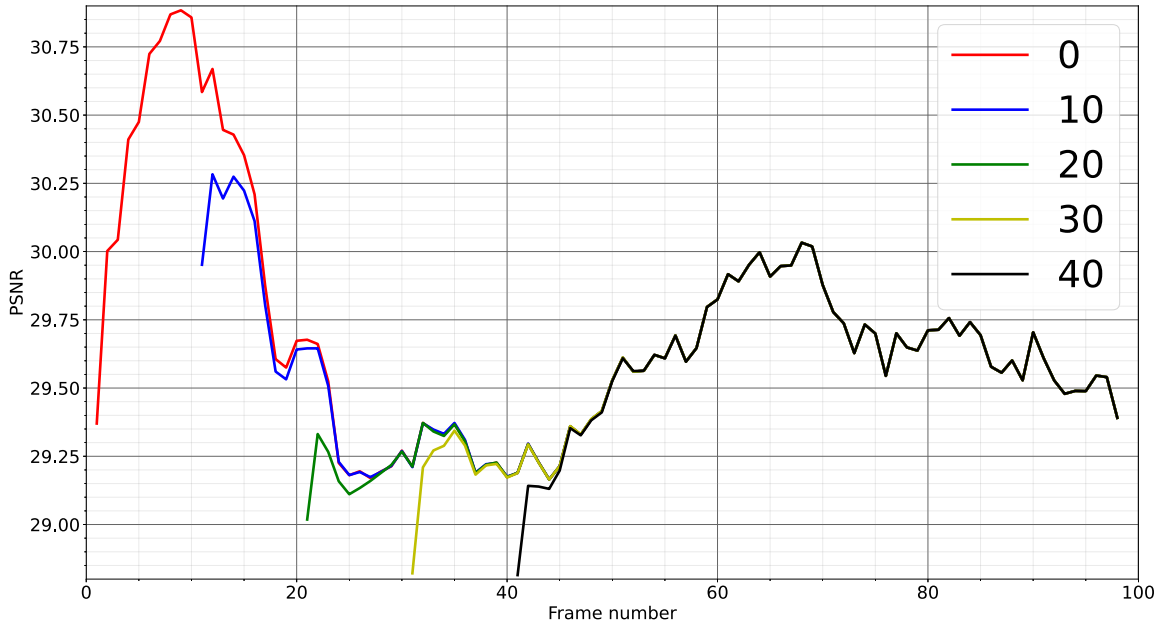
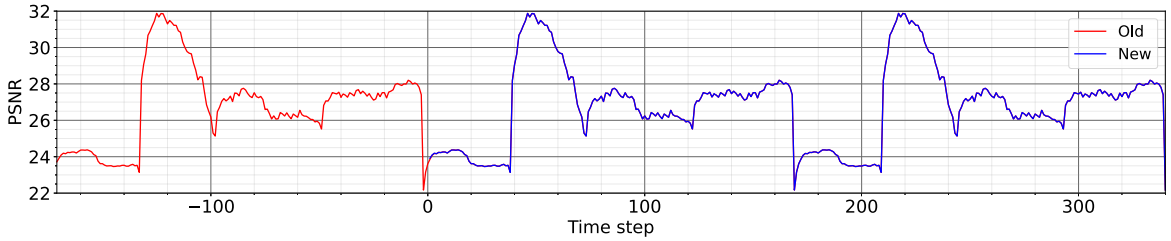
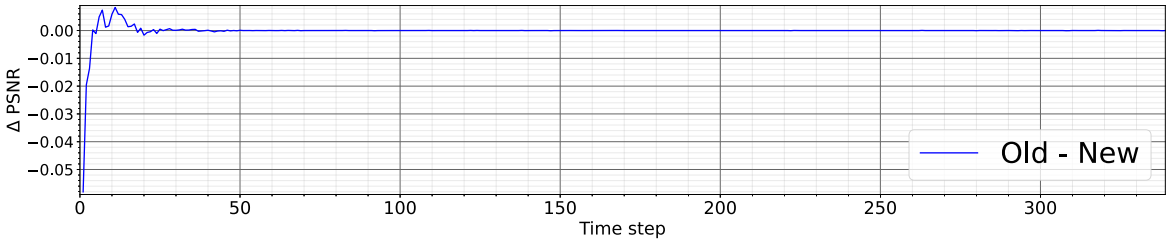


Figure B.1: Mean PSNR of our RLSP runs on the brightness channel of the RED4 test dataset [134, 97]. Numbers in the legend indicate at which time step the inference has started.



(a) Mean PSNR.



(b) Subtraction of the PSNR of the older run by the PSNR of the newer one.

Figure B.2: Comparison of an older run and the newer one of our RLSP on a long sequence formed by repeating a concatenation of the four Vid4 sequences. The newer run has started a concatenation after the older one.

videos of the Vid4 [77] dataset and concatenate them in the time dimension. We concatenate in the following order—*calendar*, *city*, *foliage*, *work*—and repeated this motif several times to form a long sequence. The latter thus contains brutal scene change with total inconsistency between successive scenes. A motif has a length of around.170 frames. We run our RLSP two times again on this sequence. Fig. B.2a compares an older run and the newer one that has started a motif after, in terms of PSNR on the brightness channel. Fig. B.2b visualizes the substraction of the PSNR of the older one by the PSNR of the newer one. We see that after around 50 frames, the newer run catches up with the older one. This means that about 50 frames after the beginning of the newer run, the older model uses no more information older than the beginning of the newer run.

B.2 About SRN-C

Algorithm. 5 details the SRN-C algorithm [123]. It involves an image size hyperparameter n [123]. This quantity should theoretically be equal or bigger than the image size used during inference to guarantee stability. This is because the matrix of a convolution applied to an image is different and bigger than the matrix of a convolution applied to a bigger image. In practice, n should be in the order of the image size at inference time. Note that n is not constrained by the image size at training time. In our experiments, we set $n = 128$.

Algorithm 5: *SRN-C–α–β* [123]

Input: Number of iterations N, learning rate η,
number of channels m, image size n,
initial $\boldsymbol{K} \in \mathbb{R}^{k\times k\times m\times m}$, initial $\boldsymbol{u} \in \mathbb{R}^{n\times n\times m}$.
Parameters: Spectral norm α, stable rank β.
begin
  for $i = 1, \dots, N$ **do**
    $\widetilde{\boldsymbol{K}} = \boldsymbol{K}$
    Power iteration:
    $\boldsymbol{v} = \widetilde{\boldsymbol{K}}^{*}\boldsymbol{u} / \|\widetilde{\boldsymbol{K}}^{*}\boldsymbol{u}\|_2$
    $\boldsymbol{u} = \widetilde{\boldsymbol{K}} * \boldsymbol{v} / \|\widetilde{\boldsymbol{K}} * \boldsymbol{v}\|_2$
    Spectral normalization:
    $\widetilde{\boldsymbol{K}} = \widetilde{\boldsymbol{K}} / (\boldsymbol{u}^{(}\widetilde{\boldsymbol{K}} * \boldsymbol{v}) + \varepsilon)$
    Stable rank ($\beta < 1$):
    $\boldsymbol{S}_1 = \nabla_{\widetilde{\boldsymbol{K}}}(\boldsymbol{u}^{(}\widetilde{\boldsymbol{K}} * \boldsymbol{v}))$
    $\boldsymbol{S}_2 = \widetilde{\boldsymbol{K}} - \boldsymbol{S}_1$
    $\gamma = \sqrt{\beta m - 1/n^2} / \|\boldsymbol{S}_2\|_F$
    if $\gamma < 1$ **then**
      $\widetilde{\boldsymbol{K}} = \boldsymbol{S}_1 + \gamma \boldsymbol{S}_2$
    end

    Training step:
    $\boldsymbol{K} = \boldsymbol{K} - \eta \nabla_{\boldsymbol{K}} L(\alpha\, \widetilde{\boldsymbol{K}})$
  end
end

B.3 A new diagnostic tool to evaluate the stability of a recurrent VSR model on sequences with low motion

We propose a simple yet effective way to diagnose the divergence of recurrent VSR models on long sequences with low motion diversity. This consists in running the analyzed VSR network on a static noise sequence. We run some of our networks on a sequence of 1000 same noise images with size 512×512. Figure B.3 shows evolutions of the output's L2 norm with increasing time step for different networks. MRVSR-SL is a model based on the same architecture as MRVSR but under SL with hyperparameters of SRN-C being $(\alpha, \beta) = (2.0, 0.1)$. We see that recurrent networks without constraint or with the soft Lipschitz constraint give diverging outputs, but MRVSR does not. This result is in accordance with the stability introduced by the contractive recurrence map. Moreover, this confirms again that SL is not enough to prevent divergence of recurrent VSR model on static sequences. Finally, this confirms what we stated in section B.2 about the image size hyperparameter n of SRN-C: even if we trained MRVSR HL with $n = 128$, this model is stable on a static sequence of size 512×512. We think that in this case, the value of $n = 128$ is large enough,

taking into account that SRN [111] that fails to set spectral norms of convolutional layers to α in the work [123] can be seen as doing SRN-C with $n = 1$.

Fig. B.4 additionally illustrate the fact that MRVSR-SL still diverges on long sequences with low motion. This confirms again that SL is not enough to prevent the instabilities.

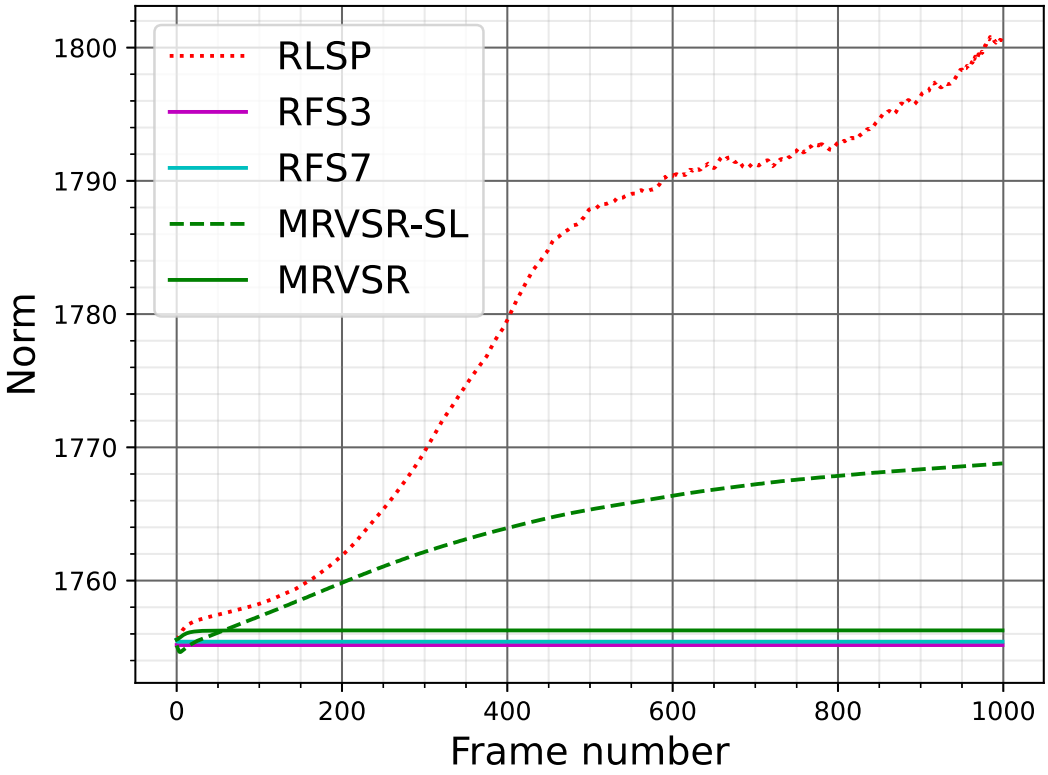


Figure B.3: Evolutions of the L2 norm of outputs when running different networks on a static sequence of a noise image of size 512×512.

B.4 Other remarks

We stated that SRN-C with $\beta = 1$, was equivalent to setting the spectral norm of the matrix of the convolutional layer to a desired value α. We noticed after our work that the same algorithm as this case of SRN-C was also proposed in [108], coined *real spectral normalization*.

Figure B.4: MRVSR SL

Appendix C

Ideas that did not work

VSR with deep unrolling and recurrent 1-Lipschitz data module

Some deep unrolling approaches factorized the network into data and denoising prior steps [156]. As we stated in Sec. 2.5.3, a denoising function can be 1-Lipschitz while correctly performing. Besides, we demonstrated that a recurrent VSR network is stable on long sequences with low motion when the Lipschitz constant of its recurrent mapping is below 1. A subsequent idea, therefore, was to combine the ideas from Secs 2.4 and 2.5 to design an unrolled VSR network with a recurrent and 1-Lipschitz prior module.

However, this approach resulted in a poor-performing network. The following results illustrate this point. First, on the Set5 test set with zoom factor $s = 4$ and standard deviation of the gaussian blur equaling 1.5, an USRNet model free of any constraint and an USRNet model with Hard Lipchitz constraint (HL) on its prior module respectively recorded the mean PSNR of 29.16 and 28.91 dB. The gap between these values was 0.25dB, which showed the performance drop provoked by the HL. This gap was smaller than the one between MRVSR and RLSP on Vid4, reported in Tab. 2.4 (the gap of 0.56dB). This observation on the SISR problem was rather promising because the HL did not provoke a substantial performance drop.

Next, we implemented a variant of USRNet adapted to the VSR problem. The latter had the same U-Net-based prior model as USRNet, but this model was made recurrent with an RLSP-like mechanism (recurrent hidden state and output). We prepared this model and another version with HL on its U-Net prior. We note that the Lipschitz constants of the downsampling and upsampling operations used in the U-Net are equal to 1. We also note that the HL introduces additional parameters to be trained during training. We unrolled both models for 4 iterations and tuned filter numbers for the U-Net prior to achieving the limit of possible memory consumption during the training with a batch size of 4 and the NVIDIA GeForce RTX 3090 GPU (24GB). These filter numbers of the U-Net were 16, 32, 64, and 64 on the downsampling branch (the ones of the upsampling branch are symmetric).

As a result, the constraint-free version of this VSR model recorded a mean PSNR on the brightness channel of Vid4 of 27.29dB. This performance is below RLSP. Besides, the one

with HL recorded a PSNR of 26.69dB, which is below the performance of MRVSR. The gap between these values is 0.6dB, slightly higher than between RLSP and MRVSR. This gap shows that the performance drop from the 1-Lipschitz recurrent U-Net prior in the VSR variant of USRNet was significant. Moreover, the VSR variant of USRNet was slower than MRVSR and RLSP.

Publications

Published

- Chiche, B. N., Frontera-Pons, J., Woiselle, A., & Starck, J. L. (2020, November). Deep Unrolled Network for Video Super-Resolution. In *2020 Tenth International Conference on Image Processing Theory, Tools and Applications (IPTA)* (pp. 1-6). IEEE.
- Chiche, B. N., Woiselle, A., Frontera-Pons, J., & Starck, J. L. (2022). Stable Long-Term Recurrent Video Super-Resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 837-846).

Submitted

- Chiche, B. N., Girard, J. N, Frontera-Pons, J., Woiselle, A., & Starck, J. L. (2022). Deep Learning-Based Deconvolution for Interferometric Radio Transient Reconstruction. *Submitted to Astronomy & Astrophysics*.

Communications

- Chiche, B. N., Woiselle, A., & Budin, J. (2022, June). Recent development for video restoration using simulation and learning. In *the 10th International Symposium on OPTRONICS IN DEFENCE AND SECURITY*. OPTRO 2022.
- Chiche, B. N., Girard, J. N, Frontera-Pons, J., Woiselle, A., & Starck, J. L. (2022, September). Apprentissage profond appliqué à la reconstruction interférométrique de transitoires radio. In *GRETSI 2022-XXVIIIème Colloque francophone de traitement du signal et des images* (p. 1269-1272). GRETSI.

Bibliography

[1] F. Abrantes et al. “Proxy calibration to instrumental data set: Implications for paleoceanographic reconstructions”. In: *Geochemistry, Geophysics, Geosystems* 10.9, Q09U07 (Sept. 2009), Q09U07. DOI: 10.1029/2009GC002604. arXiv: 0806.2228 [astro-ph].

[2] Jonas Adler and Ozan Öktem. “Learned primal-dual reconstruction”. In: *IEEE transactions on medical imaging* 37.6 (2018), pp. 1322–1332.

[3] Manya V. Afonso, José M. Bioucas-Dias, and Mário A. T. Figueiredo. “Fast Image Recovery Using Variable Splitting and Constrained Optimization”. In: *IEEE Transactions on Image Processing* 19.9 (2010), pp. 2345–2356.

[4] Hemant K Aggarwal, Merry P Mani, and Mathewsnumerical optimization within Jacob. “Modl: Model-based deep learning architecture for inverse problems”. In: *IEEE transactions on medical imaging* 38.2 (2018), pp. 394–405.

[5] Eirikur Agustsson and Radu Timofte. “NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2017.

[6] Martin Arjovsky, Amar Shah, and Yoshua Bengio. “Unitary evolution recurrent neural networks”. In: *International Conference on Machine Learning*. PMLR. 2016, pp. 1120–1128.

[7] UNL Astronomy. *Celestial-Equatorial (RA/Dec) Demonstrator*. URL: http://astro.unl.edu/classaction/animations/coordsmotion/radecdemo.html (visited on 09/01/2022).

[8] Haoran Bai and Jinshan Pan. “Self-Supervised Deep Blind Video Super-Resolution”. In: *arXiv preprint arXiv:2201.07422* (2022).

[9] Francois Bergeaud and Stephane Mallat. “Matching pursuit of images”. In: *Proceedings., International Conference on Image Processing*. Vol. 1. IEEE. 1995, pp. 53–56.

[10] Gedas Bertasius, Lorenzo Torresani, and Jianbo Shi. “Object detection in video with spatiotemporal sampling networks”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 331–346.

[11] Carla Bertocchi et al. “Deep unfolding of a proximal interior point method for image restoration”. In: *Inverse Problems* (2019).

[12] Marco Bevilacqua et al. “Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding”. In: *Proceedings of the British Machine Vision Conference*. BMVA Press, 2012, pp. 135.1–135.10. ISBN: 1-901725-46-4. DOI: http://dx.doi.org/10.5244/C.26.135.

[13] Yochai Blau and Tomer Michaeli. “The Perception-Distortion Tradeoff”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[14] RS Booth and JL Jonas. “An overview of the MeerKAT project”. In: *African Skies* 16 (2012), p. 101.

[15] Alan C Bovik. *Handbook of image and video processing*. Academic press, 2010.

[16] Stephen Boyd et al. 2011.

[17] Jose Caballero et al. “Real-Time Video Super-Resolution With Spatio-Temporal Networks and Motion Compensation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[18] G.C. Calafiore. “Outliers robustness in multivariate orthogonal regression”. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 30.6 (2000), pp. 674–679. DOI: 10.1109/3468.895890.

[19] Mingdeng Cao et al. “Vdtr: Video deblurring with transformer”. In: *arXiv preprint arXiv:2204.08023* (2022).

[20] Lawrence Carin. “On the relationship between compressive sensing and random sensor arrays”. In: *IEEE Antennas and Propagation Magazine* 51.5 (2009), pp. 72–81.

[21] Rafael E Carrillo, Jason D McEwen, and Yves Wiaux. “Sparsity averaging reweighted analysis (SARA): a novel algorithm for radio-interferometric imaging”. In: *Monthly Notices of the Royal Astronomical Society* 426.2 (2012), pp. 1223–1234.

[22] Antonin Chambolle and Thomas Pock. “A first-order primal-dual algorithm for convex problems with applications to imaging”. working paper or preprint. June 2010. URL: https://hal.archives-ouvertes.fr/hal-00490826.

[23] Kelvin C.K. Chan et al. “BasicVSR++: Improving Video Super-Resolution With Enhanced Propagation and Alignment”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 5972–5981.

[24] Kelvin C.K. Chan et al. “BasicVSR: The Search for Essential Components in Video Super-Resolution and Beyond”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 4947–4956.

[25] Kelvin C.K. Chan et al. “Understanding Deformable Alignment in Video Super-Resolution”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.2 (2021), pp. 973–981. URL: https://ojs.aaai.org/index.php/AAAI/article/view/16181.

[26] Dongdong Chen et al. “Coherent Online Video Style Transfer”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[27] Yunjin Chen and Thomas Pock. “Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (2016), pp. 1256–1272.

[28] Benjamin Naoto Chiche et al. “Stable Long-Term Recurrent Video Super-Resolution”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 837–846.

[29] Mengyu Chu et al. “Learning temporal coherence via self-supervision for GAN-based video generation”. In: *ACM Transactions on Graphics* 39 (July 2020). DOI: 10.1145/3386569.3392457.

[30] BG Clark. “An efficient implementation of the algorithm’CLEAN’”. In: *Astronomy and Astrophysics* 89 (1980), p. 377.

[31] M-C Corbineau et al. “Learned Image Deblurring by Unfolding a Proximal Interior Point Algorithm”. In: *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2019, pp. 4664–4668.

[32] Arwa Dabbech et al. “Moresane: Model reconstruction by synthesis-analysis estimators-a sparse deconvolution algorithm for radio interferometric imaging”. In: *Astronomy & Astrophysics* 576 (2015), A7.

[33] Jifeng Dai et al. “Deformable Convolutional Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[34] Aram Danielyan, Vladimir Katkovnik, and Karen Egiazarian. “Image deblurring by augmented lagrangian with bm3d frame prior”. In: *Workshop on Information Theoretic Methods in Science and Engineering* (Jan. 2010).

[35] P. E. Dewdney et al. “The Square Kilometre Array”. In: *IEEE Proceedings* 97.8 (Aug. 2009), pp. 1482–1496. DOI: 10.1109/JPROC.2009.2021005.

[36] Steven Diamond et al. “Unrolled optimization with deep priors”. In: *arXiv preprint arXiv:1705.08041* (2017).

[37] M. Elad and Y. Hel-Or. “A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur”. In: *21st IEEE Convention of the Electrical and Electronic Engineers in Israel. Proceedings (Cat. No.00EX377)*. 2000, pp. 402–405. DOI: 10.1109/EEEI.2000.924450.

[38] Gunnar Farnebäck. “Two-Frame Motion Estimation Based on Polynomial Expansion”. In: *Image Analysis*. Ed. by Josef Bigun and Tomas Gustavsson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 363–370. ISBN: 978-3-540-45103-7.

[39] Sina Farsiu, Michael Elad, and Peyman Milanfar. “Video-to-video dynamic super-resolution for grayscale and color sequences”. In: *EURASIP Journal on Advances in Signal Processing* 2006.1 (2006), p. 061859.

[40] Sina Farsiu et al. “Fast and robust multiframe super resolution”. In: *IEEE transactions on image processing* 13.10 (2004), pp. 1327–1344.

[41] B Roy Frieden. “Restoring with maximum likelihood and maximum entropy”. In: *JOSA* 62.4 (1972), pp. 511–518.

[42] Dario Fuoli, Shuhang Gu, and Radu Timofte. “Efficient Video Super-Resolution through Recurrent Latent Space Propagation”. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. 2019, pp. 3476–3485.

[43] Hugh Garsden et al. “LOFAR sparse image reconstruction”. In: *Astronomy & astrophysics* 575 (2015), A90.

[44] Pascal Getreuer. “Total Variation Deconvolution using Split Bregman”. In: *Image Processing On Line* 2 (2012). https://doi.org/10.5201/ipol.2012.g-tvdc, pp. 158–174.

[45] Julien N Girard et al. “Sparse representations and convex optimization as tools for LOFAR radio interferometric imaging”. In: *Journal of Instrumentation* 10.08 (2015), p. C08013.

[46] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[47] Tom Goldstein and Stanley Osher. “The Split Bregman Method for L1-Regularized Problems”. In: *SIAM Journal on Imaging Sciences* 2.2 (2009), pp. 323–343. DOI: 10.1137/080725891. eprint: https://doi.org/10.1137/080725891. URL: https://doi.org/10.1137/080725891.

[48] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and Harnessing Adversarial Examples”. In: *International Conference on Learning Representations*. 2015. URL: http://arxiv.org/abs/1412.6572.

[49] Ian Goodfellow et al. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[50] Henry Gouk et al. “Regularisation of neural networks by enforcing lipschitz continuity”. In: *Machine Learning* 110.2 (2021), pp. 393–416.

[51] Jinjin Gu et al. “Blind Super-Resolution With Iterative Kernel Correction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[52] Michael P van Haarlem et al. “LOFAR: The low-frequency array”. In: *Astronomy & astrophysics* 556 (2013), A2.

[53] Yoseo Han, Leonard Sunwoo, and Jong Chul Ye. “k -Space Deep Learning for Accelerated MRI”. In: *IEEE Transactions on Medical Imaging* 39.2 (2020), pp. 377–386.

[54] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[55] Geoffrey E Hinton et al. “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580* (2012).

[56] JA Högbom. “Aperture synthesis with a non-regular distribution of interferometer baselines”. In: *Astronomy and Astrophysics Supplement Series* 15 (1974), p. 417.

[57] Jie Hu, Li Shen, and Gang Sun. "Squeeze-and-Excitation Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[58] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. "A lightweight optical flow CNN—Revisiting data fidelity and regularization". In: *IEEE transactions on pattern analysis and machine intelligence* 43.8 (2020), pp. 2555–2569.

[59] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. "LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[60] Eddy Ilg et al. "FlowNet 2.0: Evolution of Optical Flow Estimation With Deep Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[61] Michal Irani and Shmuel Peleg. "Motion Analysis for Image Enhancement: Resolution, Occlusion, and Transparency". In: *Journal of Visual Communication and Image Representation* 4.4 (1993), pp. 324–335. ISSN: 1047-3203. DOI: https://doi.org/10.1006/jvci.1993.1030. URL: https://www.sciencedirect.com/science/article/pii/S1047320383710308.

[62] Takashi Isobe et al. "Video super-resolution with recurrent structure-detail network". In: *European Conference on Computer Vision*. Springer. 2020, pp. 645–660.

[63] Takashi Isobe et al. "Video Super-Resolution With Temporal Group Attention". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[64] Max Jaderberg et al. "Spatial Transformer Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/33ceb07bf4eeb3da587e268d663aba1a-Paper.pdf.

[65] Kyong Hwan Jin et al. "Deep convolutional neural network for inverse problems in imaging". In: *IEEE Transactions on Image Processing* 26.9 (2017), pp. 4509–4522.

[66] Younghyun Jo et al. "Deep Video Super-Resolution Network Using Dynamic Upsampling Filters Without Explicit Motion Compensation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[67] Cijo Jose, Moustapha Cissé, and Francois Fleuret. "Kronecker recurrent units". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2380–2389.

[68] A. Kappeler et al. "Video Super-Resolution With Convolutional Neural Networks". In: *IEEE Transactions on Computational Imaging* 2.2 (2016), pp. 109–122.

[69] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. "Accurate Image Super-Resolution Using Very Deep Convolutional Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[70] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1412.6980.

[71] Wei-Sheng Lai et al. “Deep Laplacian Pyramid Networks for Fast and Accurate Super-Resolution”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[72] Christian Ledig et al. “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[73] Jingyun Liang et al. “Recurrent Video Restoration Transformer with Guided Deformable Attention”. In: *arXiv preprint arXiv:2206.02146* (2022).

[74] Jingyun Liang et al. “SwinIR: Image Restoration Using Swin Transformer”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 2021, pp. 1833–1844.

[75] Jingyun Liang et al. “Vrt: A video restoration transformer”. In: *arXiv preprint arXiv:2201.12288* (2022).

[76] Bee Lim et al. “Enhanced Deep Residual Networks for Single Image Super-Resolution”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2017.

[77] C. Liu and D. Sun. “A Bayesian approach to adaptive video super resolution”. In: *CVPR 2011*. 2011, pp. 209–216.

[78] Xiaohong Liu et al. “End-To-End Trainable Video Super-Resolution Based on a New Mechanism for Implicit Motion Estimation and Compensation”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2020.

[79] Ziwei Liu et al. “Video Frame Synthesis Using Deep Voxel Flow”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[80] D. R. Lorimer et al. “A Bright Millisecond Radio Burst of Extragalactic Origin”. In: *Science* 318.5851 (2007), pp. 777–780. DOI: 10.1126/science.1147532. eprint: https://www.science.org/doi/pdf/10.1126/science.1147532. URL: https://www.science.org/doi/abs/10.1126/science.1147532.

[81] Morteza Mardani et al. “Neural Proximal Gradient Descent for Compressive Imaging”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS’18. Montréal, Canada: Curran Associates Inc., 2018, 9596–9606.

[82] Ivan Marti-Vidal. “APSYNSIM: An Interactive Tool To Learn Interferometry”. In: *arXiv preprint arXiv:1706.00936* (2017).

[83] D. Martin et al. “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics”. In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 2. 2001, 416–423 vol.2. DOI: 10.1109/ICCV.2001.937655.

[84] Gary Mataev, Peyman Milanfar, and Michael Elad. “DeepRED: Deep Image Prior Powered by RED”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 2019.

[85] Tim Meinhardt et al. “Learning Proximal Operators: Using Denoising Networks for Regularizing Inverse Imaging Problems”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[86] Zakaria Mhammedi et al. “Efficient orthogonal parametrisation of recurrent neural networks using householder reflections”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 2401–2409.

[87] John Miller and Moritz Hardt. “Stable Recurrent Models”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=Hygxb2CqKm.

[88] Dennis Mitzel et al. “Video super resolution using duality based tv-l 1 optical flow”. In: *Joint Pattern Recognition Symposium*. Springer. 2009, pp. 432–441.

[89] Takeru Miyato et al. “Spectral Normalization for Generative Adversarial Networks”. In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=B1QRgziT-.

[90] Vishal Monga, Yuelong Li, and Yonina C. Eldar. “Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing”. In: *IEEE Signal Processing Magazine* 38.2 (2021), pp. 18–44. DOI: 10.1109/MSP.2020.3016905.

[91] Seungjun Nah et al. “NTIRE 2019 Challenge on Video Deblurring and Super-Resolution: Dataset and Study”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019.

[92] F. Nammour et al. “ShapeNet: Shape Constraint for Galaxy Image Deconvolution”. In: *in press*, arXiv:2203.07412 (Mar. 2022), arXiv:2203.07412. arXiv: 2203.07412 [astro-ph.IM].

[93] Benjamin Naoto Chiche et al. “Deep Unrolled Network for Video Super-Resolution”. In: *2020 Tenth International Conference on Image Processing Theory, Tools and Applications (IPTA)*. 2020, pp. 1–6. DOI: 10.1109/IPTA50016.2020.9286636.

[94] Ramesh Narayan and Rajaram Nityananda. “Maximum entropy image restoration in astronomy”. In: *Annual review of astronomy and astrophysics* 24.1 (1986), pp. 127–170.

[95] Michael Ng, Pierre Weiss, and Xiao-Ming yuan. “Solving Constrained Total-variation Image Restoration and Reconstruction Problems via Alternating Direction Methods”. In: *SIAM Journal on Scientific Computing* 32.5 (Jan. 2010), pp. 2710–2736. DOI: 10.1137/090774823. URL: https://hal.inria.fr/hal-03101413.

[96] Ngoc Long Nguyen et al. “Self-Supervised Multi-Image Super-Resolution for Push-Frame Satellite Images”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2021, pp. 1121–1131.

[97] Jinshan Pan et al. “Deep Blind Video Super-Resolution”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 4811–4820.

[98] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks”. In: *International conference on machine learning*. PMLR. 2013, pp. 1310–1318.

[99] Adam Paszke et al. “Automatic differentiation in pytorch”. In: (2017).

[100] Nelly Pustelnik et al. “Wavelet-based Image Deconvolution and Reconstruction”. In: *Wiley Encyclopedia of Electrical and Electronics Engineering* (2016). URL: https://hal.archives-ouvertes.fr/hal-01164833.

[101] Anurag Ranjan and Michael J. Black. “Optical Flow Estimation Using a Spatial Pyramid Network”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[102] Urvashi Rau and Tim J Cornwell. “A multi-scale multi-frequency deconvolution algorithm for synthesis imaging in radio interferometry”. In: *Astronomy & Astrophysics* 532 (2011), A71.

[103] Edward T. Reehorst and Philip Schniter. “Regularization by Denoising: Clarifications and New Interpretations”. In: *IEEE Transactions on Computational Imaging* 5.1 (2019), pp. 52–67. DOI: 10.1109/TCI.2018.2880326.

[104] Yaniv Romano, Michael Elad, and Peyman Milanfar. “The Little Engine That Could: Regularization by Denoising (RED)”. In: *SIAM Journal on Imaging Sciences* 10.4 (2017), pp. 1804–1844. DOI: 10.1137/16M1102884. eprint: https://doi.org/10.1137/16M1102884. URL: https://doi.org/10.1137/16M1102884.

[105] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4.

[106] Stefan Roth and Michael J Black. “Fields of experts”. In: *International Journal of Computer Vision* 82.2 (2009), pp. 205–229.

[107] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms". In: *Physica D: Nonlinear Phenomena* 60.1 (1992), pp. 259–268. ISSN: 0167-2789. DOI: https://doi.org/10.1016/0167-2789(92)90242-F. URL: https://www.sciencedirect.com/science/article/pii/016727899290242F.

[108] Ernest Ryu et al. "Plug-and-Play Methods Provably Converge with Properly Trained Denoisers". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 5546–5557. URL: https://proceedings.mlr.press/v97/ryu19a.html.

[109] Mehdi S. M. Sajjadi, Bernhard Scholkopf, and Michael Hirsch. "EnhanceNet: Single Image Super-Resolution Through Automated Texture Synthesis". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[110] Mehdi S. M. Sajjadi, Raviteja Vemulapalli, and Matthew Brown. "Frame-Recurrent Video Super-Resolution". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[111] Amartya Sanyal, Philip H. Torr, and Puneet K. Dokania. "Stable Rank Normalization for Improved Generalization in Neural Networks and GANs". In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=H1enKkrFDB.

[112] AMM Scaife. "Big telescope, big data: towards exascale with the Square Kilometre Array". In: *Philosophical Transactions of the Royal Society A* 378.2166 (2020), p. 20190060.

[113] Kevin Schmidt et al. "Deep Learning-based Imaging in Radio Interferometry". In: *arXiv preprint arXiv:2203.11757* (2022).

[114] Christian J. Schuler et al. "A Machine Learning Approach for Non-blind Image Deconvolution". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[115] FR Schwab. "Relaxing the isoplanatism assumption in self-calibration; applications to low-frequency radio interferometry". In: *The Astronomical Journal* 89 (1984), pp. 1076–1081.

[116] Hanie Sedghi, Vineet Gupta, and Philip M. Long. "The Singular Values of Convolutional Layers". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=rJevYoA9Fm.

[117] Wenzhe Shi et al. "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[118] Assaf Shocher, Nadav Cohen, and Michal Irani. ""Zero-Shot" Super-Resolution Using Deep Internal Learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[119] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *arXiv 1409.1556* (Sept. 2014).

[120] Deqing Sun et al. "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[121] Florent Sureau, Alexis Lechat, and J-L Starck. "Deep learning for a space-variant deconvolution in galaxy surveys". In: *Astronomy & Astrophysics* 641 (2020), A67.

[122] John D. Swinbank et al. "The LOFAR Transients Pipeline". In: *Astronomy and Computing* 11 (June 2015), pp. 25–48. DOI: 10.1016/j.ascom.2015.03.002. arXiv: 1503.01526 [astro-ph.IM].

[123] Thomas Tanay et al. "Diagnosing and Preventing Instabilities in Recurrent Video Processing". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), pp. 1–1. DOI: 10.1109/TPAMI.2022.3160350.

[124] Xin Tao et al. "Detail-Revealing Deep Video Super-Resolution". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017.

[125] Matthieu Terris et al. "Image reconstruction algorithms in radio interferometry: from handcrafted to learned denoisers". In: *arXiv preprint arXiv:2202.12959* (2022).

[126] Yapeng Tian et al. "TDAN: Temporally-Deformable Alignment Network for Video Super-Resolution". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[127] Radu Timofte et al. "NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2017.

[128] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Deep Image Prior". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[129] A Vafaei Sadr et al. "DEEPSOURCE: point source detection using deep learning". In: *Monthly Notices of the Royal Astronomical Society* 484.2 (2019), pp. 2793–2806.

[130] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. "Plug-and-play priors for model based reconstruction". In: *2013 IEEE Global Conference on Signal and Information Processing*. IEEE. 2013, pp. 945–948.

[131] Aladin Virmaux and Kevin Scaman. "Lipschitz regularity of deep neural networks: analysis and efficient estimation". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/file/d54e99a6c03704e95e6965532Paper.pdf.

[132] Eugene Vorontsov et al. “On orthogonality and learning recurrent networks with long term dependencies”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 3570–3578.

[133] Xiaolong Wang et al. “Non-Local Neural Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[134] Xintao Wang et al. “EDVR: Video Restoration With Enhanced Deformable Convolutional Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019.

[135] Xintao Wang et al. “Recovering Realistic Texture in Image Super-Resolution by Deep Spatial Feature Transform”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[136] Z. Wang et al. “Multi-Memory Convolutional Neural Network for Video Super-Resolution”. In: *IEEE Transactions on Image Processing* 28.5 (2019), pp. 2530–2544.

[137] Zhendong Wang et al. “Uformer: A General U-Shaped Transformer for Image Restoration”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 17683–17693.

[138] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.

[139] Andreas Wedel et al. “An Improved Algorithm for TV-L 1 Optical Flow”. In: *Statistical and Geometrical Approaches to Visual Motion Analysis*. 2008.

[140] Stephen J Wernecke. “Two-dimensional maximum entropy reconstruction of radio brightness”. In: *Radio Science* 12.5 (1977), pp. 831–844.

[141] Yves Wiaux et al. “Compressed sensing imaging techniques for radio interferometry”. In: *Monthly Notices of the Royal Astronomical Society* 395.3 (2009), pp. 1733–1742.

[142] Norbert Wiener et al. *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*. Vol. 113. 21. MIT press Cambridge, MA, 1949.

[143] Thomas L Wilson, Kristen Rohlfs, and Susanne Hüttemeister. *Tools of radio astronomy*. Vol. 5. Springer, 2009.

[144] Scott Wisdom et al. “Full-Capacity Unitary Recurrent Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/file/d9ff90f4000eacd3a6c9cb27f78994cf-Paper.pdf.

[145] Li Xu et al. “Deep convolutional neural network for image deconvolution”. In: *Advances in neural information processing systems*. 2014, pp. 1790–1798.

[146] Tianfan Xue et al. “Video enhancement with task-oriented flow”. In: *International Journal of Computer Vision* 127.8 (2019), pp. 1106–1125.

[147] Jianchao Yang et al. “Image super-resolution as sparse representation of raw image patches”. In: *2008 IEEE conference on computer vision and pattern recognition*. IEEE. 2008, pp. 1–8.

[148] yan yang et al. “Deep ADMM-Net for Compressive Sensing MRI”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/file/1679091c5a880faf6fb5e6087eb1b2dc-Paper.pdf.

[149] Peng Yi et al. “Omniscient Video Super-Resolution”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 4429–4438.

[150] Peng Yi et al. “Progressive Fusion Video Super-Resolution Network via Exploiting Non-Local Spatio-Temporal Correlations”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[151] Jason J Yu, Adam W Harley, and Konstantinos G Derpanis. “Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 3–10.

[152] Christopher Zach, Thomas Pock, and Horst Bischof. “A duality based approach for realtime TV-L 1 optical flow”. In: *Joint pattern recognition symposium*. Springer. 2007, pp. 214–223.

[153] Syed Waqas Zamir et al. “Restormer: Efficient Transformer for High-Resolution Image Restoration”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 5728–5739.

[154] Roman Zeyde, Michael Elad, and Matan Protter. “On Single Image Scale-Up Using Sparse-Representations”. In: *Curves and Surfaces*. Ed. by Jean-Daniel Boissonnat et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 711–730. ISBN: 978-3-642-27413-8.

[155] Jiong Zhang, Qi Lei, and Inderjit Dhillon. “Stabilizing gradients for deep neural networks via efficient svd parameterization”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 5806–5814.

[156] Kai Zhang, Luc Van Gool, and Radu Timofte. “Deep Unfolding Network for Image Super-Resolution”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[157] Kai Zhang, Wangmeng Zuo, and Lei Zhang. “Learning a Single Convolutional Super-Resolution Network for Multiple Degradations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[158] Kai Zhang et al. “Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising”. In: *IEEE Transactions on Image Processing* 26.7 (2017), pp. 3142–3155.

[159] Kai Zhang et al. “Learning Deep CNN Denoiser Prior for Image Restoration”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[160] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[161] Yulun Zhang et al. “Image super-resolution using very deep residual channel attention networks”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 286–301.

[162] Hang Zhao et al. “Loss Functions for Image Restoration With Neural Networks”. In: *IEEE Transactions on Computational Imaging* 3.1 (2017), pp. 47–57. DOI: 10.1109/TCI.2016.2644865.

[163] Yue Zhao, Yuanjun Xiong, and Dahua Lin. “Trajectory convolution for action recognition”. In: *Advances in neural information processing systems* 31 (2018).

[164] Ding-Xuan Zhou. “Universality of deep convolutional neural networks”. In: *Applied and computational harmonic analysis* 48.2 (2020), pp. 787–794.

[165] Daniel Zoran and Yair Weiss. “From learning models of natural image patches to whole image restoration”. In: *2011 International Conference on Computer Vision*. 2011, pp. 479–486. DOI: 10.1109/ICCV.2011.6126278.