Hybrid Non-blind Image Deblurring for Real Scenarios

Thomas Eboli

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure

Hybrid Non-Blind Image Deblurring for Real Scenarios

Soutenue par

Thomas Eboli

Le 21 septembre 2021

École doctorale n°386

Sciences Mathématiques de Paris Centre

Spécialité

Informatique

Composition du jury :

Julien Mairal INRIA	*Rapporteur*
Jean-Michel Morel École Normale Supérieure de Paris-Saclay	*Rapporteur*
Julie Delon MAP5, Université de Paris	*Présidente du jury*
Patrick Pérez Valeo AI	*Examinateur*
Daniel Cremers Technical University of Munich	*Examinateur*
Jean Ponce INRIA	*Directeur de thèse*
Jian Sun Xi'an Jiaotong University	*Directeur de thèse*

ENS | PSL

Résumé

La netteté est un critère important lorsque l'on souhaite prendre de bonnes photographies. Plusieurs facteurs tels que les paramètres de l'appareil photo, mouvement et defocus peuvent réduire la netteté d'une photographie et causer du flou qui détruit des details de l'image. Ce contenu peut être retrouvé par le défloutage, un problème inverse mal posé qui utilise la photographie floue, et le flou si connu, pour estimer une version nette.

Les techniques de traitement d'image habituelles reposent sur l'optimisation utilisant des priors empiriques sur les images ou des approches d'apprentissage automatique exploitant des paires d'images nettes et floue en guise de supervision. Dans cette thèse, nous suivons une tendance récente visant à combiner les deux types de techniques présentés précédemment et produisant l'état de l'art pour le défloutage. De telles méthodes hybrides exploitent un modèle de formation de flou et une solution basée sur un algorithme d'optimisation, amélioré avec une approche d'apprentissage automatique. Nous abordons de tels modèles pour le défloutage non-aveugle d'images où l'opérateur de flou est supposé connu, ce qui est réaliste dans de rélles situations.

Nous présentons d'abord une fonction paramétriques pour déflouter des images RVB, en incorporant des itérations de point-fixe de Richardson préconditionnées pour remplacer la classique transformée de Fourier rapide (TFR), sujette aux artefacts d'ondulation, en particulier aux bords d'une image. Nous exploitons le modèle de formation de flou implémenté par la convolution avec un filtre linéaire pour efficacement calculer le préconditioneur avec la TFR, résultant en une approche pour le défloutage aveugle rapide et performante. Nous comparons ce modèle à d'autres méthodes hybride de l'état de l'art pour le défloutage non-aveugle sur des images

synthétiques et obtenons les meilleurs résultats pour retirer des floues uniformes et non-uniformes, ainsi que sur de vraies images floues. Dans une seconde contribution, nous proposons un modèle pour le dématriçage et le défloutage non-aveugle simultanés pour les images “raw”. La restauration est à nouveau effectuée avec une méthode hybride évaluée sur des images synthétiques ainsi que de vraies images “raw” dégradées par des aberrations optiques.

Abstract

Sharpness is an important criterion for shooting acceptable photographs. Several factors such as the camera settings, motion or defocus may decrease image sharpness and lead to blur, resulting in the loss of details. Recovering this lost content can be casted as a deblurring problem, *i.e.,* an ill-posed inverse problem that uses the blurry photograph, and the possibly known blur, to estimate a sharp one.

Typical image processing techniques rely either on optimization algorithms using a handcrafted image prior or machine learning approaches leveraging supervisory pairs of sharp and synthetic blurry images. In this thesis, we follow a recent trend that combines the two sorts of techniques previously detailed and achieving the state-of-the-art image deblurring results. Such hybrid schemes exploit a blurry image formation model and a typical model-based solver upgraded with a learning-based method. We explore such models in the context of non-blind deblurring where the blurring operator is supposed known, which occurs in real-world scenarios.

We first present a parametric function for RGB image deblurring, embedding preconditioned Richardson fixed-point iterations to replace the classical fast Fourier transform (FFT) algorithm prone to ringing artifacts, especially at the image boundaries. We exploit the blur forward model implemented with the convolution with a linear filter to efficiently compute the preconditioner with FFT, yielding a fast and accurate non-blind deblurring approach. We compare the proposed model to other state-of-the-art hybrid non-blind deburring techniques on synthetic images and achieve the best results for both uniform and non-uniform blurs and on real blurry images as well. In a second contribution, we propose a model for joint non-blind deblurring and demosaicking of raw images. Restoration is again carried out with a

hybrid method evaluated on synthetic images but also real raw images degraded with optical aberrations.

Contents

Chapter 1

Introduction

1.1 Goal

In 2020, there were 3.5 billion smartphone users around the world and over 1.5 billion smartphone units were sold in 2019. Most of these phones are now equipped with at least one digital camera. This explains in part why 50 million pictures were uploaded on Instagram per day in 2017, which in turn indicates how important digital photography now is in everybody's life. One can shoot many images of a scene with a digital camera and retain only the few best ones, for instance those not exhibiting motion blur. The selection is thus based on criteria such as image sharpness, which also plays a key role in scientific fields such as astronomy or microscopy. We are concerned in turn with images, and focus on post-processing approaches seeking to correct their imperfections. Even though a simple solution to avoid motion blur is to shoot photographs with a sturdy tripod, all images are actually blurred to a certain extent by the optics of the camera itself, resulting in an inevitable loss of sharpness compared to what a photograph taken with a "perfect" lens would look like.

Concretely, we address the problem of non-blind deblurring of digital photographs. In this context, the blur is assumed to be known, which can be used as additional information to solve an inverse problem. This is an important setting in many fields in practice, perfectly illustrated by the Hubble space telescope (HST) sent to space in 1990. Only a few days after starting the mission, the NASA scientists noticed blur in

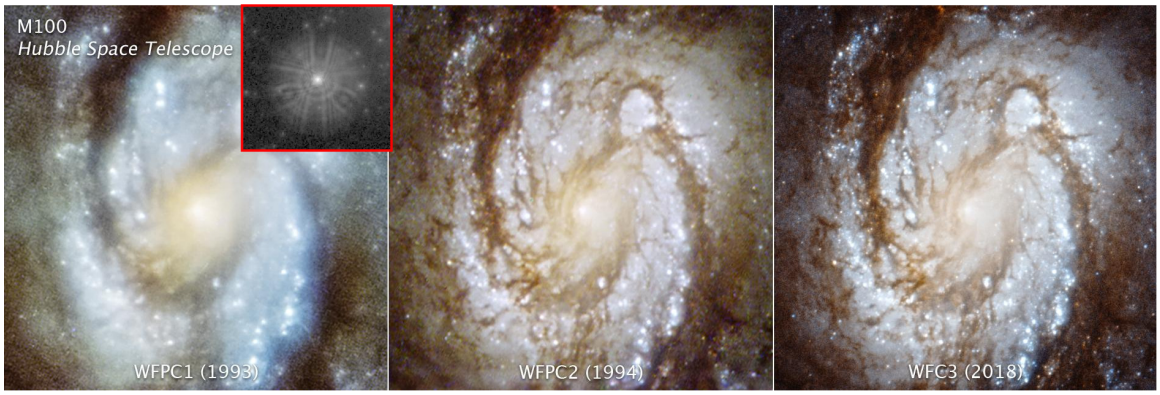


Figure 1-1: Images of the M100 galaxy by the Hubble space telescope throughout the years. From left to right: an image of M100 with the initial mirror used between 1990 and 1993 and an estimate of the blur by pointing a star in the red window, an image of M100 in 1994 after the first maintenance and in 2018 after several other corrections. The comparison between the image on the left and on the right gives an idea of the impact of the blur.

the recorded images caused by spherical aberrations linked to manufacturing errors. Figure 1-1 shows an image of the M100 galaxy taken with the HST. A NASA mission eventually fixed the mirror one year later but improved images had been obtained in the meantime by first estimating the intrinsic blur of the mirror from star images, and second improving the image sharpness with a non-blind deblurring method.

We address in this thesis diverse types of blur that may be due to motion and optical aberrations for example. Two main approaches are widely used to predict a sharp photograph. On the one hand, classical image processing algorithms use the known blur kernel and solve an inverse problem: a penalized energy embeds a blur image formation model in a fitting term and an image prior function constrains the set of solutions [6]. The penalty function design plays an important role in the ability of these approaches to reconstruct missing details but the more elaborated ones result in highly non-convex problems, hard and/or slow to solve. On the other hand, recent advances in computer vision based on supervised machine learning techniques, necessitate a large amount of annotated data, hard to collect from real photographs, but can be used for deblurring images. We will see in this thesis that the best of the two worlds can be obtained with hybrid methods, achieving state-of-the-art deblurring results.

Blur can be seen as a low-pass filtering process, setting to zero most of the high frequencies in an image's Fourier spectrum, and deblurring thus amounts to estimating this missing content. Image restoration techniques may either reconstruct the original missing content, or hallucinate the missing details to build a visually pleasing sharp photograph. Indeed, the two choice between these two options depends on the application context and the severity of blur. In astronomy or medical imagery, scientists cannot afford to hallucinate details in the image of a planet or a cell since it would alter their conclusions. In these cases, blur is often only due to the optics. It is thus generally well-understood and can be modelled with a small support kernel. Equivalently, the blur only alters a small range of the high frequencies in an image, suggesting that missing details can hopefully be restored by inverting a physical forward model leading to the blurry images. In contrast, personal photography often deals with out-of-focus and/or motion blur, which may deteriorate larger parts of an image's Fourier spectrum. In this case, exactly reconstructing the missing details is very hard because of the kind of blur but is not mandatory since most of images we take must actually look like natural photographs, not necessarily faithful to the real observed scene. Deblurring by hallucinating the missing details to build a visually pleasing image is thus a relevant alternative to the solution of an inverse problem, and is often carried out with learning-based approaches.

We thus propose in this thesis to design methods taking the best of the inverse problem world and the machine learning techniques, to handle blur alone in the case of motion blur but also jointly inverting blur and mosaicking to get rid of optical aberrations. These models are trained with synthetic but reasonable training data comprising blur but also mosaicking and saturation. The more realistic the data and the better the image formation model, the more accurate the restored images are.

1.2 Motivation

The work presented in this thesis is motivated by several observations.

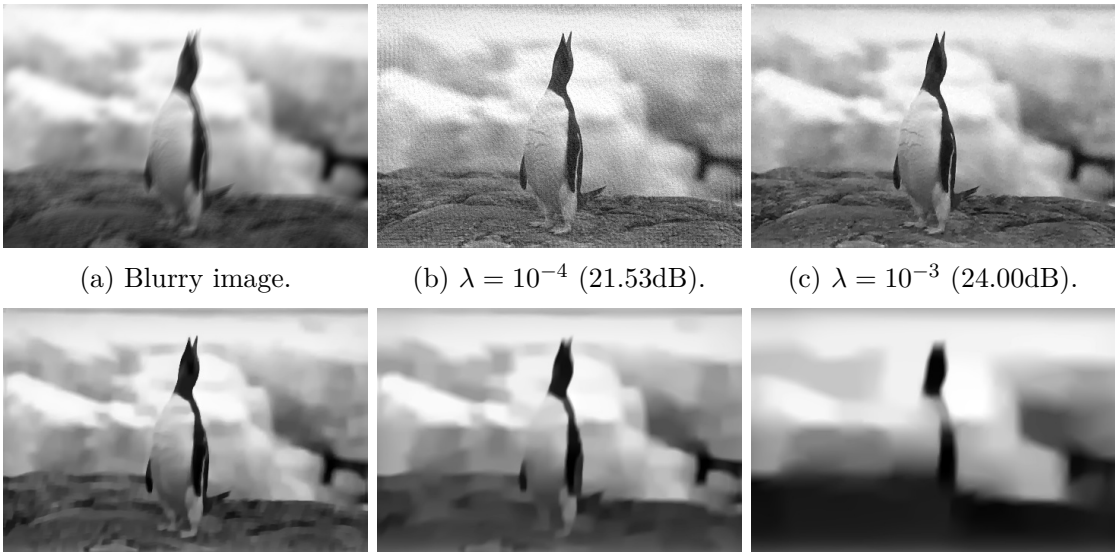
As discussed above, image deblurring can be addressed with learning-based meth-

ods in a regression setting with supervision comparing a predicted deblurred image with a sharp target thanks to a pixelwise metric such as the ℓ_2 distance. In addition to the training loss and a learnable deblurring technique, typically using a convolutional neural network (CNN) [108], one has to collect a (large) dataset of corresponding blurry and sharp images.

Recording real pairs of corresponding blurry and sharp images is very hard in practice. It can be done with two different cameras mounted on the same rig, shooting at once a sharp image and the corresponding blurry one. It can be also achieved with a single camera, but in this case, the two images taken at different moments must have the same illumination and be perfectly aligned. Both options are thus very hard to achieve in practice and synthetic data are often preferred as a result.

Fortunately, the physical and digital pipelines to transform an analog image into a digital blurry and noisy one has been the topic of numerous works in the optics, signal processing, 3D graphics and computer vision communities. If we had analog photographs, we could easily generate blurs and transform sharp images into almost-realistic blurry ones, but we only have access to digital images instead, quantized on a finite number of bits. The analog image is thus generally replaced by a digital RGB image which undergoes an approximate forward blur formation model. Previous works in the context of image denoising [2] and image upsampling [139] have shown that synthesizing training data is obviously not as interesting as using pairs of corresponding sharp and real blurry images for training restoration models. Yet, synthetic images with a reasonable forward model can approximate well real blurry photographs [32] and a large corpus of them may be enough for generalizing to real-world scenarios.

The image formation model is typically embedded in the data-fitting term of a penalized energy function, which also includes an image prior encouraging the solution to exhibit natural image features, such as the total variation (TV) prior [106] imposing the solution to have sharp edges, at the cost of smoothing textures. The balancing weight between the data-fitting term and the prior can have a dramatic impact on the solution. Figure 1-2 shows one deblurring example obtained after solving such



(a) Blurry image. (b) $\lambda = 10^{-4}$ (21.53dB). (c) $\lambda = 10^{-3}$ (24.00dB).

(d) $\lambda = 10^{-2}$ (26.40dB). (e) $\lambda = 10^{-1}$ (27.01dB). (f) $\lambda = 1$ (24.06dB).

Figure 1-2: Impact of setting the regularization parameter λ in a non-blind deblurring algorithm with the TV$-\ell_1$ prior. A small value for λ magnifies noise, but a large value over-smooth the textures. A trade-off should be met, which is hard to set. The PSNR score (in parenthesis) does not necessarily corresponds to the visual accuracy.

composed energy with a prior on the solution TV. The algorithm is called HQS-TVℓ_1 and is further detailed in Chapter 4. In this figure, we vary λ between 10^{-4} and 1, a range of relevant values, and show that visually pleasing images for this example are obtained for λ between 10^{-3} and 10^{-2} although the maximum PSNR is reached for 10^{-1}.

This quantitative metric is a classical criterion for cross-validating λ [88], even though it does not correspond to the best qualitative deblurring result. Efficient tuning of this weight value can in turn be learnt, motivating hybrid methods relying on both model-based techniques leveraging an image forward model and data-driven approaches getting rid of handcrafted criteria akin to the PSNR.

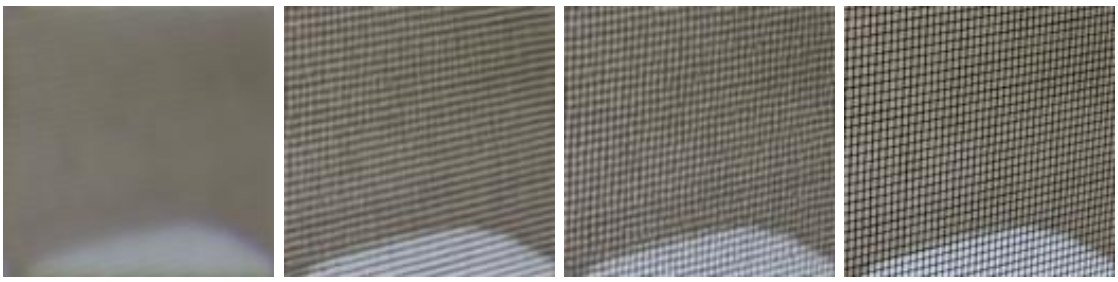
An optimization-based approach to solving the penalized energy mentioned above generally features a linear system with the blur matrix being circulant [46]. Since images can be very large, one must rely on large-scale linear system solvers such as conjugate gradient (CG). The fast Fourier transform (FFT) leverages instead the circulant structure of the linear operator, leading to a very efficient closed-form solution

(a) Blurry. (b) Approach with FFT [137]. (c) Chapter 4 approach.

Figure 1-3: The teddy bear image is blurry because of camera motion during exposure. The blur has been estimated beforehand with a blur estimation method and is shown in the red box (it is zoomed by a factor 3). The deconvolution approach based on FFT introduces artifacts around the badge on the hat and next to the buttons and collar. The proposed approach in Chapter 4 also removes the blur without producing these artifacts. The full image is shown in Chapter 4 addressing motion blur.

that may contain ringing artifacts. This is particularly true when the blur is only approximated. Figure 1-3 shows an example with an image we have taken ourselves with a handhelded camera. We have estimated the blur from the original image with the motion blur estimation technique of [95] and deblurred the image with both a state-of-the-art non-blind deblurring method using FFT [137] and the method proposed in Chapter 4. The figure shows that the baseline method introduces noticeable ringing artifacts next to salient edges, unlike ours. We discuss in Chapter 4 the performance of existing least-square solvers and propose convolutional fixed-point iterations for fast and efficient deconvolution.

Optical aberrations removal requires handling color-specific blurs as we will see in Chapter 2. A deblurring algorithm should typically be applied to an image first processed by a demosaicked algorithm. Schuler *et al.* [109] explain that an optimal chromatic aberration removal algorithm should jointly predict the missing color components in the raw image and the aberrations. We show in Figure 1-4 an example comparing a two-stage approach that first demosaicks a blurry raw image and second deblurs it, and the joint approach proposed in Chapter 5. In this figure, the two-stage approach fails to reconstruct fine details of the grid such as the vertical structures



(a) Blurry. (b) Two-stage. (c) Joint approach. (d) Ground-truth.

Figure 1-4: An image of a grid is synthetically blurred and mosaicked with a forward model detailed in Chapter 5 and a real Nikon lens optical aberration measured in [10]. The intermediate blurry image is shown on the left but the input is indeed a raw blurry image. From left to right, the remaining images shown are obtained by demosaicking then deblurring the input image in a two-step scheme, the image jointly demosaicked and deblurred with our approach in Chapter 5 and the ground-truth image exhibiting high-frequency details prone to interpolation artifacts during demosaicking from [45]. The two-stage approach cannot reconstruct finer details unlike our approach directly operating on the raw and blurry image.

whereas ours can. We discuss the reasons of this difference and the details of the proposed model in Chapter 5.

Based on the motivations described in this section, we want to design fast and efficient parametric functions for non-blind deblurring to get rid of motion blurs in colored photographs and for joint non-blind deblurring and demosaicking to remove optical aberrations from raw images. The proposed modules leverage physics-grounded blur image formation models and learning-based techniques from large corpus of synthetic, yet reasonable, training data. We will describe the main challenges related to our goal in the following section.

1.3 Challenges

The first challenge addressed in this thesis is addressing non-blind image deblurring with a hybrid method that leverages supervisory data and a blurry image formation model. Most of the state-of-the-art techniques alternate between evaluating a handcrafted or learnt proximal operator and solving a least-squares sub-problems whose linear operator implements a convolution with a linear filter. The fast Fourier trans-

form (FFT) algorithm efficiently computes the closed-form solution but is notorious to inject ringing artifacts in the solution. Conjugate gradient (CG) iteratively finds the solution of the least-squares problem but might be slow because of the conditioning of deblurring problems, limiting the possibility to embed such iterations in a learning-based structure.

The second challenge we tackle is joint image non-blind deblurring and demosaicking, for which no exact and efficient solver similar to FFT in the context of non-blind deblurring, exists according to Palyi *et al.* [92], thus preventing the use of typical non-blind deblurring hybrid techniques for this problem. Schuler *et al.*'s solution [109] is an alternate optimization scheme based on CG taking several hours to process a 12 megapixel image or an approximate faster solution deblurring with FFT a demosaicked image, empirically proved to achieve sub-optimal compared to the slow option. A practical application of such algorithms is optical aberration removal, a degradation limiting the image sharpness and observed in *any* photograph taken with a camera, even on a sturdy tripod, and equipped with a lens whose inevitable manufacturing defaults cause the blur.

1.4 Outline

This thesis is organized into five chapters besides the introduction. Chapter 2 reviews the in-camera image formation model, from a continuous focal image to a digital sRGB one. It focuses in particular on modelling intrinsic and extrinsic blurs. This analysis is used in Chapter 3 to review the existing techniques for blur estimating and non-blind deblurring.

Chapter 4 introduces a new hybrid non-blind RGB image deblurring approach. It is based on an efficient and fast preconditioned Richardson fixed-point iterations [64] to solve a least-squares problem whose linear operator is a circulant matrix representing the convolution with a linear filter. We show that the proposed method outperforms the fast Fourier transform (FFT) and the conjugate gradient (CG) descent algorithm typically used for this problem. We embed this iterative scheme

into a parametric function, trained with synthetic triplets featuring random motion blurs. It achieves state-of-the-art results for non-blind RGB image deblurring on both synthetic and real images. This work was presented at ECCV 2020.

Chapter 5 focuses on removing the blur caused by the camera's optics. Motivated by the observations of Schuler *et al.* [109], we solve this problem with a joint non-blind deblurring and demosaicking hybrid approach. In particular, we invert the joint blur and mosaick operator with a FFT-based solver previously introduced for image upsampling by Zhang *et al.* [137]. We compare this joint approach to two-stage methods that first demosaick a blurry raw image and second deblur it, each stage performed with a state-of-the-art algorithm. We show with experiments on both synthetic and real images that jointly handling deblurring and demosaicking leads to superior results over considering each step independently. The proposed approach efficiently removes the optical aberrations caused by the optics of high-end cameras and lenses from real raw photographs. This work is in preparation for submission to IEEE Transaction on Image Processing.

Chapter 6 finally concludes this thesis and opens possibilities for future work.

Chapter 2

Digital Image Formation Model

This chapter covers three main stages in a camera to convert the analog image of a scene into a digital photograph. First, we focus on the camera optics and shooting conditions, causing blur in the recorded photographs. Second, we discuss the sensor converting the analog image into a digital array. It makes use in particular of a colored filter array (or *CFA*) for recording colored images. We finally present the image signal processing (or *ISP*) pipeline that turns the raw image into an RGB image displayed on a screen. Figure 2-1 shows a diagram of the pipeline considered in this work.

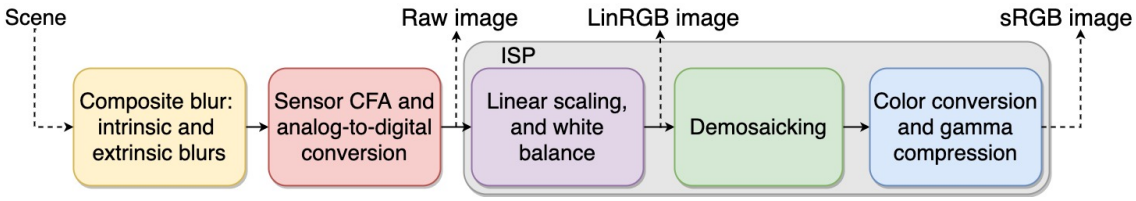


Figure 2-1: The digital camera pipeline presented in this chapter.

The yellow and red blocks represent the two first stages previously mentioned. The three remaining blocks implement the main submodules of the ISP pipeline.

2.1 Camera optics and shooting conditions

Figure 2-1 shows that the incident scene radiance first gets through the lens of the camera. Combined with the shooting conditions, *e.g.,* whether the camera is stable

or an object moves during exposure, the radiance results in a blurry analog image. We categorize in this thesis two types of blur: *Intrinsic* blur caused by the camera optics, which is systematic to each image taken with the same camera/lens pair, and *extrinsic* blur, unique to each photograph.

2.1.1 Intrinsic blur

In most cameras, a lens directs the light rays so they focus on the sensor plane. The Gauss conditions in geometrical optics [103] assume that a point in the focal image exactly corresponds to a point in the recorded photograph. In practice, the shape of a lens and manufacturing errors violate these assumptions. These imperfections, named optical aberrations, combined with diffraction due to the finite aperture and additional blur caused by the camera's antialiasing filter lead to a scene-independent blur in any image.

Diffraction

The quantity of light entering the lens of a camera is controlled by a diaphragm. The aperture of the lens is traditionally measured with the f-number denoted by F: it is the ratio of the lens focal length f and the aperture diameter D

$$F = \frac{f}{D} \tag{2.1}$$

For instance, a f-number of 5 with a focal length f of 20mm means the aperture has a diameter of 4mm. It is given in the format f/F, *e.g.,* $f/1$, $f/2.8$, $f/5.6$.

Light, as any sort of wave, is diffracted when entering finite-sized apertures. In the case of a circular diaphragm, like for DSLRs, a 2D point becomes a spot called the Airy figure [3].

Provided the f-number and the focal length (in the image EXIF metadata for instance), we can compute the aperture diameter D and thus the diameter of the

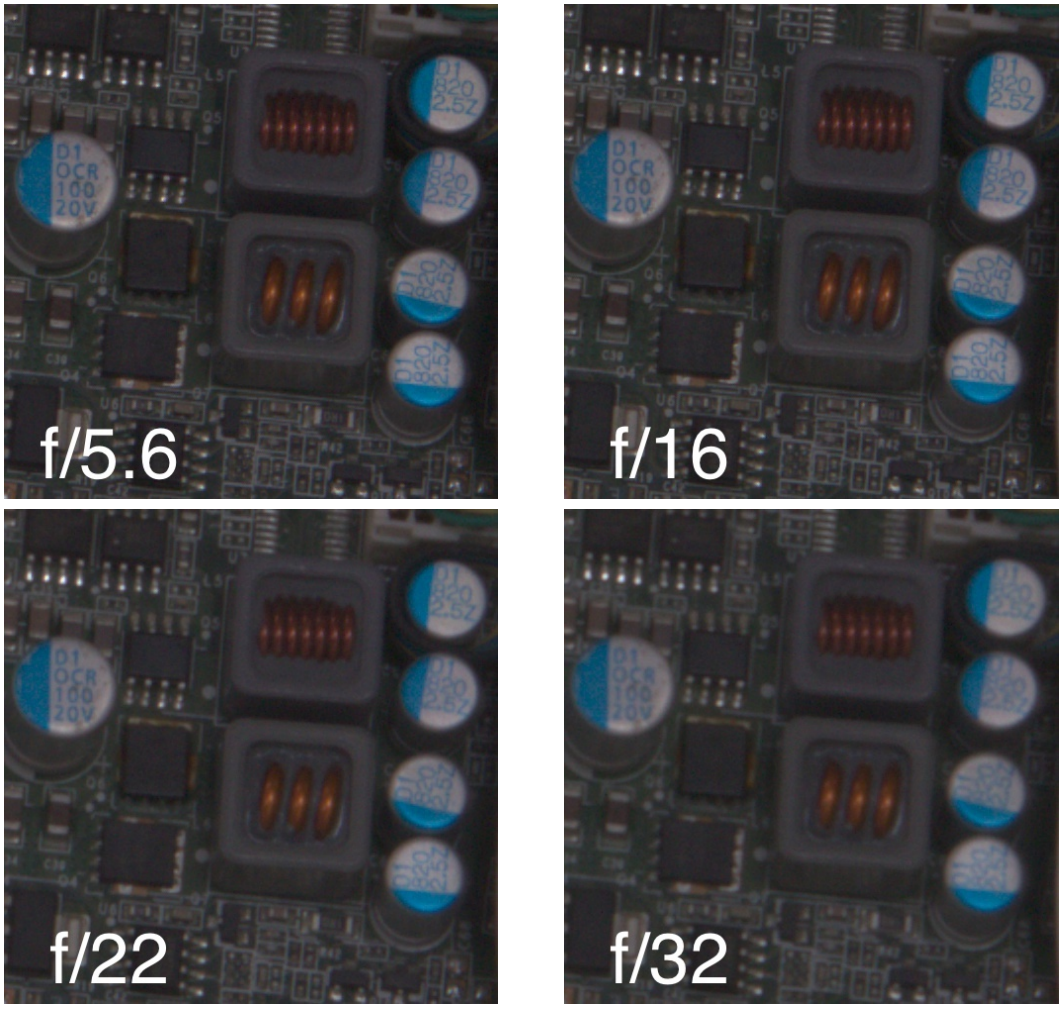


Figure 2-2: Impact of diffraction for different apertures. We used a Canon EOS 550D camera with a Canon EF-S 18-135mm lens and focal length set to 85mm on a tripod. We display the raw images demosaicked with bilinear interpolation. As the aperture narrows (the f-number goes up), less and less detail can be observed on the chip because diffraction reduces the resolution of the image. Blur becomes noticeable for apertures below $f/22$ (less than 4mm of diameter).

Airy figure main lobe with the formula

$$\sin\theta \approx 1.22\frac{\lambda}{D}, \tag{2.2}$$

where θ is the angle at which the first zero occurs and λ is the light ray's wavelength. The angle θ related to the ratio λ/D gives the angular resolution at the diffraction limit. Diffraction becomes really problematic in practice when the f-number grows, or equivalently when the aperture D decreases.

The impact of diffraction greatly depends on the size of the sensor photosite. For instance the size of a photosite for the Canon EOS 550D camera with a sensor of size 22.3×14.9mm is $4.29 \times 4.29\mu$m[1].

For a recent flagship smartphone such as the 5T model from OnePlus with a sensor of size 5.22×3.92mm, the photosite size is $1.13 \times 1.13\mu$m^2. Smartphone cameras are thus more sensible to diffraction than DSLRs due to the more compact size of the sensor.

A quick computation based on Eqs. (2.1) and (2.2) shows that for the Canon 550D reflex camera with lens at focal length $f = 50$mm and aperture set to $f/22$, if the distance between the aperture and the sensor is 2cm (a reasonable assumption), the radius of the Airy disk for red rays with $\lambda = 400$nm is about 9μm and for blue rays with $\lambda = 780$nm is about 17μm. This shows that (i) for very small apertures, the diffraction spot can cover a several photosites, impacting the image sharpness, and (ii) diffraction blur has different different behaviors depending on the color channel. Indeed, in the previous computation, the red spot covers about 2×2 photosites where the blue spot covers 4×4 photosites. For a f-number of $f/5.6$, the corresponding radii are respectively 2.27 μm and 4.27 μm for the red and blue rays considered previously. For this wider aperture and at this focal distance, diffraction blur is negligible.

Equation (2.2) is valid for in-focus points, and diffraction is even more problematic for out-of-focus ones [99] but their study is not the topic of this thesis.

[1]For the Canon EOS550D DSLR, a sensor image has size 5196×3464 pixels which leads to $0.0223/5196 \approx 4.29\mu$m and $0.0149/3464 \approx 4.29\mu$m.

[2]For the OnePlus 5T phone rear camera, a sensor image has size 4608×3456 pixels which leads to $0.0052/4608 \approx 1.13\mu$m and $0.0039/3456 \approx 1.13\mu$m.

Optical aberrations

The lenses used in DSLRs and smartphones to focus the light rays on the camera sensor are typically made of glass and are almost never flat; they are never perfect optical devices and thus modify the trajectory of incident light. A ray propagated in the air and eventually reaching the curved surface of the lens is refracted according to the Snell-Descartes formula:

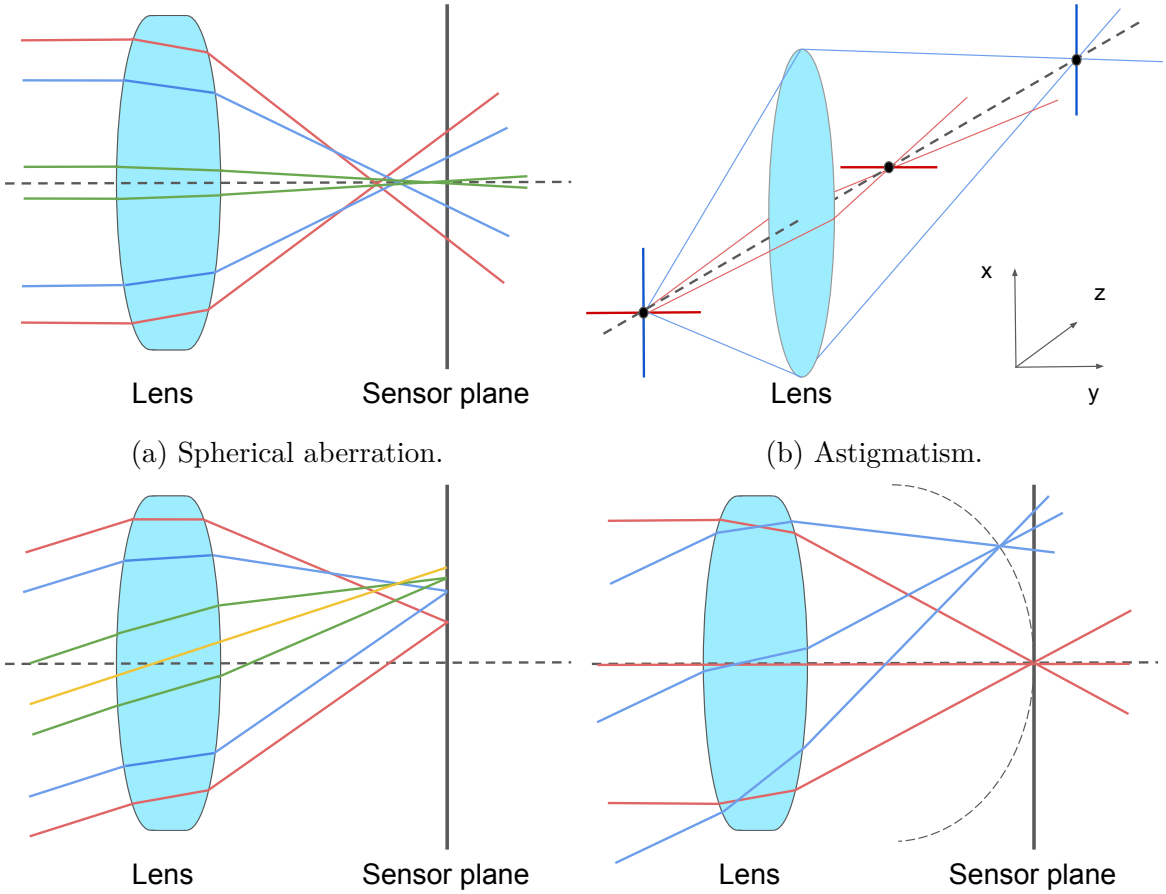
$$n_1(\lambda)\sin(\theta_1) = n_2(\lambda)\sin(\theta_2). \tag{2.3}$$

This formula gives a model of the rays' deflection at the interface of two materials. It is commonly assumed that the refraction index of the air is almost 1 for all the wavelengths in the visible spectrum whereas in glass, for the visible spectrum whose limits are about between 380nm (violet) and 700nm (red), it ranges between 1.53 and 1.51, at 15°C and a pressure of 101325Pa [28]. Careful manufacturing of a lens [54] made of various converging and diverging lenses can correct this physical phemonenon but at the cost of an optical system much more complex to analyse and *optical aberrations* [103, 109, 110, 135].

Since refraction is wavelength dependent, one should separate the behaviour of a lens between a monochromatic and a polychromatic light ray.

Several lens benchmarks characterize the optical aberrations by solely measuring the modulation transform function (MTF) [10, 35] that gives information on the aberrations along two orthogonal directions.

Monochromatic aberrations. On this paragraph, we restrict our analysis a monochromatic light ray, *i.e.,* a ray comprising a single wavelength λ, and only discuss the five sorts of primary monochromatic aberrations detailed by the Seidel theory [103, 122]: *spherical* aberrations, *coma*, *astigmatism*, *field curvature* and *field distortion*. Figure 2-3 shows the four first monochromatic aberrations previously mentioned for a *thick* lens [103]. Field distortion cannot be as easily represented as the four other monochromatic aberrations with geometrical optics and is thus not shown it this



(a) Spherical aberration.

(b) Astigmatism.

(c) Coma.

(d) Field curvature.

Figure 2-3: Illustration of four of the monochromatic aberrations of a lens. The caused of each individual aberration is discussed in the text for details and explanations of these phenomena. Spherical aberration in (a) bends the rays parallel to the optical axis and make them focus closer to the lens, creating confusion circles toward the center of the sensor plane. Astigmatism in (b) makes two lines in two different planes to focus at different depths, creating a confusion circle anywhere on the sensor plane. Coma in (c) translates on the sensor plane the focus point of transverse rays with respect to the optical axis, creating dots with tails (like a comet). Field curvature in (d) makes rays far from the optical axis to focus in a parabolic surface, creating circles of confusion on the sensor plane next to the edges.

figure.

Spherical aberration is a direct consequence of the Snell-Descartes law (2.3) on a curved surface of a lens made of glass. The rays next to the edge of the lens are more bent and thus converge closer to the lens than those next to the optical axis. As a result, light rays do not converge to a single focus but on an interval along the optical axis. This aberration can be corrected by using several converging and diverging lenses to form a compound lens [54].

Coma distorts incident rays coming with a non-zero angle compared to the optical axis. The image of a dot on the sensor plane appears with a trailing tail reminiscent of that of a comet [103].

Astigmatism causes two light rays in perpendicular planes and coming from the same 3D point to not be focused at the same 2D point, but rather at two different images planes (or *foci*) [103]. For instance, a cross has its first line focused on one image plane and its second line focused on a second plane resulting in at least one part of the cross being out-of-focus and thus appears blurry. The sharpest possible image of the cross is when the sensor plane is placed halfway between the two *foci* with the circle of least confusion. This can also happen for on-optical axis rays when the lens is not symmetric due to manufacturing errors - which happens for more than 90% of consumer-grade lenses according to the analysis of [35].

Field curvature maps a flat observed object on a parabolic image surface (or Petzval surface) [103], resulting in an out-of-focus image far from the optical axis. This is due to the curved surface of a real lens that bends off-axis light rays closer to the lens than rays next to the optical axis, resulting in a curved image surface depending only on the geometric characteristics of the lens. A solution to this aberration would be to replace the traditional flat sensor grid by a curved one, like the human eye's retina [7].

Field distortion transforms straight lines of an observed object into curved lines in the image. This aberration is symmetric and can be decomposed into two categories. The first one is the barrel distortion, featured in fish-eye lenses, that maps an image on the surface of a sphere. The second one is the pincushion distortion and can be

thought as the "negative" barrel distortion. Field distortion is noticeable for several lenses, for instance large-range zoom lenses but also prime lenses, and can be corrected with camera calibration approaches [52] or specialized pieces of software.

The four first aberrations result in diverse focusing planes and thus out-of-focus blur depending on pixel locations [122] and can be removed with deblurring methods [109]. Field distortion twists lines in an image and can be corrected with additional camera calibration manipulation [52]. In this list of optical aberration, we have not covered the case of *vignetting* [122] which darkens the corners of an image.

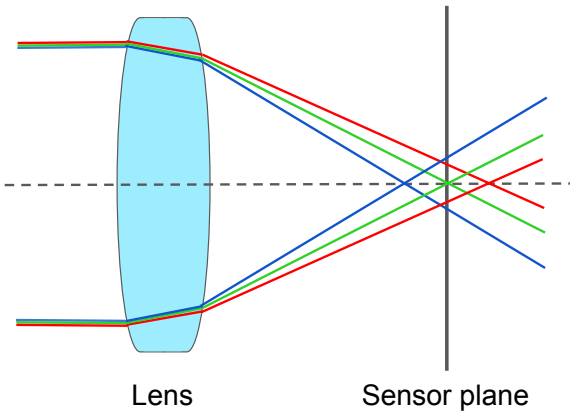
Chromatic aberrations The monochromatic aberrations reviewed above are caused by refraction in a thick lens. The Snell-Descartes law (2.3) shows that refraction is actually wavelength-dependent [62]. The deviated colored rays hit the sensor at different locations yielding colored fringes everywhere in the camera's field of view. These fringes are called *chromatic aberrations*, are caused by *any* lens, and can be categorized into two kinds: *longitudinal* and *lateral* chromatic aberrations [103]. (see Figure 2-4).

Longitudinal chromatic aberrations occur when different color components in a light ray are focused on different focal planes due to the wavelength-dependent Snell-Descartes law (2.3). Such aberrations can appear in the whole image and are typical of long focal lengths [103] (Figure 2-4a).

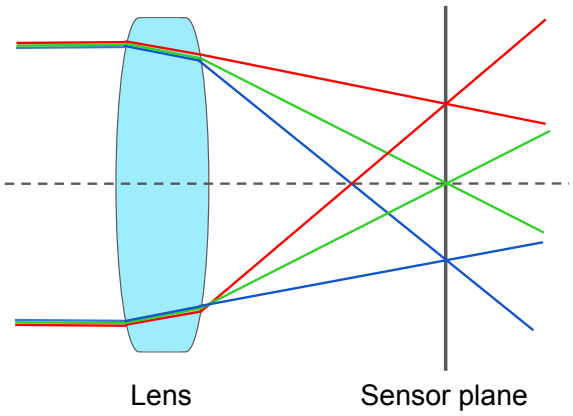
Lateral aberrations happen when different color components in a light ray are focused on the same focal plane but at different locations on its surface. Such an aberration is not visible at the center of an image but rather on the image salient edges and is typical of short focal lengths [103] (Figure 2-4b).

In practice, the chromatic aberrations are a combination of both lateral and longitudinal aberrations and are thus visible in the whole image (Figure 2-4c). In digital photography, *purple fringes* are a direct consequence of chromatic aberrations, and are the result of colored fringes truncated by sensor saturation [19]. Saturation is further discussed in this chapter.

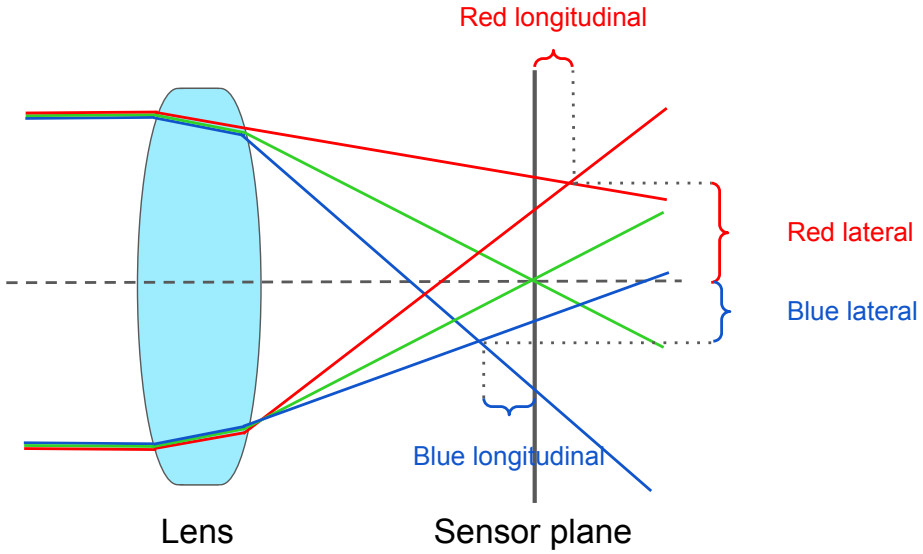
A hardware solution to chromatic aberrations is to use an achromatic lens made



(a) Longitudinal chromatic aberration.



(b) Lateral chromatic aberration.



(c) Combined chromatic aberrations.

Figure 2-4: Illustration of chromatic aberrations. We make the common and reasonable assumption here that the green channel is in focus. The longitudinal aberration in (a) causes the red and blue colors to appear as disks. The longitudinal aberration in (b) causes the red and blue channels to be translated in the sensor plane compared to their green counterpart. In practice, chromatic aberrations are observed as combinations of lateral and longitudinal aberrations, like in (c).

of, at least, two individual lenses with different dispersion properties for compensating the color dependency of the refractive indices of the lens. Another option implemented in recent mobile phones [53] is to shift and rescale the different color channels such that the color fringes next to salient edges are aligned.

Antialiasing

A specificity of digital cameras, that we will detail in the next section, is to convert an analog signal into a discrete image where incident radiance is accumulated and integrated over the sensor surface. In this case, aliasing appears if the space between two pixel centers, or equivalently the spatial resolution of the sensor, is not at least twice the size of the smaller detail in the analog image, as a consequence of the Shanon-Nyquist rule [82]. This can be avoided by adding a low-pass filter in front of the sensor that eliminates the finer details, likely to introducing aliasing, at the cost of additional blur in the sensor image [93, 97].

This filter is sometimes removable in recent DSLRs, and active discussions exist amongst photographers to decide whether fidelity to the scene a the cost of aliasing is acceptable or not.

Camera PSF

Diffraction blur, optical aberrations and antialiasing are compiled in the camera point-spread function (PSF) that explains how a point in the scene becomes a colored spot in the image [33, 61, 63, 99, 109, 110, 135].

The camera PSF is a non-uniform function across the field-of-view (FOV) of the camera [35]. As it varies smoothly, the PSF can be reasonably modelled as a locally-uniform blur [63, 109]. An example of a real camera/lens PSF measured by Bauer *et al.* [10] is shown in Figure 2-5.

We do not cover vignetting [122] that darkens corners of a digital image in this thesis since it is a modulation function of the sensor image and not a blurring operation. It is however handled in the model of Schuler *et al.* [109] and in the optical aberration removal approach proposed in Chapter 5.

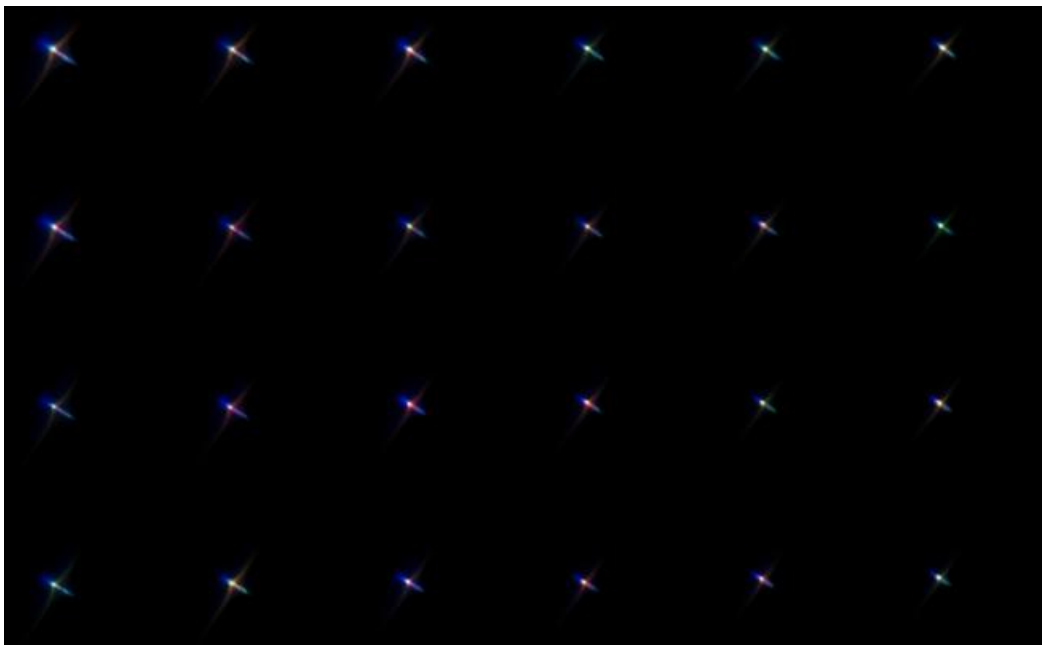


Figure 2-5: Close-up to the top-left corner of the Canon EF 35mm lens' PSF at aperture $f/1.4$ reproduced from Bauer *et al.* [10]. It comprises diffraction (almost none for this large aperture), optical aberrations and in particular coma (its signature are the trails pointing toward the bottom-right corner here) and the camera's antialiasing filter.

2.1.2 Extrinsic blur

Extrinsic blur depends on the scene and shooting conditions. Image restoration should be specifically carried out for each image to be corrected.

Defocus

Defocus corresponds to a blur caused by the convergence of light rays at a point not on the sensor of the camera. If it happens, a point in the 3D scenes will correspond to a spot, also called circle of confusion in this context, whose size depends on the distance between the convergence point and the sensor plane on the optical axis [99]. An image contains defocus blur if at least one object in the observe scene is too close or too far from the camera. This range of acceptable distances is related to the lens's properties and is called the depth of field (DOF) [99].

As detailed above, defocus is a spatially-varying blur. A naive defocus blur model uses a Gaussian filter whose radius depends on the depth [62] but more refined ap-

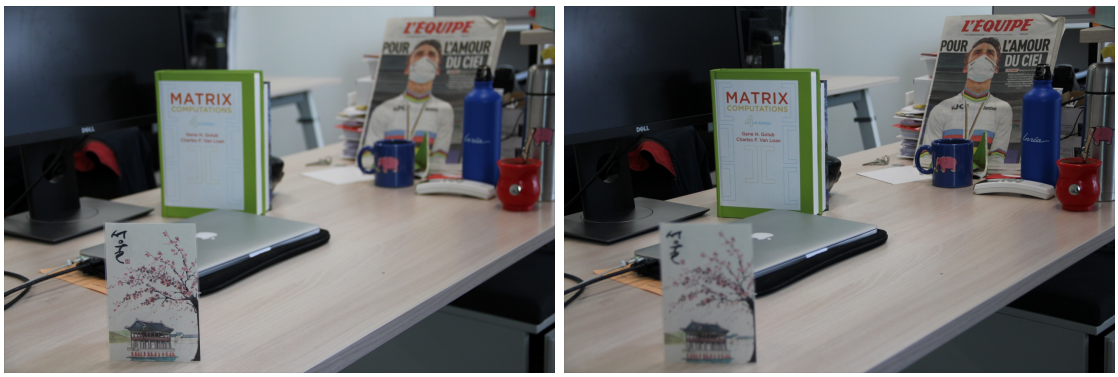


Figure 2-6: Examples of defocus. We used the Canon EOS 550D camera with the Canon EF-S 18-135mm lens and focal length set to 35mm and aperture set to $f/4.5$. The first image is focused on the postcard in the foreground, the book, newspaper and desk in the background being gradually more and more blurry. The second image is focused on the book and newspaper, resulting in the postcard to appear blurry.

proaches model defocus with a handcrafted blur basis [73, 143].

In this thesis, we assume that the images are not subject to defocus blur. This assumption is reasonable for many images even though a more general model would handle this kind of blur.

Motion

Shooting an image is not instantaneous but instead requires a certain exposure time that may vary from a few milliseconds to several seconds depending, for instance, on the amount of light in the scene necessary to record a well lit image.

During this time interval, objects in the scene or the camera itself may move, because of hand tremor [128] or shutter vibrations [100], resulting in motion blur in the image. Even small shifts in the scene can result in important displacements on the image sensor [100].

Motion blur can be represented as the integration of a moving image of the scene on the camera sensor during an exposure time T. Let (u, v) in $\mathbb{R}$ be a point of the sensor plane and (u_t, v_t) the location of this point in the focal image at time t in $[0, T]$. If there is no motion, $(u, v) = (u_t, v_t)$ for all t in $[0, T]$. Let x and y be functions from $\mathbb{R}^2$ to $\mathbb{R}$ representing respectively the image of the scene and the image recorded on

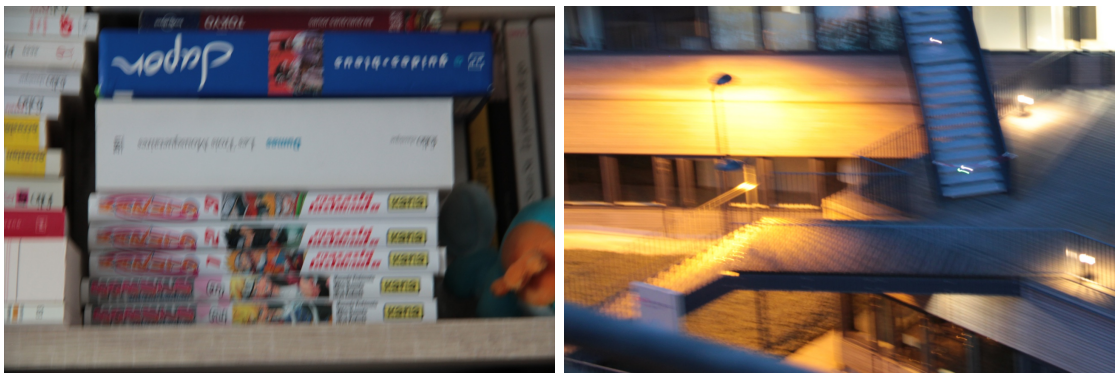


Figure 2-7: Examples of motion blurs. We took these pictures by moving the camera during exposure. In both images, we can see that details such as the books' names or the fence are lost. Texture of the wooden floor is also lost. In the case of saturated images like on the right, we can spot the camera trajectory like next to the lights on the staircase.

the sensor. A blurry pixel at location (u, v) in y is formed following

$$y(u,v) = \frac{1}{T}\int_{t=0}^{T} x(u_t, v_t)\mathrm{d}t. \tag{2.4}$$

Equation (2.4) integrates over time the contribution of neighboring pixels in x around (u, v) and is equivalently replaced by integration over the neighboring pixels in x as

$$y(u,v) = (k * x)(u,v) = \int_{s,t} k_{u,v}(u-s, v-t)x(s,t)\mathrm{d}s\mathrm{d}t, \tag{2.5}$$

where we slightly adapt the notation since in practice (non-uniform) blur k may depend on (u, v) (see Chapter 3 for more details), denoted by the coefficients $k_{u,v}$ in $\mathbb{R}_+$.

2.1.3 Composite blur

The overall blur that corrupts a sharp focal image is the combination of intrinsic and extrinsic blurs, the former being the same for each image taken with a camera, a telescope or a microscope and the latter being scene-dependent.

When taking a picture, a photographer controls several parameters such as the aperture diameter through the f-number, the focal length of a zoom lens, and the

exposure time. For a small aperture (high f-number), shooting a photograph with the same amount of light takes more time since fewer photons hit the sensor compared to a wider aperture. Longer exposure time may lead to more motion blur caused by the camera if it is not attached to a tripod, or objects in the scene such as a car or a person. However, a small aperture masks most of the lens surface except the flat part at the center, thus greatly reducing lateral chromatic aberrations and field curvature caused by the curved surface of the lens next to the edge, finally resulting in a trade-off between the sorts of blur observed in the sensor image.

In the same vein, a larger aperture cannot prevent the intrinsic and extrinsic blurs to corrupt images and would lead to the same kind of trade-off. This suggests that any photograph is blurry, the composite blur being a combination of both the intrinsic and extrinsic blurs whose impacts may be controlled by the settings of the lens (what aperture and focal length), the camera (using or not a tripod), the scene (any moving object or element not in the depth of field, for instance).

2.2 Camera sensor

In this section, we discuss the camera sensor whose role is to convert the incoming radiance of the scene, focused by the lens, into a digital raw image. This analog-to-digital converter can be equipped with a color filter array (or *CFA*) for capturing colored images in most consumer-grade cameras and DSLRs. We also cover the case of saturation in this section, a particular kind of degradation that can diminish the accuracy of many deblurring methods.

2.2.1 Analog-to-digital conversion

The camera sensor is a rectangular chip that converts the analog scene radiance into a digital image. A finite number of photosites covering its surface accumulate the incident photons with either couple-charged device (CCD) [15] or complementary metal–oxide–semiconductor (CMOS) [42] sensors during exposure . The analog-to-digital image conversion generate electrical signals whose magnitude scales linearly

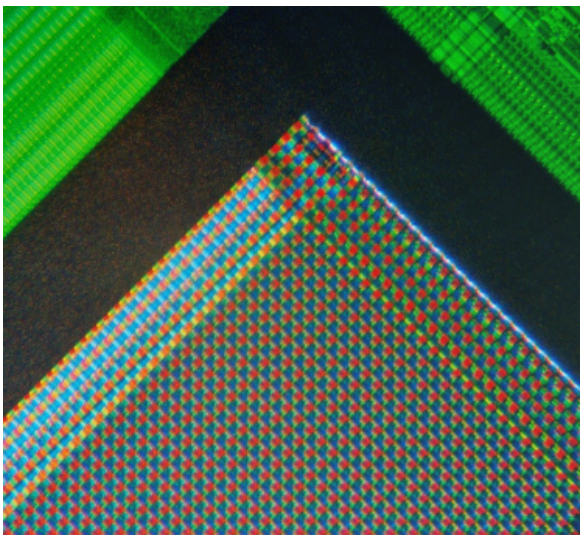


Figure 2-8: A micrograph of the corner of a webcam photosensor array by Natural Philo. The surface of the sensor is covered by squared photosites accumulating the photons during exposure. The mosaic of red, green and blue pixels in front of the photosites form the colored filter array (here the Bayer pattern) whose role is to retain only one color per site, resulting in a mosaicked image. A RGB image of the scene is later interpolated with a demosaicking algorithm.

with the number of collected photons. The electrical signal is finally quantized on a finite number of bits, in general 10 or 12 for recent smartphones and 14 for DSLRs, yielding the digital sensor image. Figure 2-8 shows a webcam sensor made of CCD sensors.

The conversion discussed in the previous paragraph converts the focal image with theorically infinite resolution to a digital one whose resolution is defined by that of the sensor. The analog image is spatially discretized by each photosite that not only temporally collect photons during exposure at a single point on the sensor surface but also spatially integrates the photons reaching a subsurface of the sensor. The spatial resolution of a digital photograph is defined by the number of pixels composing the image, or equivalently by the density of photosites on the sensor.

A forward model connecting the focal image x and the $H \times W \times 3$ sensor image

y_{sensor} can be derived from the formulation of Baker and Kanade [9]:

$$\begin{aligned} y_{\text{sensor}}(p,q) &= Q\left[Kx+\varepsilon\right](p,q), & (2.6)\\ &= Q\left[\int_{(u,v)\in\mathbb{R}^2} k_{p,q}(u,v)x(u,v)\mathrm{d}u\mathrm{d}v+\varepsilon(p,q)\right], & (2.7)\end{aligned}$$

where Q is a quantization function from $\mathbb{R}_+$ to $\{0,1,\ldots,2^N-1\}$ with N the number of bits allocated to each pixel and K models the linear blur operator on the whole image, represented with local weights $k_{p,q}$ in $\mathbb{R}^3$ combining the camera PSF, defocus and motion blurs for generating the blurry pixel $y_{\text{sensor}}(p,q)$. The weights $k_{p,q}$ have non-negative entries and follow the constraint:

$$\int_{(u,v)\in\mathbb{R}^2} k_{p,q}(u,v)\mathrm{d}u\mathrm{d}v = 1. \quad (2.8)$$

2.2.2 Color filter array

In the previous section, we only discussed about converting radiance into a digital signal assuming that the sensor can record the red, green and blue components of the incident light rays.

The CCD or CMOS-based technology presented above can, in fact, only record light intensity but cannot separate the intensity per color frequency. It is possible to stack at each pixel location on the sensor grid three local sensors, each one collecting photons for the red, green and blue light components thanks to band-pass filters. It however demands high-end sensors reserved for a limited pool of cameras that most of photographers cannot afford.

An alternative solution proposed by Bayer in 1976 [11] for shooting colored digital photographs with a single CDD or CMOS sensor per pixel location is to instead put before each photosite a band-pass filter for the red, green and blur components. The frequencies getting through the filter depends on the location of the photosite according to a given pattern all over the camera sensor surface, resulting in a colored mosaic called a color filter array (or *CFA*). For instance, at a given photosite location, the color red only is recorded whereas the other colored components are filtered out.

A second photosite next to the first one will, instead, record the green component of the incoming light, thus dumping the remaining red and blue colors.

Figure 2-8 shows an example of the Bayer CFA, repeating the same 2×2 pattern on the sensor composed of two green, one blue and one red pixels and directly takes inspiration from the density of cones and rods on the human eye retina [5]. This pattern is used in many camera brands such as Canon or Sony whereas other manufacturers, such as Fujifilm with their 6×6 X-trans pattern [45], use their own patterns. We focus in this thesis on the Bayer pattern, which is still nowadays the most commonly used CFA.

The missing colors are predicted with a *demosaicking* algorithm that may be based on bilinear interpolation [75], filter banks [83,91] or neural networks [45]. Each one of these techniques takes as input an image with missing color components and estimates a full RGB image.

Formally, the sensor image y_{sensor} from the previous section, is multiplied with a binary mask m of size $H \times W \times 3$ that sets to 0 the color components at each pixel location not retained by the CFA in front of the photosites, yielding the recorded raw image y_{raw}:

$$y_{\text{raw}} = m \odot y_{\text{sensor}} = m \odot Q\left[Kx + \varepsilon\right], \tag{2.9}$$

with $\odot$ the pixelwise product operator. The binary mask m sets to 0 the color components in the quantized image not covered by the mosaicking pattern just like the CFA is a pass-band, filter wiping out the color components not in the corresponding frequency interval at a given photosite.

The resulting image is a $H \times W \times 3$ array with zeros at locations where the binary mask m is 0 but is more frequently represented as a mosaicked $H \times W$ image. In the specific case of the Bayer pattern, it can alternatively be represented as a $H/2 \times W/2 \times 4$ image with R,G,G,B components.

2.2.3 Saturation

During exposure, photons are accumulated within the photosites, which have a finite capacity. In the case of long exposure, *e.g.,* several seconds, or when one shoots a particularly bright object such as a light bulb or the Sun, more and more photons are accumulated in the photosites and eventually overflow when the whole capacity allocated to each site is full. This phenomenon is called saturation and breaks the linear relationship between scene radiance and recorded electrical signal magnitude. We focus on this phenomenon in Chapter 5.

We upgrade the formation model of Eq. (2.9) to take into account this physical phenomenon by adding a clipping function s, defined as $s(x) = \min(x, 2^N - 1)$ for a N-bit sensor and applied pixelwise to the image (with the common assumption that the white value is set to 1) [40, 125]:

$$y_{\text{saturated}} = s(y_{\text{raw}}) = s(m \odot Q\,[Kx + \varepsilon]). \tag{2.10}$$

If no pixel is saturated in the y_{raw} image, this model boils down to the linear formation model of Eq. (2.9) and $y_{\text{raw}} = y_{\text{saturated}}$.

2.3 Image signal processing pipeline

The image signal processing (ISP) pipeline is the concatenation of several functions gradually converting the (possibly) raw image into a visually acceptable RGB image. Figure 2-1 details three main operations within the ISP pipeline: the purple block converts the values corresponding to the amount of photons accumulated in each photosite into pixel values, the green block predicts the missing color components for each pixel filtered out by the sensor CFA with a demosaicking algorithm and the blur block finally adjusts the color space and brightness of the digital image. Each manufacturer and each camera has its own ISP pipeline, making an exhaustive description of all the existing components impossible to achieve. We review in what follows the main stages, common to most of cameras.

The first stage consists in substracting the black level value provided by ISP to set the 0 in the image. The pixels under this values are clipped or sometimes kept for estimating the noise variance [40]. The white level similarly sets the maximal value, corresponding the white color. As discussed by Guillermo Luijk in his `dcraw` tutorial[3], it is crucial to correctly set this value since it flags the saturated pixel and prevents colored artifacts in the highlights such as magenta stains. White balance is then applied to revert the colored effects produced by the camera and render the image under neutral illumination [18], yielding a $H \times W$ array with minimum value set to 0 and maximal value rescaled to 1.

Demosaicking, as explained in the previous section, yields a RGB image further sharpened with a linear filter [62] to increase the sharpness lost during the antialiasing stage, just before integration on the camera sensor.

The demosaicked image is later rendered in the in-camera RGB space, which usually does not correspond to the actual scene colors. A color-conversion stage transforms the colors of the image at hand to fit the standard RGB colorspace (or sRGB) [114], that makes sure that any photograph is rendered in a common colorspace. This transformation takes the form of a 3×3 matrix, shipped with ISP and applied to each RGB pixel of the demosaicked image the corresponding triplet in the sRGB color space. The resulting image is still, up to the demosaicking stage, a linear output of the incoming radiance on the sensor. Yet, this linear relationship looks unnatural for the eyes since the darks are too dark and the lights are to bright. The relation between pixel intensity of the displayed image and the recorded electric signal should instead be non-linear to push darker and lighter pixels to the grays. In practice it is done with an S-shaped function such as the gamma correction function historically used for cathode-ray tubes television screens [101].

Formally, let G be the gain/white balance coefficient matrix, D a demosaicking algorithm, *e.g.,* [84] or [45], C the color conversion matrix and Γ the gamma correction

[3] http://guillermoluijk.com/tutorial/dcraw/index_en.htm

function. The final sRGB image, denoted by y_{sRGB}, is related to $y_{\text{saturated}}$ with

$$y_{\text{sRGB}} = \Gamma(C(D(G(y_{\text{saturated}})))). \tag{2.11}$$

A practical Matlab implementation of the ISP pipeline is detailed in [115]. Combining the saturated image formation model from the continuous image x in Eq. (2.10) and the conversion to an sRGB image in Eq. (2.11) yields a link between the analog observed image x and the digital sRGB image y_{sRGB} that we can display on a computer or smartphone screen.

These are the main steps of a typical ISP pipeline but in practice, manufacturers implements additional steps that may vary from one camera to another depending on the computational budget or the actual need of further corrections, *e.g,* high-dynamical range (HDR) imagery [101] or chromatic aberration removal [62]. Hasinoff *et al.* [53], for instance, give a brief summary of 13 in-camera stages for converting the raw image into a sRGB one in a recent flagship smartphone.

Chapter 3

Related Work

In this section, we review the literature related to this thesis. We first cover the choice of a blurry image formation model. We present the different typical approaches to modelling both uniform and non-uniform blur and the classical noise models used in image processing. We then review non-blind deblurring techniques for inverting these forward models, ranging from classical optimization-based deconvolution techniques to learning-based approaches. We finally survey several motion blur and camera PSF estimation techniques.

3.1 Approximate image formation model

As noted by Baker and Kanade [9], the analog, sharp focal image cannot be recovered since we can only predict discrete signals. A typical approximation in image processing [9] is to find a sharp discrete image instead.

Formally, we replace the forward model connecting the focal image x with a $H \times W \times 3$ (possibly saturated) image $y_{\text{saturated}}$ in (2.10) with a simpler equation featuring the blurry image y, a simplification of $y_{\text{saturated}}$, and x is now a $H \times W \times 3$ digital image too:

$$y = s(MKx + \varepsilon). \tag{3.1}$$

The $(3HW \times 3HW)$ matrix K implements the overall blur comprising both intrinsic

and extrinsic blurs, and M corresponds to the pointwise multiplication with a binary mask representing the camera's CFA. Finally, ε models the in-camera additive noise. We make the assumption that no multiplicative noise such as salt-and-pepper [27] degrades the images in this thesis, which is a reasonable assumption in most real-world scenarios. The function s is the same as in Eq. (2.10) and denote the saturation function defined pixelwise as $s(p) = \min(p, 1)$ for a pixel p. This formation model will be used in Chapters 5.

A classical approximation is to ignore saturation and consider y as an RGB image with floating-point pixel values in $[0, 1]^3$, simplifying the forward model as

$$y = Kx + \varepsilon. \tag{3.2}$$

This is the model typically considered in the deblurring community and is at the core of Chapter 4.

When the linear filter is the same for each pixel location in y, the blur is called *uniform*. However, when it varies depending on the blurry pixel location, it is *non-uniform*. We now cover different models for the linear operator K and the noise vector ε.

3.1.1 Noise models

Recorded images are in practice not only blurry but noisy as well.

Gaussian model. A typical image noise model for ε is a zero-mean Gaussian distribution with variance σ^2. This noise model is image-independent, *i.e.,* the pixel values of the image do not impact the intensity of noise. Injected in Eq. (3.1), this gives

$$y \sim \mathcal{N}(Kx, \sigma^2). \tag{3.3}$$

It is a common model for the stationary noise caused by the camera hardware, the *read* noise in the image processing literature, but it does not fit empirical image noise distributions in raw or demosaicked images [1, 40, 98]. Gaussian noise is however

widely used to train non-blind sRGB image deblurring approaches [37, 71, 137], generalizing well in practice on blurry RGB images. This noise model will be considered in Chapter 4.

Poissonian model. The Poissonian noise model with parameter λ traditionally models the noise to errors made by the sensor when counting the photons accumulated in a photosite. This noise is called *shot* noise in the image processing literature and is signal-dependent. Injected in Eq. (3.1), this gives [40]

$$y \sim \mathcal{P}(\lambda K x). \tag{3.4}$$

Deblurring methods explicitly handling Poissonian instead of Gaussian noise are less common because of this pixel-dependent aspect, greatly complicating the corresponding inverse problem [92].

An alternative strategy to handle images degraded with Poissonian noise is to convert them into a scene-independent Gaussian noise with a variance-stabilization transformation [8, 81]. As a result, the numerous of methods designed to address deblurring with the assumption of Gaussian noise can be directly used in this setting.

Poissonian-Gaussian model. Empirical noise recorded in raw images is actually caused by both shot and read noise and can thus be reasonably modelled with a mixture of independent Gaussian and Poissonian noises [40]. It can be reasonably approximated by a Gaussian distribution with pixel-varying variance [40]:

$$y \sim \mathcal{N}(Kx, \lambda K x + \sigma^2). \tag{3.5}$$

This model better fits the empirical distribution of noise, resulting in more realistic denoising evaluation datasets [2, 98] or learning-based models that better generalize to real data [18]. It will be used in Chapter 5.

3.1.2 Blur models

Blur typically softens abrupt changes in an image, for instance it erases details such as textures and edges, and thus roughly corresponds to a low-pass filter. Formally, at a given pixel location (u, v) in the image support, K in Eq. (3.1) corresponds to replacing a pixel value in the image x by a linear combination of the pixel values in the neighborhood of (u, v), which is easily implemented with a 2D convolution replacing the matrix multiplication Kx. The convolution features a linear filter $k_{u,v}$ whose coefficients are the local non-negative weights allocated to the pixels next to (u, v), and the sharp image x in image format. Based on Eq. (3.1) and on the equivalence between the matrix-vector product Kx and a convolution, generating a blurry pixel in y at location (u, v) reads

$$y[u, v] = (k_{u,v} * x)[u, v] + \varepsilon[u, v], \tag{3.6}$$

where $*$ denotes the 2D convolution operator. This model is attractive as it represents any kind of blur, but it is of course unrealistic in practice to use a different linear filter per pixel in the image. For a tiny 256×256 image, it would results in 65536 filters, which cannot be done in practice, in most cases.

We detail below three typical approximations, from the most aggressive one assuming uniform blur on the image, to an accumulation model directly inspired from integration over time of a sequence of focal images during exposure.

Global convolution. An aggressive, yet common, approximation is to assume that the blur is uniform. This corresponds to a global convolution which applies the same linear filter everywhere, *i.e.,* $k_{u,v} = k$ for all pixel locations (u, v). The forward model Eq. (3.1) becomes:

$$y = k * x + \varepsilon. \tag{3.7}$$

This model is favored by most blur estimation methods [21,26,39,58,68,95,133] since it only requires to predict a single linear filter for a given image. In this situation, the linear operator K implemementing the convolution becomes a circulant matrix that

can be diagonalized in the Fourier basis [46], and non-blind deblurring methods can thus be built on top of FFT for quickly computing inverse filters [127, 129].

Local convolution. The uniform blur assumption may not be realistic, for instance in the context of camera shake, where the motion field can be modelled by rotations about the optical center of the camera [126].

Another simplification is to assume that the blur is uniform over a set of (possibly overlapping) patches. For instance, if we assume that the blur is uniform on 9×9 patches, the number of linear filters to be computed in a 256×256 image drops to fewer than 1,000 filters. Given R patches, where the blur is supposed locally uniform and thus modelled by R linear filters $k_1, \ldots, k_R$, the forward model in Eq. (3.2) may be written instead as

$$y = \sum_{i=1}^{R} r_i(w_i \odot [k_i * c_i(x)]) + \varepsilon, \tag{3.8}$$

where c_i is a cropping routine that extracts the i-th region, *e.g.,* a patch, where the linear filter k_i approximates the real local blur, r_i puts back the patch at its initial location in the image and w_i is a weighting window [57] to prevent mismatches at the patch boundaries in the reconstructed image.

The efficient filter flow (EFF) of Hirsch *et al.* [57] is an instance of such an approximation, used to remove optical aberrations from 18-megapixel images taken with a DSLR in [109], where the camera PSF is supposed locally uniforms for overlapping patches with approximate size 500×500.

In the same vein, several works model the global motion blur with linear filters [17, 30, 37, 48, 65, 85, 116] applied to overlapping patches. These local linear filters can model global complex motions such as rotations and moving objects such as cars or people, and can be used to transform the initial regression problem of estimating the 65536 filters into a classification problem [30, 48, 116], easier to solve and less costly in terms of memory.

Accumulation model. In the case of motion blur, it is also possible to eliminate the operator K and directly blurs an image using an *accumulation* model [51, 60, 120,

126]. The sensor image is obtained by integrating focal images taken during exposure. The continuous stream of images is approximated by T digital images averaged as:

$$y = \frac{1}{T} \sum_{t=1}^{T} x_t + \varepsilon. \tag{3.9}$$

This model has the benefit to be more general for representing non-uniform blur than the locally uniform model of EFF [57] and does not require modeling or predicting linear filters.

This accumulation model is used to generate realistic blurry images by averaging consecutive frames of videos taken with a high-frequency camera such as the 240fps GoPro camera used in [87,123] or from the Internet [89]. One the one hand, this model possibly generates the best possible synthetic images featuring non-uniform motion. On the other hand, such a dataset may lack of diversity due to the limited pool of curated videos from which the images are taken from, *e.g.,* about thirty sequences for the GoPro dataset of [87]. It also requires to shoot videos at a very high-frame rate, for instance the 240fps mentioned above. When a correct speed is not reached, averaging consecutive frames may otherwise introduce ghosting artifacts [17].

3.2 Non-blind deblurring

Non-blind image deblurring was historically introduced by Wiener [127], Richardson [102] and Lucy [78] for deconvolution in scientific fields relying on imagery such as astronomy where the telescope's PSF is supposed known or can be easily estimated by observing a star for example. As we have seen with the HST example of Chapter 1, it is a reasonable assumption. Most recent non-blind deblurring techniques can be classified into three categories: optimization-based methods relying on a forward image model and priors, learning-based algorithms depending on supervisory data, and hybrid methods considering the two. We review in this section the literature related to each of these three paradigms.

3.2.1 Model-based methods

Model-based approaches leverage the forward model (3.1) to build an energy function solved with inverse problem techniques [6]. The first proposed non-blind deconvolution method is Wiener filtering [127], obtained as the solution of a least-squares problem.

More generally, an estimate of the sharp image is a minimizer of a composite energy function made of a data-fidelity term L and an image prior Ω, weighted by a scalar λ:

$$\min_{x \in \mathbb{R}^N} L(y, Kx) + \lambda \Omega(x). \tag{3.10}$$

When the noise model follows a Gaussian distribution, the data-term L takes the form of a ℓ_2 loss [6, 125]:

$$L(y, Kx) = \frac{1}{2} \sum_{i=1}^{N} ([y]_i - [Kx]_i)^2 = \frac{1}{2} \|y - Kx\|_F^2. \tag{3.11}$$

The minimizer is the solution of the following linear system:

$$K^\top K x = K^\top y. \tag{3.12}$$

The solution can be found using conjugate gradient (CG) [46, 124, 144], FFT [70, 129, 140] or fixed-point iterations [64] for example. The regularization term Ω, such as Tikhonov regularization [46], is often used to improve the conditioning of this system since deblurring is an ill-posed problem [74]. We discuss efficient linear system solvers in the context of non-blind deblurring in Chapter 4.

When the noise model follows a Poisson distribution instead, the data-fitting term becomes [102, 125]:

$$L(y, Kx) = -\sum_{i=1}^{N} ([y]_i \log([Kx]_i) - [Kx]_i). \tag{3.13}$$

The Richardson-Lucy (RL) algorithm [78, 102] is a classical iterative approach for minimizing this energy, based on gradient descent with an adaptive stepsize. An RL

iteration reads:

$$x^{(t+1)} = x^{(t)} \odot \left(\frac{K^\top y}{K^\top K x^{(t)}} \right), \tag{3.14}$$

where $\odot$ and the division bar are the pixelwise product and division.

The prior Ω typically encodes the specifics of the image restoration application, for example the fact that images should look like photographs. A common observation is that natural image gradients are sparse [70]. An often used prior is total-variation (TV) of an image, *i.e.,* the sum of the magnitude of all the gradients in an image for a given norm [82], which can be minimized by the mean of sparsity-inducing metrics, *e.g.* ℓ_1 [106, 124], ℓ_0 [130] or $\ell_{0.5}$ [70], and alternative finite-difference filters per color channel [16] or mixing different color channels [54]. A general form for TV-based priors uses a non-negative function ρ and a filter bank $\{G_i\}$ $(i = 1, \ldots, n)$ such that Ω can be written as

$$\Omega(x) = \sum_{i=1}^{n} \rho(G_i x). \tag{3.15}$$

For instance, for $n = 2$ with G_1 and G_2 corresponding to the convolution with finite-difference filters and ρ is the ℓ_1 norm, Ω boils down to the TV-ℓ_1 prior in [124]. When ρ is the ℓ_γ semi-norm with $0 < \gamma < 1$, Ω corresponds to the hyper-Laplacian prior of [70]. Priors specific to certain kinds of images have also been proposed, achieving superior results compared to the generic TV prior above. For instance the ℓ_0-based prior of Pan *et al.* [94] works particularly well on text images.

When the weight λ is greater than 0, the composite energy might not be easily minimized since it is a composite function with terms having different properties such as convexity and differentiability. Optimization can be carried out either with gradient descent or LBFGS iterations, for instance, if we assume that both L and Ω or differentiable or a splitting algorithm such as half-quadratic splitting (HQS) [43, 44] or the alternate direction multiplier method (ADMM) [13] otherwise. We explicitly treat the case of HQS in this thesis in Chapters 4 and 5.

Handcrafted priors, such as in Eq. (3.15), can only enforce a finite number of features typical of desired images such as natural photographs. Heide *et al.* [55] propose to build a new penalty term combining several priors, each one constraining

a few features such as strong edges with a TV prior [106, 124] or patch redundancy with a term derived from BM3D [31]. This combination may lead to superior results, both visually and quantitatively, but at the cost of solving composite and possibly non-convex problems with many penalty weights to handtune.

The choice of the prior is also subject to a trade-off between accuracy and computational speed. Indeed, the simplest image priors such as TV-ℓ_1 have closed-form proximal operators [124], resulting in fast algorithms, even for high-resolution images, whereas more elaborated priors, such as EPLL [144], can take several minutes or hours for a 800×800 image.

3.2.2 Learning-based methods

Synthetic training data composed of a blur operator, a sharp and a corresponding blurry image based on the image formation model (3.1), providing large datasets for training deblurring methods. Provided such corpus, supervised regression techniques are particularly appealing for image deblurring.

Takeda *et al.* [121] use a variant of the Nadaraya-Watson interpolation algorithm to predict sharp pixels. Couzinie-Devy *et al.* [29] learn a dictionary for a given set of blurry and sharp patches. Large-scale optimization is carried out with stochastic gradient descent. In [36], we cast non-blind image deblurring as a structured prediction problem, solved with kernel ridge regression.

The learning-based deblurring algorithms are, in general, parametric functions such as convolutional neural networks (CNNs), particularly adapted to process images in the computer vision literature. Suppose we have N triplets $(x^{(i)}, y^{(i)}, K^{(i)})$ $(i = 1, \ldots, N)$, composed of a sharp image, its blurry version and the blur operator, a parametric deblurring algorithm ϕ_θ , *e.g.,* a CNN, with parameter θ and a pixelwise loss L such as the ℓ_2 or ℓ_1 norm. The optimal parameter θ for a given dataset is obtained as the solution of

$$\min_\theta \frac{1}{N} \sum_{i=1}^{N} F(\phi_\theta(y^{(i)}, K^{(i)}), x^{(i)}). \tag{3.16}$$

CNNs for non-blind deblurring have been trained in two settings. The first one comprises two-stage methods composed of an initial simple and fast deconvolution methods, *e.g.,* Wiener filtering, followed by a CNN trained to further refine the image estimated at the first stage. For instance, Schuler *et al.* [108] corrects with a neural network the typical ringing artifacts introduced by a FFT-based inverse blur filter. Dong *et al.* [34] improve this method by alternating several times between FFT-based deconvolution and CNN-based post-processing. Xu *et al.* [131] train separately modules mimicking Wiener deconvolution with additional denoising and ringing artifact removal stages. Fine-tuning these modules together yields the final model. The second one corresponds instead to black-box CNNs for deblurring an image, without any interpretable component. These approaches, besides proposing adequate architectures, also focus on building better training data and loss functions. For instance, Nah *et al.* [87] use the accumulation model of Eq. (3.9) to generate realistic blurry images from videos taken with a 240fps GoPro cameras. They train a multiscale CNN with a loss function supervising the deblurred image at each scale. The loss function is composed of the classical ℓ_2 distance and an adversarial loss. Alternatively, Noroozi *et al.* [89] use videos from the Internet for generating the training data and Brooks and Barron [17] build blurry images from only two photographs of the same scene taken at different moments and interpolate the missing frames to generate blurry images.

Despite good quantitative results on evaluation benchmarks, learning-based methods are only competitive on images from the same distribution as the training data, limiting them to fields where a lot of curated pairs of sharp and corresponding images are available, unlike astronomy or microscopy.

3.2.3 Hybrid methods

The optimization-based methods minimize composite energy functions including weights, hard to handtune in practice, illustrated with the example of the parameter λ with the TVℓ_1 prior in Figure 1-2.

Bi-level optimization techniques [6] are hybrid techniques combining a typical

composite energy function for predicting a sharp image and a learning-based problem to estimate an optimal parameter, for instance the penalty weight λ in Eq.(3.15) [72], circumventing the difficulty of handtuning λ. In contrast to using handcrafted potentials ρ and n filters $G_1, \ldots, G_n$ in a classical TV prior in Eq.(3.15), a few works treat these quantities as a parameter learnt from training data. For instance, filters can be learnt from data in a Markov random field [141, 142] and pairs of filters/potentials with either Bayesian inference [105], in their so-called field-of-experts (FoE) prior. Zoran and Weiss [144] cluster hundreds of thousands of image patches to learn centroids further used in a prior implemented by a Gaussian mixture model. The sharp image is finally estimated with the HQS splitting method. Finally a "plug-and-play" method implements the prior Ω with a denoising routine [104], a choice motivated by the Bayesian perspective. Many denoisers can be work in practice, for instance a denoising CNN trained beforehand [138].

Traditional optimization algorithms are designed to iterate until convergence, but in practice a fixed number of iterations may be sufficient, especially in a context where the goal is to learn an appropriate prior. Gregor and Lecun [50] implement a classical iterative sparse coding algorithm ISTA [12], that typically converges after hundreds of iterations, as a recurrent neural network with finite depth, for instance ten or twenty. Each stage initially corresponds to an iteration of ISTA but is parameterized and updated with supervisory data. The resulting function approximates the performance of hundreds of ISTA iterations but with the running time and memory usage of only ten or twenty iterations.

The same approach is adopted in image processing: a finite number of iterations of optimization methods such as gradient descent or splitting algorithms are embedded in parametric functions, later learned in a regression setting with the backpropagation algorithm,

Chen and Pock [24] parameterize five stages of a diffusion equation with the learnable potential/filter pairs of the FoE model of [105] for image denoising and deblurring. Schmidt *et al.* [107] learn a model whose architecture combines HQS and the FoE prior of [105]. They also propose to supervise each stage separately, exploiting

the fact that each stage predicts an intermediate estimate of the solution. Zhang *et al.* [136] replace the soft-thresholding operator in the TV-ℓ_1 prior in HQS with a CNN, thus seen as the proximal operator of an unknown potential function. Kruse *et al.* [71] upgrade the learnable potentials of Schmidt *et al.* [107] with CNNs. Zhang *et al.* [137] also parameterize the proximal operator with a CNN but also predict the different hyper-parameters, with a second neural network.

3.3 Non-blind deblurring for saturated images

As discussed in Chapter 2, saturation breaks the classical linear model encountered in several image processing application and most be specifically handle by deblurring approaches. Indeed, deblurring algorithms listed above almost all involve local filtering operations that might mix saturated pixels in non-saturated regions, resulting in ringing artifacts according to Whyte *et al.* [125].

Cho *et al.* [27] deblur a saturated images by predicting with the expectation-maximization algorithm a sharp estimate and a mask of the saturated pixels in the latent sharp image. The methods leaves untouched the saturated pixels since the information is lost anyway and solely focuses on the non-saturated regions.

Whyte *et al.* [125] adapt the Richardson-Lucy [78, 102]. At each iteration, the mask of the unsaturated pixels is computed and two instances of the RL iteration is computed: one for updating the non-saturated "far" from the saturated regions and one for the non-saturated pixels at the vicinity of the saturated regions.

Mosleh *et al.* [86] include the clipping function in the data-fitting term of a penalized energy function and perform splitting over it to run the HQS algorithm.

3.4 Joint demosaicking and non-blind deblurring

The problem of joint demosaicking and non-blind deblurring is a much less studied problem than RGB image non-blind deblurring, but it is key to optical aberration removal, for instance, since demosaicking alone may mix the chromatic aberration

specific to a single color band, across the different color channels and thus may be at the origin of colored artifacts [109].

Palyi *et al.* [92] explain that no FFT-based solver can be found because the pixelwise mask M in Eq. (2.9) corresponding to the sensor CFA, cannot be as easily exploited in the Fourier domain as the blur operator K when it corresponds to a global convolution. The authors propose to instead jointly deblur and interpolate the color channels with a bank of approximate inverse deblurring filters combined with demosaicking interpolation kernels.

Schuler *et al.* [109] show that in the context of optical aberrations removal, good results can be achieved when the blur and mosaicking are jointly considered in a single linear operator instead of decomposing it into a blur operator K and a binary mask M as modelled in Eq. (2.9). They adapt the non-blind deblurring approach of Krishnan and Fergus [70] to the joint deblurring and demosaicking setting by replacing the initial FFT-based solver for deblurring with a CG-based one, which results in a slow restoration algorithms that takes several hours to restore a 18 Megapixel image from a standard DSLR. Soulez and Thiébaut [112] and Luong *et al.* [79] also cast joint non-blind deblurring and demosaicking as the minimization of penalized energies with dedicated priors on RGB images with optimization carried out respectively with LBFGS and a primal-dual algorithm.

Chi *et al.* [25] and Liang *et al.* [76] propose multiscale CNNs for motion blur removal from raw images, reminiscent of Nah *et al.* [87], but with additional submodules to handle the four color channels of a Bayer image. In the fashion of the GoPro dataset [87], Liang *et al.* collect sequences of sharp images to generate raw and blurry images.

3.5 Blur estimation

Non-blind deblurring techniques demand an estimate of the blur, which can be estimated thanks to prior knowledge in a specific field such as astronomy or microscopy. Once the camera or telescope has been calibrated and the blur measured, it can be

used for the other images. A second option is to estimate the blur from the degraded image at hand, which frequently happens for extrinsic blurs where the blur depends on shooting conditions specific to an image. We detail in this section only methods for estimating the blur from a single image since it is the setting relevant to this thesis.

3.5.1 Motion blur estimation

The simplest model for motion blur is to assume that it is uniform on the whole image. This assumption is generally incorrect but Hu and Yang [58] show that a uniform blur can be a good approximation in practice if it is estimated in an adequate region containing enough salient edges.

Fergus *et al.* [39] use the Bayes rule to relate the blurry image and the uniform blur kernel. The latter is predicted with probabilistic inference in a multiscale scheme. The blurry image is deblurred with a classical non-blind deconvolution technique. Despite good results, this approach remains extremely slow and can take several hours, even for low-resolution images.

Cho *et al.* [26] accelerate this method by finding the blur kernel as the solution of an inverse problem. This approach alternates between predicting salient edges in the current latent sharp image at hand with shock filters [90] and estimating in closed form the current blur kernel and sharp image. Blur estimation takes a few minutes for 1280×720 images, which is a dramatic improvement compared to the marginalization approach of Fergus *et al.* [39].

Xu *et al.* [129] further improve Cho *et al.*'s technique with a better edge detection approach than shock filters. Further improvements taking the form of an approximate ℓ_0 TV term in Xu *et al.* [133] or a dark channel-based prior by Pan *et al.* [95] yield state-of-the-art results for predicting motion blur from a single blurry image. These priors enforce sparsity on the latent image, forcing the solution to have non-zero components only next to the most salient edges. This allows the elimination of the edge-prediction stage in [26] and also dramatically improves blur prediction results overall. Contrary to previous sparsity-inducing priors, Michaeli and Irani [85] leverage

the redundancy of patches at several scales to predict local blurs.

Whyte *et al.* [126] adapt the multiscale approach of Cho *et al.* [26] to the non-uniform setting of camera shake, assumed to be reasonably modelled as a series of rotations about the camera's optical center with the accumulation model in Eq. (3.9). A set of angular positions about this optical center is first discretized and represents the possible positions a camera can have during motion. The blur kernel prediction stage of the Cho *et al.*'s technique is converted into estimating the most likely angular positions, resulting in a 3D blur model. The corresponding 3D blurring operator is implemented with a variant of the EFF model of Hirsch *et al.* [57] in Eq. (3.8). Overall, the method runs in about 3 minutes for a 1280×720 image and is, even nowadays, a state-of-the-art approach for camera shake removal.

Learning-based methods have also been employed for blur estimation but for specific sorts of blur. For instance, Couzinie-Devy *et al.* [30] predict horizontal translations, common in car images, with a linear classifier. Sun *et al.* [116] go a step beyond by learning a CNN to predict for a given a blurry image, a linear motion represented by an angle and a magnitude per overlapping patch. The CNN's local predictions are later fused with a random Markov field into a global non-uniform blur that represents camera translation, rotations or moving objects' trajectories. Gong *et al.* [48] propose a CNN that directly takes as input the blurry image and achieves in one stage the prediction of the locally-linear non-uniform blur.

Contrary to the algorithmic solutions covered in the previous paragraphs, [59, 60] use sensors to record the camera's motion during exposure.

We recall that in this thesis, we assume that images are in focus. We do not detail further methods for defocus blur estimation from a single image, *e.g.,* the work of Zhu *et al.* [143] or Levin *et al.* [73] estimating both defocus blur and depth at the same time.

3.5.2 Point-spread function estimation

Non-blind deblurring techniques are historically used in scientific domains where the intrinsic blur can be estimated, like in the HST example in Chapter 1.

More generally, optical aberration removal can be formulated as an inverse problem featuring color-specific blur kernels. Since consumer-grade lenses have locally-varying PSFs across the field of view, multiple measurements at different pixel locations are required to actually record an approximate global camera/lens PSF [10,109]. One of these measurements is shown in Figure 2-5.

Camera calibration techniques are limited to specific camera/lens pairs at a given aperture and focal length, which greatly limits the amount of images eligible for restoration. Conversely, the forward model (3.1) can also be used in an inverse problem to predict the camera PSF, assuming that a pair of corrupted and sharp images is available. Joshi *et al.* [61] use a calibration pattern and a photograph of it to build such pairs. Delbracio *et al.* [33] use a Bernoulli noise-based pattern to predict the PSF at subpixel resolution.

Contrary to the previous techniques requiring careful manipulations for each new corrupted image to record a ground-truth photograph, blind algorithms automatically estimate the PSF from a single blurry image, further used in an inverse problem with respect to the blur operator. Schuler *et al.* [110] estimate the PSF with an optimization-based formulation, solved with the multiscale variational approach of Cho *et al.* [26]. The vanilla algorithm takes a couple hours for a 18 megapixel image but can be accelerated by assuming that the PSF features several symmetries, reducing the running time to a couple of minutes.

This technique is further improved by Yue *et al.* [135] by assuming that the green channel is a sharp image and by imposing stricter symmetry constraints on the solution. The last assumption is actually not true in practice, because of manufacturing errors [10, 35]. Sun *et al.* [119] instead start by deconvolving, in a blind setting, one of the color channels, typically the green one supposed to be less blurry than the two other ones [54] and estimate the PSF for the two remaining channels by minimizing an energy function penalized with cross-color correlation term, achieving state-of-the-art optical aberration removal results for a single blurry RGB image.

The optimization approach of [110] and its follow-ups can only predict the global PSF at predefined locations on the field of view. Hirsch and Schölkopf [56] propose a

kernel regression technique to interpolate a local PSF at an unknown location given estimates elsewhere on the field of view. Shih *et al.* [111] propose an interpolation function that uses measurements of optical aberrations for certain lens settings to predict the blur at never-measured settings such as new focal length and aperture.

Chapter 4

An Interpretable Learning Approach to Non-blind Deblurring

Abstract

As already mentioned in Chapter 3, non-blind image deblurring is typically formulated as a linear least-squares problem regularized by natural image priors on the corresponding sharp picture's gradients, which can be solved, for example, using a half-quadratic splitting method with Richardson fixed-point iterations for its least-squares updates and a proximal operator for the auxiliary variable updates. In this chapter, we propose to precondition the Richardson solver using approximate inverse filters of the (known) blur and natural image prior kernels. Using convolutions instead of a generic linear preconditioner allows extremely efficient parameter sharing across the image, and leads to significant gains in accuracy and/or speed compared to classical FFT and conjugate-gradient methods. More importantly, the proposed architecture is easily adapted to learning both the preconditioner and the proximal operator (and thus, the corresponding prior) using CNN embeddings. This yields a simple and efficient algorithm for non-blind image deblurring which is fully interpretable, can be learned end to end, and whose accuracy matches or exceeds the state of the art, quite significantly in the non-uniform case.

4.1 Introduction

This chapter addresses the problem of non-blind image deblurring–that is, the recovery of a sharp image given its blurry version and the corresponding uniform or non-uniform motion blur kernel. Applications range from photography [59] to as-

tronomy [113] and microscopy [49]. As mentioned in the previous chapters, classical approaches to this problem include least-squares and Bayesian models, leading to Wiener [127] and Lucy-Richardson [102] deconvolution techniques for example. Since many sharp images can lead to the same blurry one, blur removal is an ill-posed problem. To tackle this issue, variational methods [106] inject *a priori* knowledge over the set of solutions using penalized least-squares. Geman and Yang [44] introduce an auxiliary variable to solve this problem by iteratively evaluating a proximal operator [96] and solving a least-squares problem. The rest of this chapter builds on this *half-quadratic splitting* approach. Its proximal part has received a lot of attention through the design of complex model-based [70, 117, 134, 144] or learning-based priors [105]. Far less attention has been paid to the solution of the companion least-squares problem, typically relying on techniques such as conjugate gradient (CG) descent [14] or fast Fourier transform (FFT) [57, 132]. CG is relatively slow in this context, and it does not exploit the fact that the linear operator corresponds to a convolution. FFT exploits this property but is only truly valid under periodic conditions at the boundaries, which are never respected by real images.

We propose instead to use Richardson fixed-point iterations [64] to solve the least-squares problem, using approximate inverse filters of the (known) blur and natural image prior kernels as preconditioners. Using convolutions instead of a traditional linear preconditioner allows efficient parameter sharing across the image, which leads to significant gains in accuracy and/or speed over FFT and conjugate-gradient methods. To further improve performance and leverage recent progress in deep learning, several recent approaches to denoising and deblurring unroll a finite number of proximal updates and least-squares minimization steps [4, 24, 67, 71, 107]. Compared to traditional convolutional neural networks (CNNs), these algorithms use interpretable components and produce intermediate feature maps that can be directly supervised during training [71, 107].

We propose a solver for non-blind deblurring, also based on the splitting scheme of [44] but, in addition to learning the proximal operator (and thus, the prior) as in [136], we also learn parameters in the fixed-point algorithm by embedding the

preconditioner into a CNN whose bottom layer's kernels are the approximate filters discussed above. Unlike the algorithm of [136], our algorithm is trainable end to end, and achieves accuracy that matches or exceeds the state of the art. Furthermore, in contrast to other state-of-the-art CNN-based methods [71, 136] relying on FFT, it operates in the pixel domain and thus easily extends to non-uniform blurs scenarios.

4.1.1 Contributions of this chapter

Our contributions can be summarized as follows.

- We introduce a convolutional preconditioner for fixed-point iterations that efficiently solves the least-squares problem arising in splitting algorithms to minimize penalized energies. It is faster and/or more accurate than FFT and CG for the non-blind deblurring task, with theoretical convergence guarantees.

- We propose a new end-to-end trainable algorithm that implements a finite number of stages of half-quadratic splitting [44] and is fully interpretable. It alternates between proximal updates and preconditioned fixed-point iterations. The proximal operator and linear preconditioner are parameterized by CNNs in order to learn these functions from a training set of clean and blurry images.

- We evaluate our approach on several benchmarks with both uniform and non-uniform blur kernels. We demonstrate its robustness to significant levels of noise, and obtain results that are competitive with the state of the art for uniform blur and significantly outperforms it in the non-uniform case.

4.2 Deconvolution algorithms

We survey in this section several approaches for solving a least-squares problem. We focus on FFT, CG and fixed-point iterations, at the core of this chapter. In this chapter, we will consider the simplified forward model of Eq. (3.2) which only features a blur matrix K applied to the sharp image in vector format x. We also set ourselves

in the case where Kx is equivalent to the convolution of x, in image format, with a linear k, thus $k * x$, which corresponds to the uniform blur model in Eq. (3.7). In this section, we will often switch between both formats, using the image format to derive the practical algorithms and the matrix format when dealing with the underlying optimization theory.

4.2.1 Classical approaches

We consider the forward model of Eq. (3.2) where the blur matrix K corresponds to the convolution with a linear filter k as detailed in Eq. (3.7). In the noise-free setting, it reads

$$y = k * x, \tag{4.1}$$

We make the assumption that x and y are $H \times W$ grayscale images but extending this model to RGB images is straightforward and is thus not explicitly covered in this section.

We recall from Chapter 3 that a classical approach to find x in this case is to solve the least-squares problem

$$\min_{x \in \mathbb{R}^{H \times W}} \|y - k * x\|_F^2, \tag{4.2}$$

or equivalently in matrix format

$$\min_{x \in \mathbb{R}^{HW}} \|y - Kx\|_F^2. \tag{4.3}$$

The solution of this least-squares problem has a closed form, derived from the blur matrix pseudo-inverse [46], but in general computing the pseudo-inverse can be prohibitive since it requires inverting a large matrix. In the case of the uniform blur model we consider, we can use the Parseval equivalence [82] to write Eq. (4.2) in the Fourier domain where the convolution becomes a pointwise multiplication. Let $\mathcal{F}(x)$ be the 2D Fourier transform of the image x and $\odot$ the pointwise product. Equation (4.2) becomes

$$\min_{\mathcal{F}(x)} \|\mathcal{F}(y) - \mathcal{F}(k) \odot \mathcal{F}(x)\|_F^2. \tag{4.4}$$

A few computations yields the closed-form solution

$$\widehat{x} = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(k)^*}{|\mathcal{F}(k)|^2} \odot \mathcal{F}(y)\right), \tag{4.5}$$

where $\mathcal{F}^{-1}$ is the inverse Fourier transform, $\mathcal{F}(k)^*$ is the complex conjugate of $\mathcal{F}(k)$ and $|\mathcal{F}(k)|^2$ is the pixelwise modulus of $\mathcal{F}(k)$. The division is pixelwise and since $\mathcal{F}$ is in practice implemented with FFT, Eq. 4.5 can be computed in a few milliseconds, even for high-resolution images. Despite excellent speed and accuracy, the Fourier transform and FFT are valid for an image with circular boundary conditions, which is generally not true in practice, resulting in artifacts next to the edges of the restored image $\widehat{x}$.

Iterative methods are classical alternative solvers for finding a solution to the normal equation related

$$(K^\top K)x = K^T y, \tag{4.6}$$

As we have seen in Chapter 3, the general blur matrix K has in practice poor conditioning. Using iterative methods to solve a normal equation corresponds to solve a linear system with a linear operator $K^\top K$ which has even poorer conditioning, yielding slow convergence.

4.2.2 Convolutional fixed-point iterations

We propose an alternate iterative scheme in this section for solving a least-squares problem where the matrix corresponds to the convolution with a linear filter, based on preconditionned Richardson fixed-point [64]. Starting with $x_0 = y$, it reads in matrix format:

$$x_{t+1} = (I - CK)x_t + Cy. \tag{4.7}$$

A valid preconditioner C is an approximate inverse of K [64], solution of

$$\min_C \|I - CK\|_F^2 + \rho\|C\|_F^2. \tag{4.8}$$

Figure 4-1: Comparison of three deconvolution methods for inverting the forward model (4.1) in a synthetic blurry image. The proposed CPCR method achieves a similar visual result compared to CG and avoids the boundary artifacts of the FFT-based image.

Since K and I are circulant, it is reasonable to find C as a circulant matrix too, a typical assumption for preconditionning CG iterations when the linear operator is a block-Toeplitz matrix [23]. In image format, it thus corresponds to find a linear filter c, solution of

$$\min_{C} \|\delta - c * k\|_F^2 + \rho\|c\|_F^2, \tag{4.9}$$

whose closed-form solution can be safely computed with FFT since the boundaries of a linear filter are in general circular

$$c = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(k)^*}{|\mathcal{F}(k)|^2 + \rho J}\right), \tag{4.10}$$

with J a matrix full of ones. Kelly [64] shows that the iteration (4.7) with C computed as the solution of (4.9) is a converging sequence. For now, we assume that K corresponds to the uniform convolution with a linear filter k, defined in Eq. (3.7). The vectors x_t and y are now $H \times W$ images. In the image format, the rearranged Richardson iterations read

$$x_{t+1} = (\delta - c * k) * x_t + c * y, \tag{4.11}$$

$$= x_t + c * (y - k * x_t). \tag{4.12}$$

This iteration can be seen as an iterative residual update algorithm, gradually restoring x_t. We call these iterations convolutional preconditionned Richardson (CPCR) iterations.

We illustrate the performance of FFT and 100 iterations of CG and CPCR on a synthetic image in Figure 4-1. FFT, despite specific padding, introduces boundaries artifacts because of the non-circular edges of this image, which is indeed typical of natural images. CG and CPCR achieve similar visual and quantitative results but 100 CPCR iterations run in about 4 seconds on a CPU, in contrast to the 22 seconds for 100 CG iterations.

4.3 Proposed method

We now incorporate CPCR iterations in a penalized deconvolution scheme since the deconvolution is an ill-posed problem. We formulate non-blind deblurring as

$$\min_x \frac{1}{2}||y - k * x||_F^2 + \lambda\Omega(\sum_{i=1}^{n} k_i * x). \tag{4.13}$$

The filters k_i $(i = 1, \ldots, n)$ are typically partial derivative operators, and Ω acts as a regularizer on x, enforcing natural image priors. One often takes $\Omega(z) = ||z||_1$ (TV-ℓ_1 model). We propose in this section an end-to-end learnable variant of the method of *half-quadratic splitting* (or *HQS*) [44] to solve Eq. (4.13). As shown later, a key to the effectiveness of our algorithm is that all linear operations are explicitly

represented by convolutions.

Let us first introduce notations that will simplify the presentation. Given some linear filters a_i and b_i $(i = 0, \ldots, n)$ with finite support (square matrices), we borrow the Matlab notation for "stacked" linear operators, and denote by $A = [a_0, \ldots, a_n]$ and $B = [b_0; \ldots; b_n]$ the (convolution) operators respectively obtained by stacking "horizontally" and "vertically" these filters, whose responses are

$$A * x = [a_0 * x, \ldots, a_n * x];\ B * x = [(b_0 * x)^\top, \ldots, (b_n * x)^\top]^\top; \tag{4.14}$$

We also define $A * B = \sum_{i=0}^{n} a_i * b_i$ and easily verify that $(A * B) * x = A * (B * x)$.

4.3.1 A convolutional HQS algorithm

Equation (4.13) can be rewritten as

$$\min_{x,z} \frac{1}{2}||y - k * x||_F^2 + \lambda\Omega(z) \text{ such that } z = F * x, \tag{4.15}$$

where $F = [k_1; \ldots; k_n]$. Let us define the energy function

$$E(x, z, \mu) = \frac{1}{2}||y - k * x||_F^2 + \lambda\Omega(z) + \frac{\mu}{2}||z - F * x||_F^2. \tag{4.16}$$

Given some initial guess x for the sharp image, (*e.g.* $x = y$) we can now solve our original problem using the *HQS* method [44] with T iterations of the form

$$\begin{aligned} &z \leftarrow \text{argmin}_z\, E(x, z, \mu); \\ &x \leftarrow \text{argmin}_x\, E(x, z, \mu); \\ &\mu \leftarrow \mu + \delta t. \end{aligned} \tag{4.17}$$

The μ update can vary with iterations but must be positive. We could also use the alternating direction method of multipliers (or *ADMM* [96]), for example, but this is

left to future work. Note that the update in z has the form

$$z \leftarrow \underset{z}{\operatorname{argmin}} \frac{\mu}{2}||z - F * x||_F^2 + \lambda\Omega(z) = \varphi_{\lambda/\mu}(F * x), \tag{4.18}$$

where $\varphi_{\lambda/\mu}$ is, by definition, the *proximal operator* [96] associated with Ω (a soft-thresholding function in the case of the ℓ_1 norm [38]) given λ and μ.

The update in x can be written as the solution of a linear least-squares problem:

$$x \leftarrow \underset{x}{\operatorname{argmin}} \frac{1}{2}||u - L * x||_F^2, \tag{4.19}$$

where $u = [y; \sqrt{\mu}z]$ and $L = [k; \sqrt{\mu}F]$.

4.3.2 Convolutional PCR iterations

Many methods are of course available for solving Eq. (4.19). We propose to compute x as the solution of $C * (u - L * x) = 0$, where $C = [c_0, \ldots, c_n]$ is composed of $n + 1$ filters and is used in *preconditioned Richardson* (or *PCR*) fixed-point iterations [64].

Briefly, in the generic linear case, PCR is an iterative method for solving a square, nonsingular system of linear equations $Ax = b$. Given some initial estimate $x = x_0$ of the unknown x, it repeatedly applies the iterations

$$x \leftarrow x - C(Ax - b), \tag{4.20}$$

where C is a preconditioning square matrix. When C is an *approximate inverse* of A, that is, when the spectral radius η of $I - CA$ is smaller than one, preconditioned Richardson iterations converge to the solution of $Ax = b$ with a linear rate proportional to η [64]. When A is an $m \times n$ matrix with $m \geq n$, and x and b are respectively elements of $\mathbb{R}^n$ and $\mathbb{R}^m$, PCR can also be used in Cimmino's algorithm for linear least-squares, where the solution of $\min_x ||Ax - b||^2$ is found using $C = \rho A^\top$, with $\rho > 0$ sufficiently small, as the solution of $A^\top Ax - A^\top b = 0$, with similar guarantees. Finally, it is also possible to use a different $n \times m$ matrix. When the spectral radius η of the $n \times n$ matrix $I - CA$ is smaller than one, the PCR iterations converge once again

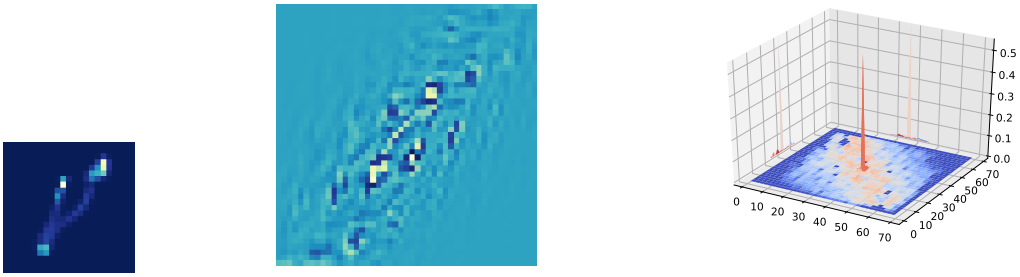


Figure 4-2: From left ot right: An example of a blur kernel k from the Levin *et al.* dataset [74]; its approximate inverse kernel c_0; the resulting filter resulting from the convolution of k and c_0 (represented as a surface). It gives an approximate Dirac filter δ.

at a linear rate proportional to η. However, they converge to the (unique in general) solution of $C(Ax - b) = 0$, which may of course be different from the least-squares solution.

This method is easily adapted to our context. Since L corresponds to a bank of filters of size $w_k \times w_k$, it is natural to take $C = [c_0, \ldots, c_n]$ to be another bank of $n + 1$ linear filters of size $w_c \times w_c$. Unlike a generic linear preconditioner satisfying $CA \approx \mathrm{Id}$ in matrix form, whose size depends on the square of the image size, C exploits the structure of L and is a linear operator with *much* fewer parameters, *i.e.* $n + 1$ times the size of the c_i's. Thus, C is an approximate inverse filter bank for L, in the sense that

$$\delta \approx L * C = C * L = c_0 * k + \sqrt{\mu} \sum_{i=1}^{n} c_i * k_i, \tag{4.21}$$

where δ is the Dirac filter. In this setting, C is computed as the solution of

$$C = \mathrm{argmin}_C \, ||\delta - L * C||_F^2 + \rho \sum_{i=0}^{n} ||c_i||_F^2, \tag{4.22}$$

The classical solution using the pseudo inverse of L has cost $\mathcal{O}\left((w_k + w_c - 1)^{2\times 3}\right)$.

$$c_i = \mathcal{F}^{-1}\left(\frac{\tilde{K}_i^*}{\rho J + \sum_{j=0}^{n} |\tilde{K}_j|^2}\right) \quad \text{for } i = 0 \text{ to } n, \tag{4.23}$$

using the fast Fourier transform (FFT) with cost $\mathcal{O}(w_c^2 \log(w_c))$ [41]. $\mathcal{F}^{-1}$ is the

inverse Fourier transform, J is a matrix full of ones, $\tilde{K}_i$ is the Fourier transform of k_i (with $k_0 = k$), $\tilde{K}_i^*$ is its complex conjugate and the division in the Fourier domain is entrywise. Note that the use of FFT in this context has nothing to do with its use as a deconvolution tool for solving Eq. (4.19). Figure 4-2 shows an example of a blur kernel from [74], its approximate inverse when $n = 0$ and the result of their convolution. Let us define $[A]_*$ as the linear operator such that $[A]_* B = A * B$. Indeed, a *true* inverse filter bank such that equality holds in Eq. (4.21) does not exist in general (*e.g.,* a Gaussian filter cannot be inverted), but all that matters is that the linear operator associated with $\delta - C * L$ has a spectral radius smaller than one [64]. We have the following result.

Lemma 4.3.1 *The spectral radius of the linear operator* $I - [L]_*[C]_*$*, where* C *is the optimal solution of* (4.22) *given by* (4.23) *is always smaller than* 1 *when* $[L]_*$ *has full rank.*

A detailed proof can be found in Appendix A. We now have our basic non-blind deblurring algorithm, in the form of the Matlab-style CHQS (for *convolutional HQS*, primary) and CPCR (for *convolutional PCR*, auxiliary) functions below.

function $x = \text{CHQS}(y, k, F, \mu 0)$
$x = y$; $\mu = \mu 0$;
for $t = 0 : T - 1$ **do**
    $u = [y; \sqrt{\mu}\varphi_{\lambda/\mu}(F * x)]$;
    $L = [k; \sqrt{\mu}F]$;
    $C = \text{argmin}_C \, ||\delta - C * L||_F^2 + \rho \sum_{i=0}^{n} ||c_i||_F^2$;
    $x = \text{CPCR}(L, u, C, x)$;
    $\mu = \mu + \delta_t$;
end for
end function

function $x = \text{CPCR}(A, b, C, x_0)$

$x = x_0$;

for $s = 0 : S - 1$ **do**

$x = x - C * (A * x - b)$;

end for

end function

4.3.3 An end-to-end trainable CHQS algorithm

To improve on this method, we propose to learn the proximal operator φ and the preconditioning operator C. The corresponding *learnable* CHQS (LCHQS) algorithm can now be written as a function with two additional parameters θ and ν as follows.

function $x = \text{LCHQS}(y, k, F, \mu_0, \theta, \nu)$

$x = y$; $\mu = \mu_0$;

for $t = 0 : T - 1$ **do**

$u = [y; \sqrt{\mu}\varphi^{\theta}_{\lambda/\mu}(F * x)]$;

$L = [k; \sqrt{\mu}F]$;

$C = \text{argmin}_C \, ||\delta - C * L||_F^2 + \rho \sum_{i=0}^{n} ||c_i||_F^2$;

$x = \text{CPCR}(L, u, \psi^{\nu}(C), x)$;

$\mu = \mu + \delta_t$;

end for

end function

The function LCHQS has the same structure as CHQS but now uses two parameterized embedding functions φ^{θ}_{τ} and ψ^{ν} for the proximal operator and preconditioner. In practice, these functions are CNNs with learnable parameters θ and ν as detailed in Sec. 4.4. Note that θ actually determines the regularizer through its proximal operator. The function LCHQS is differentiable with respect to both its θ and ν parameters. Given a set of training triplets $(x^{(i)}, y^{(i)}, k^{(i)})$ (in $i = 1, \ldots, N$), the parameters θ and ν can thus be learned end-to-end by minimizing

$$F(\theta, \nu) = \sum_{i=1}^{N} ||x^{(i)} - \text{LCHQS}(y^{(i)}, k^{(i)}, F, \theta, \nu)||_1, \tag{4.24}$$

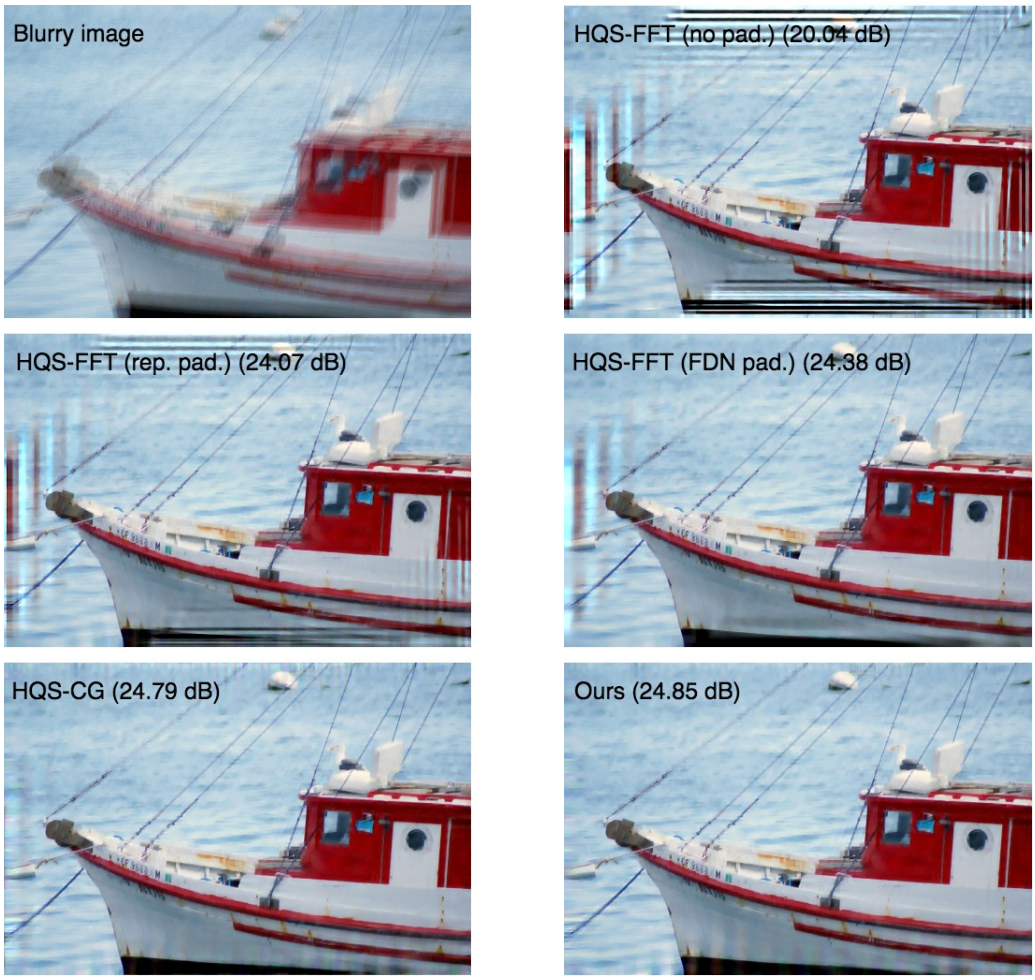


Figure 4-3: A blurry image from our test set with 2% white noise and the solutions of Eq. (4.13) with TV-ℓ_1 regularization obtained with different HQS-based methods. From the same optimization problem, HQS-FFT displays boundary artifacts. HQS-CG and CHQS produce images with similar visual quality and PSNR values but HQS-CG is much slower.

with respect to these two parameters by "unrolling" the HQS iterations and using backpropagation, as in [24, 136] for example. This can be thought of as the "compilation" of a fully interpretable iterative optimization algorithm into a CNN architecture. Empirically, we have found that the ℓ_1 norm gives better results than the ℓ_2 norm in Eq. (4.24).

4.4 Experiments

4.4.1 Implementation details

Network architectures. The global architecture of LCHQS shares the same pattern than FCNN [136], *i.e.*, $n = 2$ in Eq. (4.13) with $k_1 = [1, -1]$ and $k_2 = k_1^\top$, and the model repeats between 1 and 5 stages alternatively solving the proximal problem (4.18) and the linear least-squares problem (4.19). The proximal operator φ^θ is the same as the one introduced in [136], and it is composed of 6 convolutional layers with 32 channels and 3×3 kernels, followed by ReLU non-linearities, except for the last one. The first layer has 1 input channel and the last layer has 1 output channel. The network ψ^ν featured in LCHQS is composed of 6 convolutional layers with 32 channels and 3×3 kernels, followed by ReLU non-linearities, except for the last one. The first layer has $n+1$ input channels (3 in practice with the setting detailed above) corresponding to the filtered versions of x with the c_i's, and the last layer has 1 output channel. The filters c_1 and c_2 are of size 31×31. This size is intentionally made relatively large compared to the sizes of k_1 and k_2 because inverse filters might have infinite support in principle. The size of c_0 is twice the size of the blur kernel k. This choice will be explained in Sec. 4.4.2. In our implementation, each LCHQS stage has its own θ and ν parameters. The non-learnable CHQS module solves a TV-ℓ_1 problem; the proximal step implements the soft-thresholding operation φ with parameter λ/μ and the least-squares step implements CPCR. The choice of μ will be detailed below.

Datasets. The training set for uniform blur is made of 3000 patches of size 180×180 taken from BSD500 dataset and as many random 41×41 blur kernels synthesized with the code of [21]. We compute ahead of time the corresponding inverse filters c_i and set the size of c_0 to be 83×83 with Eq. (4.23) where ρ is set to 0.05, a value we have chosen after cross-validation on a separate test set. We also create a training set for non-uniform motion blur removal made of 3000 180×180 images synthesized with the code of [48] with a locally linear motion of maximal magnitude of 35 pixels. For both training sets, the validation sets are made of 600 additional samples. In both

cases, we add Gaussian noise with standard deviation matching that of the test data. We randomly flip and rotate by 90° the training samples and take 170×170 random crops for data augmentation.

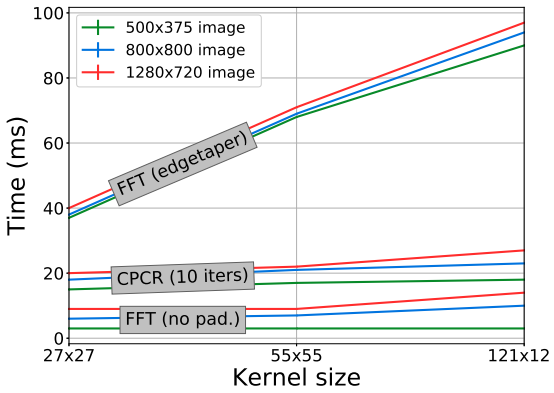
Optimization. Following [71], we train our model in a two-step fashion: First, we supervise the sharp estimate output by each iteration of LCHQS in the manner of [71] with Eq. (4.24). We use an Adam optimizer with learning rate of 10^{-4} and batch size of 1 for 200 epochs. Second, we further train the network by supervising the final output of LCHQS with Eq. (4.24) on the same training dataset with an Adam optimizer and learning rate set to 10^{-5} for 100 more epochs *without* the per-layer supervision. We have obtained better results with this setting than using either of the two steps separately.

4.4.2 Experimental validation of CPCR and CHQS

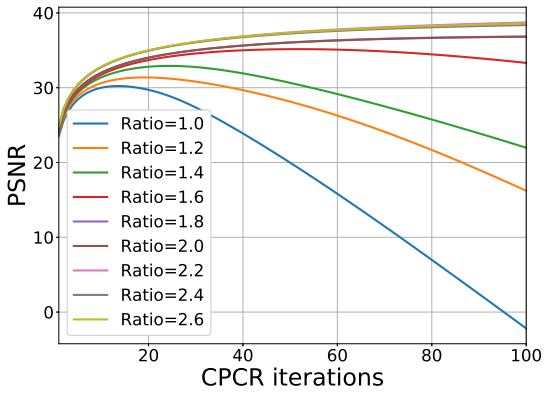
In this section, we present an experimental sanity check of CPCR for solving (4.19) and CHQS for solving (4.13) in the context of a basic TV-ℓ_1 problem.

Inverse kernel size. We test different sizes for the $w_{c_0} \times w_{c_0}$ approximate inverse filter c_0 associated with a $w_k \times w_k$ blur kernel k, in the non-penalized case, with $\lambda = 0$. We use Eq. (4.23) with ρ set to 0.05. We use 160 images obtained by applying the 8 kernels of [74] to 20 images from the Pascal VOC 2012 dataset. As shown in Fig. 4-4 , the PSNR increases with increasing w_{c_0}/w_k ratios, but saturates when the ratio is larger than 2.2. We use a ratio of 2 which is a good compromise between accuracy and speed.

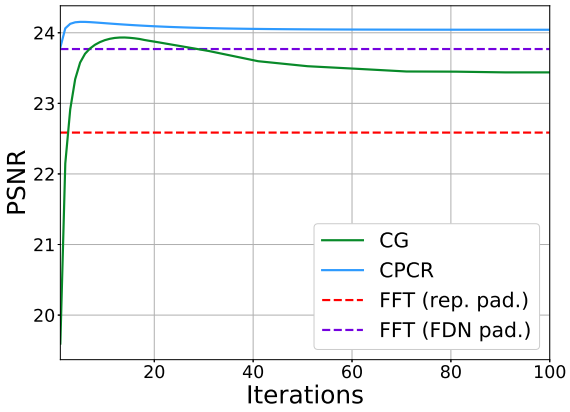
CPCR accuracy. We compare the proposed CPCR method to FFT-based deconvolution (FFT) and conjugate gradient descent (CG), to solve the least-squares problem of Eq. (4.19) in the setting of a TV-ℓ_1 problem. We follow [71] and, in order to limit boundary artifacts for FFT, we pad the images to be restored by replicating the pixels on the boundary with a margin of half the size of the blur kernel and then use the "edgetaper" routine. We also run FFT on images padded with the "replicate" strategy consisting in simply replicating the pixels on the boundary. We solve Eq. (4.19) with $\mu_0 = 0.008$, $\lambda = 0.003$ and z computed beforehand with



(a) Running time comparison.



(b) Inverse kernel size ratio impact.



(c) Accuracy comparison for deconvolution.

Figure 4-4: From left to right: Computation times for CPCR (including computation of C) with FFT applied on images padded with "edgetaper" as recommended in [71] and non-padded images for three image formats; effect of the w_{c_0}/w_k ratio on performance; comparison of CG, FFT and CPCR for solving (4.19).

Eq. (4.18). The 160 images previously synthesized are degraded with 2% additional white noise. Figure 4-4 shows the average PSNR scores for the three algorithms optimizing Eq. (4.19). After only 5 iterations, CPCR produces an average PSNR higher than the other methods and converges after 10 iterations. The "edgetaper" padding is crucial for FFT to compete with CG and CPCR by reducing the amount of border artifacts in the solution.

CPCR running time. CPCR relies on convolutions and thus greatly benefits from GPU acceleration. For instance, for small images of size 500×375 and a blur kernel of size 55×55, 10 iterations of CPCR are in the ballpark of FFT without padding: CPCR runs in 20ms, FFT runs in 3ms and FFT with "edgetaper" padding

	ker-1	ker-2	ker-3	ker-4	ker-5	ker-6	ker-7	ker-8	Aver.	Time (s)
HQS-FFT (no pad.)	21.14	20.51	22.31	18.21	23.36	20.01	19.93	19.02	20.69	0.07
HQS-FFT (rep. pad.)	26.45	25.39	**26.27**	22.75	27.64	27.26	24.84	23.54	25.53	0.07
HQS-FFT (FDN pad.)	**26.48**	25.89	**26.27**	23.79	27.66	27.23	25.26	25.02	25.96	0.15
HQS-CG	26.39	25.90	26.24	24.88	27.59	27.31	25.39	25.19	26.12	13
CHQS	26.45	**25.96**	26.26	**25.06**	**27.67**	**27.51**	**25.81**	**25.48**	**26.27**	0.26

Table 4.1: Comparison of different methods optimizing the same TV-ℓ_1 deconvolution model (4.13) on 160 synthetic blurry images with 2% white noise. We run all the methods on a GPU. The running times are for a 500×375 RGB image.

takes 40ms. For a high-resolution 1280×720 image and the same blur kernel, 10 iterations of CPCR run in 22ms, FFT without padding runs in 10ms and "edgetaper" padded FFT in 70ms. Figure 4-4 compares the running times of CPCR (run for 10 iterations) with padded/non-padded FFT for three image (resp. kernel) sizes: 500×375, 800×800 and 1280×720 (27×27, 55×55 and 121×121) pixels. Our method is marginally slower than FFT without padding in every configuration (within a margin of 20ms) but becomes much faster than FFT combined to "edgetaper" padding when the size of the kernel increases. FFT with "replicate" padding runs in about the same time as FFT (no pad) and thus is not shown in Fig. 4-4 . The times have been averaged over 1000 runs.

Running times for computing the inverse kernels with Eq. (4.23)**.** Computing the inverse kernels c_i, with an ratio w_c/w_k set to 2, takes 1.0 ± 0.2ms for a blur kernel k of size 27×27 and 5.4 ± 0.5ms (results averaged in 1000 runs) for a large 121×121 kernel. Thus, the time for inverting blur kernels is negligible in the overall pipeline.

CHQS validation. We compare several iterations of HQS using unpadded FFT (HQS-FFT (no pad.)), with "replicate" padding (HQS-FFT (rep. pad)), and the padding strategy proposed in [71] (HQS-FFT (FDN pad.)), CG (HQS-CG), or CPCR (CHQS) for solving the least-squares problem penalized with the TV-ℓ_1 regularized in Eq. (4.13) and use the same 160 blurry and noisy images than in previous paragraph as test set. We set the number of HQS iterations T to 10, run CPCR for 5 iterations and CG for at most 100 iterations. We use $\lambda = 0.003$ and $\mu_t = 0.008 \times 4^t$ ($t =$

	FCNN [136]	EPLL [144]	RGCD [47]	FDN [71]	CHQS	LCHQS$_{\text{G}}$	LCHQS$_{\text{F}}$
Levin [74]	33.08	34.82	33.73	35.09	32.12	<u>35.11 ± 0.05</u>	**35.15 ± 0.04**
Sun [118]	32.24	32.46	31.95	32.67	30.36	<u>32.83 ± 0.01</u>	**32.93 ± 0.01**

Table 4.2: PSNR scores for Levin [74] and Sun [118] benchmarks, that respectively feature 0.5% and 1% noise. Best results are shown in bold, second-best underlined. The difference may not always be significant between FDN and LCHQS for the Levin dataset.

$0, \ldots, T-1$). Table 4.1 compares the average PSNR scores obtained with the different HQS algorithms over the test set. As expected, FDN padding greatly improves HQS-FFT results on larger kernels over naive "replicate" padding, *i.e.* "ker-4" and "ker-8", but overall does not perform as well as CHQS. For kernels 1, 2, 3 and 5, the four methods yield comparable results (within 0.1 dB of each other). FFT-based methods are significantly worse on the other four, whereas our method gives better results than HQS-CG in general, but is 100 times faster. This large speed-up is explained by the convolutional structure of CPCR whereas CG involves large matrix inversions and multiplications. Figure 4-3 shows a deblurring example from the test set. HQS-FFT (with FDN padding strategy), even with the refined padding technique of [71], produces a solution with boundary artifacts. Both HQS-CG and CHQS restore the image with a limited amount of artifacts, but CHQS does it much faster than HQS-CG. This is typical of our experiments in practice.

Discussion. These experiments show that CPCR always gives better results than CG in terms of PSNR, sometimes by a significant margin, and it is about 50 times faster. This suggests that CPCR may, more generally, be preferable to CG for linear least-squares problems when the linear operator is a convolution. CPCR also dramatically benefits from its convolutional implementation on a GPU with speed similar to FFT and is even faster than FFT with FDN padding for large kernels. These experiments also show that CHQS surpasses, in general, HQS-CG and HQS-FFT for deblurring.

Next, we further improve the accuracy of CHQS using supervised learning, as done in previous works blending within a single model variational methods and learning.

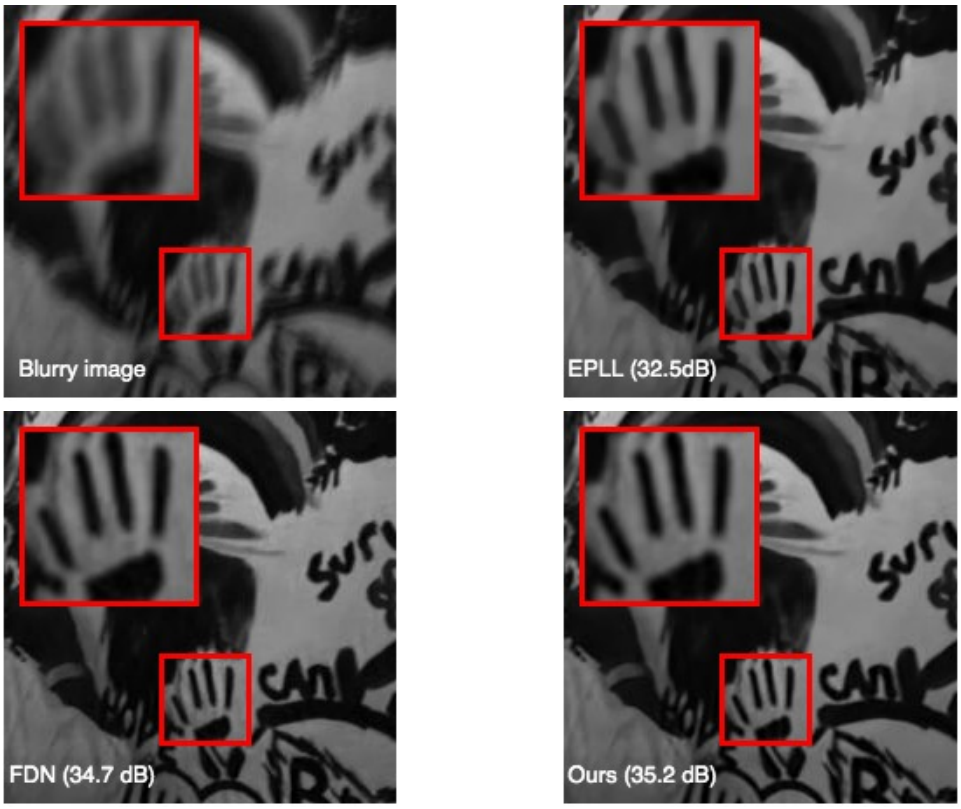


Figure 4-5: Comparison of state-of-the-art methods and the proposed LCHQS for one sample of the Levin dataset [74] (better seen on a computer screen). FDN effectively removes the blur but introduces artifacts in flat areas, unlike EPLL and LCHQS.

4.4.3 Uniform deblurring

We compare in this section CHQS and its learnable version LCHQS with the non-blind deblurring state of the art, including optimization-based and CNN-based algorithms.

Comparison on standard benchmarks. LCHQS is first trained by using the loss of Eq. (4.24) to supervise the output of each stage of the proposed model and second trained by only supervising the output of the final layer, in the manner of [71]. The model trained in the first regime is named LCHQS_G and the one further trained with the second regime is named LCHQS_F. The other methods we compare our learnable model to are HQS algorithms solving a TV-ℓ_1 problem: HQS-FFT (with the padding strategy of [71]), HQS-CG and CHQS, an HQS algorithm with a prior over patches (EPLL) [144] and the state-of-the-art CNN-based deblurring methods FCNN [136] and FDN [71]. We use the best model provided by the authors

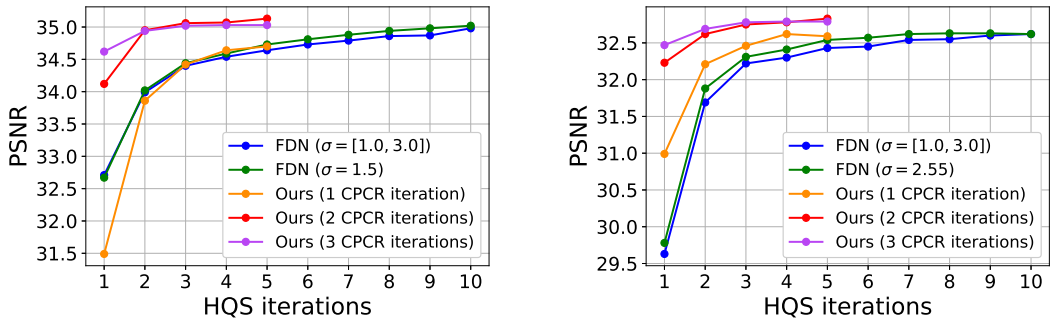


Figure 4-6: Performance of FDN [71] and LCHQS on the Levin [74] (left) and Sun [118] (right) datasets.

of [71], denoted as $\text{FDN}_{\text{T}}^{10}$ in their paper. Table 4.2 compares our method with these algorithms on two classical benchmarks. We use 5 HQS iterations and 2 CPCR iterations for CHQS and LCHQS. Except for EPLL that takes about 40 seconds to restore an image of the Levin dataset [74], all methods restore a 255×255 black and white image in about 0.2 second. The dataset of Sun *et al.* contains high-resolution images of size around 1000×700 pixels. EPLL removes the blur in 20 minutes on a CPU while the other methods, including ours, do it in about 1 second on a GPU. In this case, our learnable method gives comparable results to FDN [71], outputs globally much sharper results than EPLL [144] and is much faster. As expected, non-trainable CHQS is well behind its learned competitors (Tab. 4.2).

Number of iterations for LCHQS_G and CPCR. We investigate the influence of the number of HQS and CPCR iterations on the performance of LCHQS_G on the benchmarks of Levin *et al.* [74] and Sun *et al.* [118]. FDN implements 10 HQS iterations parameterized with CNNs but operates in the Fourier domain. Here, we compare LCHQS_G to the FDN model trained in a stage-wise manner (denoted as $\text{FDN}_{\text{G}}^{10}$ in [71]). Figure 4-6 plots the mean PSNR values for the datasets of Levin *et al.* and Sun *et al.* [118] after each stage. FDN comes in two versions: one trained on a single noise level (green line) and one trained on noise levels within a given interval (blue line). We use up to 5 iterations of our learnable CHQS scheme, but it essentially converges after only 3 steps. When the number of CPCR iterations is set to 1, FDN and our model achieve similar results for the same number of HQS iterations. For

2/3 CPCR iterations, we do better than FDN for the same number of HQS iterations by a margin of +0.4/0.5dB on both benchmarks. For 3 HQS iterations and more, LCHQS saturates but systematically achieves better results than 10 FDN iterations: +0.15dB for [74] and +0.26dB for [118].

Robustness to noise. Table 4.3 compares our methods for various noise levels on the 160 RGB images introduced previously, dubbed from "PASCAL benchmark". FDN corresponds to the model called $\text{FDN}_{\text{T}}^{10}$ in [71]. For this experiment, (L)CHQS uses 5 HQS iterations and 2 inner CPCR iterations. We add 1%, 3% and 5% Gaussian noise to these images to obtain three different test sets with gradually stronger noise levels. We train each model to deal with a specific noise level (non-blind setting) but also train a single model to handle multiple noise levels (blind setting) on images with 0.5 to 5% of white noise, as done in [71]. For each level in the non-blind setting, we are marginally above or below FDN results. In terms of average PSNR values, the margins are +0.12dB for 1%, +0.06dB for 3% and -0.05dB for 5% when comparing our models with FDN, but we are above the other competitors by margins between 0.3dB and 2dB. Compared to its noise-dependent version, the network trained in the blind setting yields a loss of 0.2dB for 1% noise, but gains of 0.14 and 0.27dB for 3 and 5% noises, showing its robustness and adaptability to various noises. Figure 4-7 compares results obtained on a blurry image with 3% noise.

4.4.4 Non-uniform motion blur removal

Typical non-uniform motion blur models assign to each pixel of a blurry image a local uniform kernel [22]. This is equivalent to replacing the uniform convolution in Eq. (4.13) by local convolutions for each overlapping patch in an image, as done by Sun *et al.* [116] when they adapt the solver of [144] to the non-uniform case. Note that FDN [71] and FCNN [136] operate in the Fourier domain and thus cannot be easily adapted to non-uniform deblurring, unlike (L)CHQS operating in the spatial domain. We handle non-uniform blur as follows to avoid computing different inverse filters at each pixel. As in [116], we model a non-uniform motion field with *locally* linear motions that can well approximate complex *global* motions such as camera

	1% noise	3% noise	5% noise	Time (s)
HQS-FFT	26.48	23.90	22.15	0.2
HQS-CG	26.45	23.91	22.27	13
EPLL [144]	28.83	24.00	22.10	130
FCNN [136]	29.27	25.07	23.53	0.5
FDN [71]	29.42	25.53	23.97	0.6
CHQS	27.08	23.33	22.38	0.3
LCHQS_G (non-blind)	**29.54 ± 0.02**	25.59 ± 0.03	23.87 ± 0.06	0.7
LCHQS_F (non-blind)	**29.53 ± 0.02**	25.56 ± 0.03	23.95 ± 0.05	0.7
LCHQS_G (blind)	29.22 ± 0.02	25.55 ± 0.03	24.05 ± 0.02	0.7
LCHQS_F (blind)	29.35 ± 0.01	**25.71 ± 0.02**	**24.21 ± 0.01**	0.7

Table 4.3: Uniform deblurring on 160 test images with 1%, 3% and 5% white noise. Running times are for an 500×375 RGB image. The mention "blind" (resp. "non-blind) indicates that a single model handles the three (resp. a specific) noise level(s).

	HQS-FFT	HQS-CG	EPLL [144]	CHQS	LCHQS_G	LCHQS_F
1% noise	23.49	25.84	25.49	25.11	26.83 ± 0.08	**26.98 ± 0.08**
3% noise	23.17	24.18	23.78	23.74	24.91 ± 0.05	**25.06 ± 0.06**
5% noise	22.44	23.10	23.34	22.65	23.97 ± 0.05	**24.14 ± 0.05**
Time (s)	13	212	420	0.8	0.9	0.9

Table 4.4: Non-uniform deblurring on 100 test images with 1%, 3% and 5% white noise. Running times are for an 500×375 RGB image.

rotations. We discretize the set of the linear motions by considering only those with translations (in pixels) in $\{1, 3 \ldots, 35\}$ and orientations in $\{0°, 6°, \ldots, 174°\}$. In this case, we know in advance all the 511 35×35 local blur kernels and compute their approximate inverses ahead of time. During inference, we simply determine which one best matches the local blur kernel and use its approximate inverse in CPCR. This is a parallelizable operation on a GPU. Table 4.4 compares our approach (in non-blind setting) to existing methods for locally-linear blur removal on a test set of 100 images from PASCAL dataset non-uniformly blurred with the code of [48] and with white noise. For instance for 1% noise, LCHQS_G scores +0.99dB higher than CG-based method, and LCHQS_F pushes the margin up to +1.13dB while being 200 times faster. Figure 4-8 shows one non-uniform example from the test set.

4.4.5 Deblurring with approximated blur kernels

In practice one does not have the ground-truth blur kernel but instead an *approximate* version of it, obtained with methods such as [95, 126]. We show that (L)CHQS works well for approximate and/or large filters, different from the ones used in the training set and without any training or fine-tuning.

We show in Figure 4-9 a deblurred image with an approximate kernel obtained with the code of [95] and of support of size 101×101 pixels. We obtain with LCHQS_F (blind) of Table 4.3 a sharper result than FCNN and do not introduce artifacts as FDN, showing the robustness of CPCR and its embedding in HQS to approximate blur kernels.

We also compare on two photographs taken with a handhelded Canon EOS 550D camera. In the first one, in Figure 4-10, the blurry image is obtained after cropping a 2400×1600 window. A uniform blur kernel is then estimated with [95]. The second one in Figure 4-11 is a full-frame image whose blur is estimated with the camera shake estimation technique of [126]. We compare LCHQS with USRNet [137], a state-of-the art hybrid method and the hyper-Laplacian prior technique of [70] which is purely model-based. Both [70] and [137] rely on FFT whereas we embed CPCR in our model. These two techniques introduce ringing artifacts next to edges such as the badge on the teddy bear's hat or on the books on the shelf whereas our approach does not, validating the choice of our approach on real-world scenarios.

4.5 Conclusion

We have presented a new learnable solver for non-blind deblurring. It is based on the HQS algorithm for solving penalized least-squares problems but uses preconditioned iterative fixed-point iterations for the x-update. Without learning, this approach is superior both in terms of speed and accuracy to classical solvers based on the Fourier transform and conjugate gradient descent. When the preconditioner and the proximal operator are learned, we obtain results that are competitive with or better than the state of the art. Our method is easily extended to non-uniform

deblurring, and it outperforms the state of the art by a significant margin in this case. We have also demonstrated its robustness to important amounts of white noise. Explicitly accounting for more realistic noise models [40] and other degradations such as downsampling is left for future work.

Figure 4-7: Example of image deblurring with an additive noise of 3% (better seen on a computer screen). In this example, we obtain better PSNR scores than competitors and better visual results, for example details around the door or the leaves.

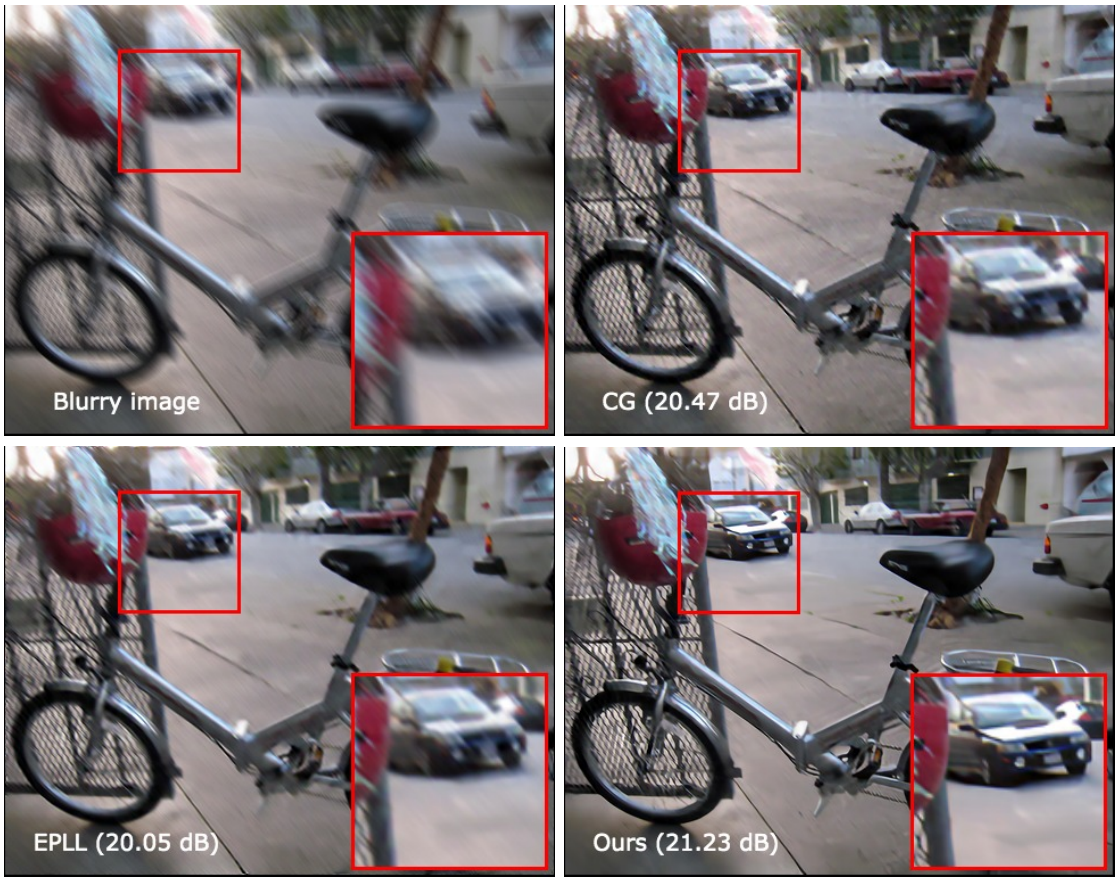


Figure 4-8: Non-uniform motion deblurring example with 1 % additive Gaussian noise (better seen on a computer screen). The car and the helmet are sharper with our method than in the images produced by our competitors.

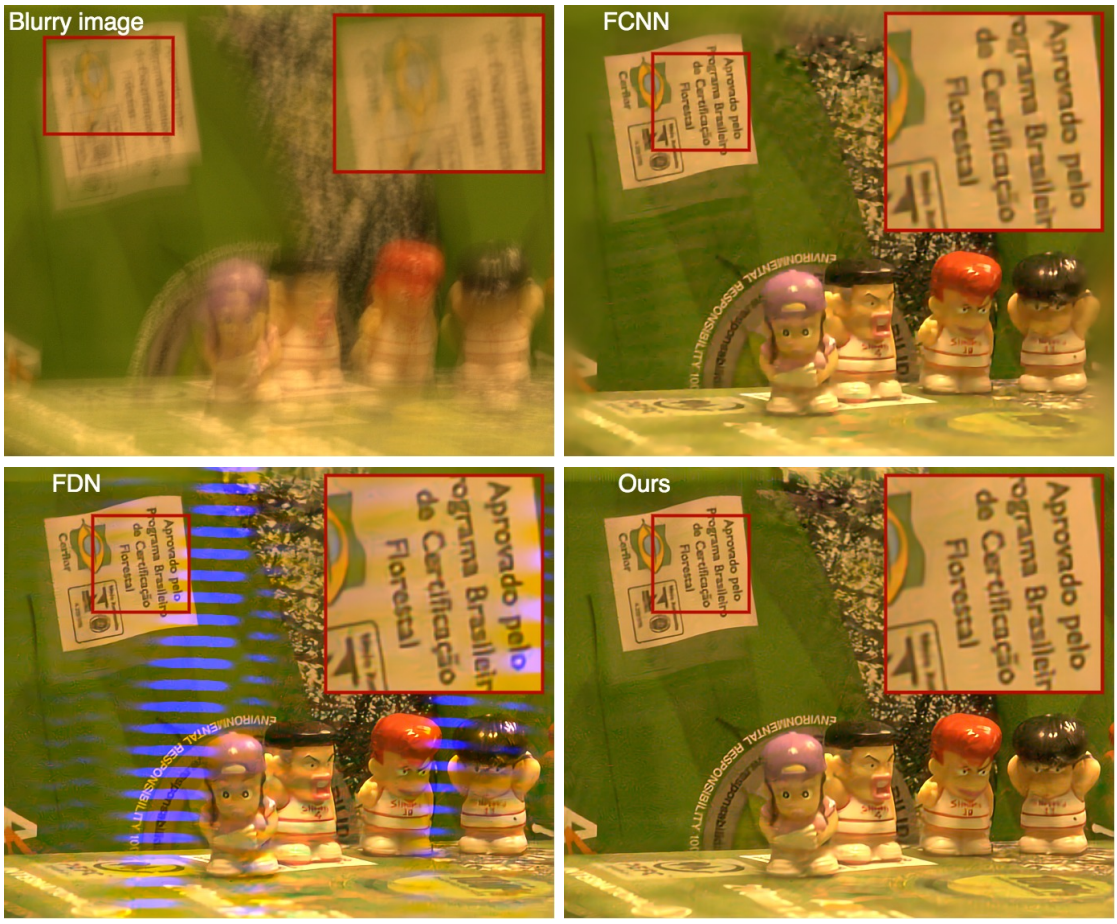


Figure 4-9: Real-world blurry images deblurred with an 101×101 blur kernel estimated with [95]. We can restore fine details with approximate, large kernels.

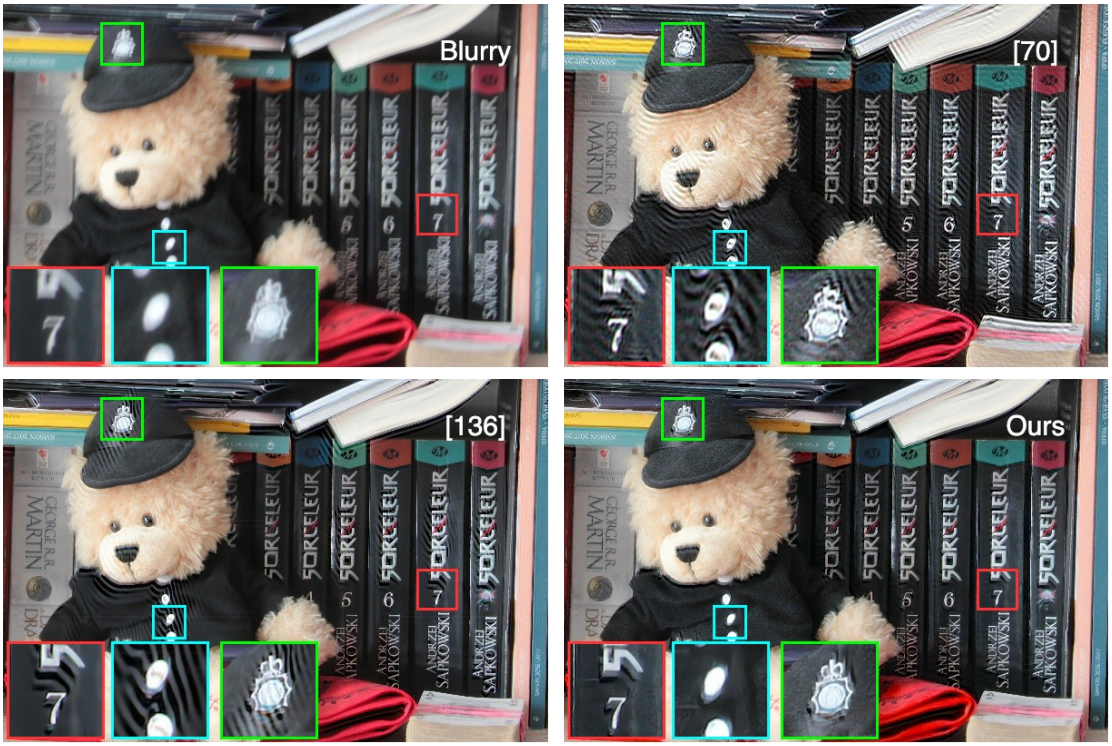


Figure 4-10: Real-world blurry images deblurred with camera shake estimated with the approach of Whyte *et al.* [126]. The proposed approach is robust to approximations in the predicted non-uniform blur like next to the buttons or the badge on the hat of the bear, whereas the competitors relying on FFT introduce ringing artifacts.

Figure 4-11: Real-world blurry images deblurred with camera shake estimated with the approach of Whyte *et al.* [126]. The proposed approach restores sharp edges next to the books cover without introducing ringing artifact nor amplifying the noise.

Chapter 5

Learning to Remove Optical Aberrations

Abstract

In this chapter, we address optical aberration removal mentioned in Chapter 2 or, in essence deblurring sharp images. Restoration is carried out by jointly deblurring and demosaicking noisy and raw images in a non-blind setting where the camera PSF is supposed known, either with camera calibration or estimated from blurry images as detailed in Chapter 3. We adapt an existing learning-based approach to RGB image deblurring to handle raw images by introducing a new interpretable module that jointly demosaicks and deblurs them. We train this model on RGB images converted into raw ones following a realistic invertible camera pipeline. We demonstrate the effectiveness of this model over two-stage approaches stacking demosaicking and deblurring modules on quantitive benchmarks and real photographs taken with different lenses and cameras.

5.1 Introduction

Our objective is to deblur, denoise and demosaick raw images. Raw data is important since it captures the most direct information we have about the observed scene, before any digital post-processing such as color transformations and gamma correction [18]. An important application is the removal of the optical aberrations introduced by the lens point-spread function (PSF). Indeed, any photograph, even perfectly focused and in the absence of any motion, contains some blur caused by its optics, ranging from

geometric distortions to chromatic aberrations [109, 135]. Removing these artifacts is a (little explored) instance of joint image demosaicking and non-blind deblurring addressed in this presentation.

Most approaches to image deblurring focus on sophisticated priors [70, 144], convolutional neural networks (CNNs) [87,123], or a combination of both [24,37,71,137]. Recent algorithms are robust to various noise levels [71], large [37, 137] and even approximate kernels [95], but they often ignore several stages of the camera pipeline connecting the analog image in the focal plane to the digital blurry image recorded by the camera.

As shown in Chapter 2, blur is caused by various, color-dependent optical phenomena [109, 135], camera and/or scene motion, and spatial, spectral and temporal integration over the pixel area. In particular, a single grey value is typically recorded at each pixel according to the Bayer pattern to form the final *raw* image.

Raw images are interpolated with filtering techniques [75, 84] or learning-based approaches [45, 69] into linear RGB (aka linRGB) images [18, 98]. This *demosaicking* operation is often highly non-linear. Sensor noise follows a statistical model whose parameters are estimated empirically from raw images [40] or learnt with a neural network on a corpus of image pairs [1], which is much more realistic than a Gaussian model. LinRGB images are finally converted into the standard RGB (aka sRGB) format through an image signal processing (ISP) pipeline [18].

This complex process suggests that the classical model of convolving a sharp RGB image with a linear filter to form its blurry version can be improved to better fit real digital cameras. We thus start from a raw, blurry and noisy image to predict a sharp and denoised linRGB image. We use a realistic image formation model, embed it into a penalized energy term, and unroll a few stages of an iterative solver within a parametric function inspired by [137] and trained with samples preprocessed with a modified variant of the linRGB-to-sRGB conversion pipeline from [18]. We finally apply this model to optical aberrations removal from images taken with high-end cameras whose PSF has been estimated separately.

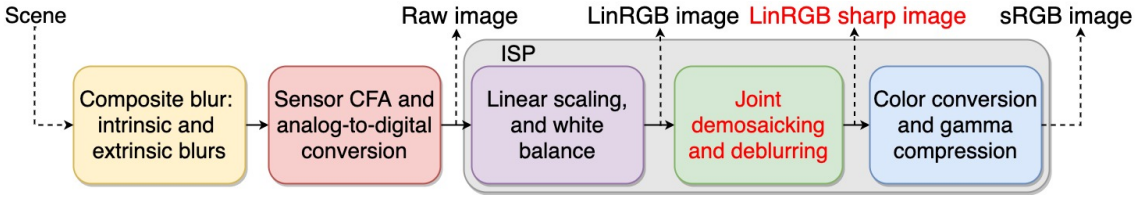


Figure 5-1: The modified camera pipeline we consider in this chapter. We replace a typical demosaicking algorithm with a joint demosaicking and deblurring technique whose role, in addition to predict the missing color components filtered out by the sensor CFA in the red block, is to remove the blur caused by the lens and/or the shooting conditions in the yellow block. Our module predicts a sharp image in the camera color space (or linRGB), further converted to a sRGB by the remaining operations on the ISP pipeline.

5.1.1 Contributions of this chapter

Our main contributions can be summarized as follows:

- We introduce a joint deblurring, demosaicking and denoisng forward model motivated by a realistic camera pipeline;
- the corresponding energy function is based on this forward model and optimized with a splitting method. We unroll a few stages of the splitting technique within a parametric function trained on blurry, mosaicked and noisy images;
- we present an experimental comparison with two-stage methods demonstrating the benefits of our approach and;
- we present an application of the proposed model to the removal of blur and chromatic aberrations caused by a camera's PSF on both synthetic and real images.

5.2 Image formation model

5.2.1 Camera pipeline overview

The overall image acquisition pipeline we consider in this chapter is represented in Figure 5-1 and upgrades the typical camera pipeline detailed in Chapter 2 and illus-

trated in Figure 2-1. A typical digital camera model starts with the sensor image, mosaicked by a colored filter array (CFA) whose pixel values corresponds to the focal image radiance, blurred by both intrinsic and extrinsic blurs. During the recording process, noise due to the camera hardware also degrades the image, resulting in a blurry, mosaicked and noisy sensor image. In this work, we upgrade the demosaicking module, initially converting the sensor raw image into an RGB version, to also remove the camera's intrinsic blur. We first review what happens before and after demosaicking in the camera ISP pipeline to target what are the input and the output of our optical aberration removal function.

Before demosaicking, black and white levels clip in the raw image to the pixels considered too dark, used in practice for noise estimation [40], and the pixel too bright, mainly because of saturation [27]. White balance for correcting the scene color palette and color-specific digital gains for compensating the CFA filtering behavior both linearly increase the pixel values of the sensor image. These transformations are channel-independent and thus do not mix the chromatic aberrations across different color channels. Demosaicking predicts the missing color components filtered out by the sensor CFA, and outputs an RGB image whose pixel values linearly corresponds to the focal image radiance, up to the few outliers corrected by black and white levels clipping. As a result, we call this demosaicked image the linear RGB image (linRGB in short) in the rest of this chapter.

After demosaicking, the ISP pipeline converts the in-camera linRGB image into the standard RGB format [114] and proceeds in several contrast and brightness non-linear corrections, the most famous one being gamma compression.

Classical demosaicking algorithms focus on predicting the missing color components in a noise-free image, *e.g.,* [75, 80, 84]. The physics of a camera suggests that there is always noise in a raw image, encouraging researcher to instead focus on joint demosaicking and denoising algorithms, *e.g.,* [45, 66, 91], hence replacing the single demosaicking stage in the classical ISP pipeline.

The proposed approach goes a step further since we remove optical aberrations from the raw image to predict a denoised linRGB image free of most of the camera's

intrinsic blurs. We follow Schuler *et al.* [109] and estimate such linRGB image by the means of a joint non-blind deblurring and demosaicking problem, which also takes into account noise. One could simply stack on top of a performing joint demosaicking and denoising algorithm a state-of-the-art deblurring model, such as USRNet [137], but in the context of optical aberration removal, Schuler *et al.* [109] show that a joint approach to demosaicking and deblurring yields better quantitative and qualitative results over two separate modules. In this chapter, we also deal with saturation, frequently encountered in many digital images and notorious to introduce artifacts during deblurring [27, 125]. We now formalize the inverse problem we seek to solve to obtain the clean linRGB discussed above.

5.2.2 Approximate forward model

We recall here the general approximate forward model of Eq. (3.1), central to this chapter:

$$y = s(MKx + \varepsilon) \sim \mathcal{N}(0, \lambda_{\mathrm{s}} MKx + \lambda_{\mathrm{r}}). \tag{5.1}$$

A digital sharp image x is first blurred by the composite blur, modelled by matrix K, and composed of intrinsic blur caused by the camera and lens and extrinsic blur due to shooting conditions. The blurry image is further mosaicked by the sensor CFA, represented by M, and further noised by the camera, represented by a random vector ε. Since the photosites have limited capacity, the maximal value a pixel can have is bounded. Saturation is represented with the non-linear function s.

Saturation happens in photographs containing bright items such as light bulbs or the Sun, but in most cases images do not have saturated areas. We can thus remove the saturation function s from the previous equation and consider the forward model:

$$y = MKx + \varepsilon \quad \text{with} \quad \varepsilon \sim \mathcal{N}(0, \lambda_{\mathrm{s}} MKx + \lambda_{\mathrm{r}}), \tag{5.2}$$

where, according to [18, 40], the vector ε is the sensor's noise seen on a raw or linRGB image can be modelled as a pixel-varying zero-mean Gaussian distribution with vari-

ance $\lambda_s MKx + \lambda_r$. It linearly depends on MKx and two scalars: a scaling weight λ_s and an offset λ_r representing respectively the impact of shot and read noise [40].

The output image y is the blurry and noisy raw image that has its color channels purged from the outlier pixel values with the black and white levels and multiplied by the white balance coefficients and the digital gains. The image y thus has correct in-camera linRGB values at the locations retained by the CFA, elsewhere it is zero. The input image x in this approximate model is a digital sharp linRGB image, mimicking the true analog focal image that has infinite resolution [9]. In practice we do not seek to increase the image resolution, only its sharpness and thus x and y have the same number of pixels. The matrix K corresponds to the convolution approximate digital filters representing the intrinsic camera blur, which is in reality an analog filtering operation. M represents the CFA. The joint deblurring and demosaicking setting is particularly adapted to optical aberration removal the properties of most recent lenses are accurately measured and tabulated[1] or can be estimated with calibration of a camera, *e.g.,* [10, 109], or with an optimization-based technique, *e.g.* [110, 119, 135]. This ensures that K is known for this task. The mosaicking pattern in M is a feature of the camera and can reasonably be supposed known too in general. We can now derive a hybrid method for solving joint deblurring and demosaicking.

5.3 Proposed approach

A natural approach for solving a joint deblurring, demosaicking and denoising problem is to leverage the important previous work on image demosaicking and denoising and non-blind RGB image deblurring by using a two-stage method stacking a joint demosaicking and denoising solver followed by a non-blind deblurring approach, *e.g.* [109], as shown in Figure 5-1. One of the main contributions of this work is to instead predict a sharp, demosaicked and denoised image from the observation y.

[1] https://www.dxomark.com/Lenses/

5.3.1 Energy function and splitting strategies

We integrate the forward model (5.2) into a penalized energy function with an image prior Ω whose solution is a denoised and deblurred linRGB image:

$$\min_x \frac{1}{2} \|y - MKx\|_F^2 + \lambda\Omega(x). \tag{5.3}$$

Optimization of (5.3) is traditionally carried out with splitting algorithms such as half-quadratic splitting (or HQS) [43]. We introduce an auxiliary variable z and solve

$$\min_{x,z} \frac{1}{2}\|y - MKz\|_F^2 + \lambda\Omega(x) \quad s.t. \quad z = x, \tag{5.4}$$

which becomes, when relaxed with a weight $\beta > 0$:

$$\min_{x,z} \frac{1}{2}\|y - MKz\|_F^2 + \frac{\beta}{2}\|z - x\|_F^2 + \lambda\Omega(x). \tag{5.5}$$

Optimization requires to jointly handle the operators M and K. We will detail the calculations in the next paragraphs.

Alternatively, we first demosaick the image, for instance with a CNN ξ with parameter ν [45], and second use a non-blind deblurring approach on the demosaicked linRGB image to predict the final sharp linRGB image x. The same relaxation of a constraint on an auxiliary variable z leads to

$$\begin{aligned} &\min_{x,z} \frac{1}{2}\|d - Kz\|_F^2 + \frac{\beta}{2}\|z - x\|_F^2 + \lambda\Omega(x), \\ &\text{with} \quad d = \xi_\nu(y, \lambda_\mathrm{r}, \lambda_\mathrm{s}). \end{aligned} \tag{5.6}$$

The demosaicking approach takes as input the noise parameters λ_r and λ_s, in the vein of [45, 69]. In this case, the reference image in the non-blind deblurring problem is d and not y.

5.3.2 Solving the intermediate problems

Predicting x in both Eqs (5.5) and (5.6) is done by solving

$$\min_x \lambda\Omega(x) + \frac{\beta}{2}\|z - x\|_F^2. \tag{5.7}$$

The minimizer of energy can be computed by evaluating in z the proximal operator ϕ of Ω with parameter λ/β [96]:

$$x = \text{prox}_\Omega(z, \lambda/\beta) = \phi(z, \lambda/\beta). \tag{5.8}$$

The intermediate deblurred image in (5.6) is the solution of:

$$\min_z \|d - Kz\|_F^2 + \beta\|z - x\|_F^2, \tag{5.9}$$

which is classically solved with fast Fourier transform (FFT) [137], conjugate gradient (CG) [55], or Richardson fixed-point iterations [37]. FFT is generally favored over CG since it scales with the size of an image. For instance a 3644×5734 RGB image can is deblurred in less than 2 minutes with [70], based on FFT, compared to 7 hours when FFT is replaced with CG in the work of Schuler *et. al.* [109].

However estimating z in Eq. (5.5) requires solving instead:

$$\min_z \|y - MKz\|_F^2 + \beta\|z - x\|_F^2, \tag{5.10}$$

Paliy *et. al.* [92] and Schuler *et. al.* [109] claim that M is not diagonalized in the Fourier basis, unlike K. This suggests that no fast solver exists for joint deblurring and demosaicking. Schuler *et. al.* say: "We believe that our approach is able to compete with state-of-the-art demosaicing algorithm because separating demosaicing and deblurring has the disadvantage that it does not require the result to be consistent with the image formation model. [...] Furthermore, typical demosaicing algorithms do not take chromatic aberration into account, which lead to a spatial separation of edge information across different color channels."

For any mosaicking pattern choice, M independently samples the color channels resulting in

$$MKz = \begin{bmatrix} D_R K_R & 0 & 0 \\ 0 & D_G K_G & 0 \\ 0 & 0 & D_B K_B \end{bmatrix} \begin{bmatrix} z_R \\ z_G \\ z_B \end{bmatrix} \tag{5.11}$$

where z_R, z_G and z_B are the red, green and blue components and K_R, K_G and K_B are the color-specific components of the blur K. One of our contributions is to estimate $z_R^\star$, $z_G^\star$ and $z_B^\star$, the red, green and blue images forming $z^\star$, the solution of (5.10), with a FFT-based module similar to the one of [137] in the case of the Bayer pattern.

5.3.3 FFT-based solver for least-squares (5.10)

The conclusions of Palyi *et al.* [92] suggests that no convolutional preconditioner can be computed for joint deblurring and deblurring, preventing us from using CPCR to solve the problem with respect to z as done in Chapter 4. We solve Eq. (5.10) instead with a FFT-based module inspired by [137] in the context of image upsampling. The linear operator MK in (5.11) decomposes the least-squares problem Eq. (5.10) into three independent terms. Figure 5-2 shows a mosaicked image y obtained sampled with M and whose y_R and y_B components are sampled versions of the corresponding RGB image where in each 2×2 non-overlapping patch, only one pixel value is retained per color. Similarly, the y_G is the sum of two images y_{G_1} and y_{G_2} such that $y_G = y_{G_1} + y_{G_2}$ (and $D_G = D_{G_1} + D_{G_2}$), each one sampling a single green pixel in the 2×2 non-overlapping patches in Fig. 5-2. We interpolate the missing values by solving, with c in $\{G_1, R, G_2, B\}$:

$$\min_{z_c} \|y_c - D_c K_c z_c\|_F^2 + \beta \|z_c - x_c\|_F^2. \tag{5.12}$$

These four problems are similar to the upsampling approach of [137] with rate 2 solving the linear system, with c in $\{G_1, R, G_2, B\}$:

$$(K_c^\top D_c^\top D_c K_c + \beta I) z_c^\star = K_c^\top D_c^\top y + \beta x_c. \tag{5.13}$$

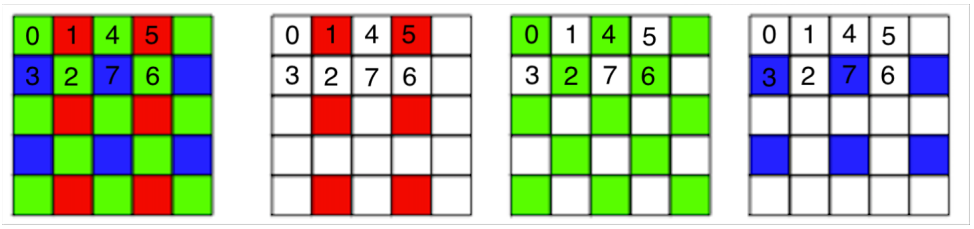


Figure 5-2: A mosaicked image with the Bayer pattern and its three sampled colored components.

By splitting the green image into y_{G_1} and y_{G_2}, we efficiently solve the four least-squares with an adapted version of the FFT-based approach of [137, 140]. We detail the implementation details and modifications on the code of [137] in the supplemental material. The images $z_R^\star$ and $z_B^\star$ are the solutions of (5.13) and $z_G^\star$ is obtained from $z_{G_1}^\star$ and $z_{G_2}^\star$ as follows: the pixels at locations $0, 4, \dots$ (resp. $2, 6, \dots$) in Fig. 5-2 are copied from the ones from $z_{G_1}^\star$ (resp. $z_{G_2}^\star$) and the remaining pixels at locations $1, 3, 5, \dots$ are the corresponding values in $(z_{G_1}^\star + z_{G_2}^\star)/2$. Stacking the three images $z_R^\star$, $z_G^\star$ and $z_B^\star$ yields the RGB solution $z^\star$ of Eq. (5.10).

5.3.4 Learnable embedding

Similar to Chapter 4, we improve the performance of the restoration method for solving either (5.6) or (5.5), we embed a few stages of HQS in the USRNet model of Zhang *et al.* [137] featuring two modules for learning the proximal step (5.7) and estimating on-the-fly the optimal weights $\beta^{(t)}$ and $\gamma^{(t)} = \lambda/\beta^{(t)}$ $(t = 1, \dots, T)$. We parameterize the proximal operator of ϕ with the same Unet model as in [137] with parameter θ such that for a given estimate $z^{(t)}$, we predict an RGB image x as

$$x^{(t+1)} = \phi_\theta(z^{(t)}, \gamma^{(t+1)}). \tag{5.14}$$

We predict $z^{(t+1)}$ from $x^{(t+1)}$, K, M and $\beta^{(t+1)}$ with the mapping ψ estimating its R, G and B components with the approach of Section 5.3.3:

$$z^{(t+1)} = \psi(x^{(t+1)}, K, M, \beta^{(t+1)}), \tag{5.15}$$

where $x^{(t+1)}$, K, M and $\beta^{(t+1)}$ are used to build the least-squares problems (5.13). The weights $\beta^{(t)}$ and $\gamma^{(t)}$ with a 3-layer perceptron, as detailed in [137] dubbed χ and with parameter ω. In our case, it becomes a function of the read and shot noise coefficients λ_{r} and λ_{s} defined as:

$$[\{\beta^{(t)}\}_{t=1}^{T}, \{\gamma^{(t)}\}_{t=1}^{T}] = \chi_{\omega}(\lambda_{\mathrm{r}}, \lambda_{\mathrm{s}}). \tag{5.16}$$

Our proposed approach for joint deblurring, demosaicking and denoising of raw images embeds the weight predictor χ_{ω}, the learnable proximal operator ϕ_{θ} and the FFT-based solver for joint deblurring and demosaicking ψ in (5.15) into the state-of-the-art model USRNet [137] initially designed for sRGB image non-blind deblurring and upsampling.

The two-stage approach first jointly demosaicks and denoises a raw image with a module ξ_{ν} that we implement with the learning-free approach of [84] or the state-of-the-art approach of [45] dubbed Deepjoint. It is followed by a non-blind deblurring module that we implement with the USRNet model of [137]. We modify Deepjoint and USRNet for processing the noise parameters λ_{s} and λ_{r} in place of the variance of a traditional Gaussian noise model. These two approaches are summarized in Algorithms 1.

5.4 Experiments

We run the experiments on an NVIDIA Tesla V100 graphic card. The code will be made available if the paper is accepted.

5.4.1 Experimental setting

We extract 96×96 patches in the training images of DIV2K and Flickr2K datasets, often used for training image upsampling models and featuring high-resolution edges, Moiré artifacts-prone textures and little compression artifacts. We convert these patches into the linRGB format with the pipeline of [18], blur them we with uniform

Algorithm 1: Parametric function for solving (5.5).

Data: y, K, M, λ_{r}, λ_{s}, θ, ω
// HQS weights prediction
$[\{\beta^{(t)}\}_{t=1}^{T}, \{\gamma^{(t)}\}_{t=1}^{T}] = \chi_{\omega}(\lambda_{\mathrm{r}}, \lambda_{\mathrm{s}})$;
// Initialization with naive demosaicking ξ, *e.g.*, bilinear interpolation
$x \leftarrow \xi(y)$;
$t \leftarrow 1$;
// Joint deblurring and demosaicking: the true observation y is the reference image in the fitting term.
for $t \leq T$ **do**
  $z \leftarrow \psi(y, x, K, M, \beta^{(t)})$;
  $x \leftarrow \phi_{\theta}(z, \gamma^{(t)})$;
  $t \leftarrow t + 1$;
end
Result: Restored linRGB image x.

blur from [137] and add affine noise generated with the code of [18]. We randomly flip and rotate the patches as augmentation. We extract 5000 patches from the 100 images of DIV2K validation set to form ours. Since blur is color-dependent [109, 110, 135], we synthesize RGB blur kernels (details in what follows) that are more realistic than grayscale filters such as the ones of [74]. We use Adam optimizer with initial learning rate set to 10^{-4}. The learning rate is divided by 2 whenever the validation loss plateaus during 15 epochs until reaching 10^{-6}. We use a batch size of 32. We use the ℓ_1 loss to compare the ground-truth patches and the predictions in the sRGB format as done in [18].

Color-specific blur kernels. We use the PSFs measured by Bauer *et. al.* [10] consisting in about 280,000 $111 \times 111 \times 3$ RGB linear filters, recorded by calibrating a Canon EOS 5DR camera with 20 different lenses set at different focal lengths and apertures pointing a grid made of 80×60 lights. These local PSFs thus represent the intrinsic blurs of a lens for certain focal length and a given aperture, resulting in a much more realistic blur kernel than simply deforming a grayscale filter as suggested in [20]. We use the regression technique discussed by [10] to generate on-the-fly unlimited local PSFs at any location on the camera's field-of-view. These filters are

cropped to a support of size $55 \times 55 \times 3$, which better fits the actual shape of the filter.

5.4.2 Joint deblurring and demosaicking evaluation

RGB motion blurs We evaluate our method on two synthetic datasets. We convert the 24 and 80 images of the Kodak [75] and Sun [118] datasets into linRGB images with the pipeline of [18] and blur them with the 8 filters of [74] convolved with a RGB filter from the PSFs of [10] to build RGB motion kernels. We add noise with the code of [18] with $\log(\lambda_s)$ chosen in $[10^4, 3 \times 10^{-3}]$ and corresponding λ_r in the model of [18], *i.e.*small to moderate noise, and mosaick them to form respectively 192 and 640 test samples. We compare our approach on raw images with two-stage methods. For demosaicking, we use either the filtering approach of [84] or the CNN for joint demosaicking and denoising of [45]. We retrain [45] to take into account the noise distribution of [18] on the 2.5 million patches of [45]. For non-blind deblurring, we use [70] based on a hyper-Laplacian image prior or the unrolled model of [137]. We retrain [137] each time on the images predicted by the first stage implemented with [45] or [84] to take into account prediction errors. As we predict linRGB images but ultimately want enhanced sRGB images, with compare three kinds of supervision: (i) the intermediate image d is converted into an sRGB image and [137] is supervised with sRGB targets (s/s), (ii) [137] deblurs a linRGB demosaicked image and is supervised with linRGB targets (lin/lin), and (iii) [137] deblurs a linRGB demosaicked image and is supervised with sRGB targets as in [18] (lin/s). We train a variant of our model in the "lin/lin" setting and three in the "lin/s" setting: one with initial guess demosaicked with simple bilinear interpolation, one with initial guess obtained with [45] (lin,s [45]) and one trained with grayscale kernels only (lin/s, gray). We unroll $T = 6$ iterations of HQS in the different implementations of [137] in Algorithm (1) implemented by the baselines and our approach.

Table 5.1 shows the PSNR and SSIM scores on the images after we crop 50 pixels on the borders to discard any boundary artifact in the measurements. Our method achieves on the four sets the best PSNR/SSIM score by PSNR margins of 0.5dB and

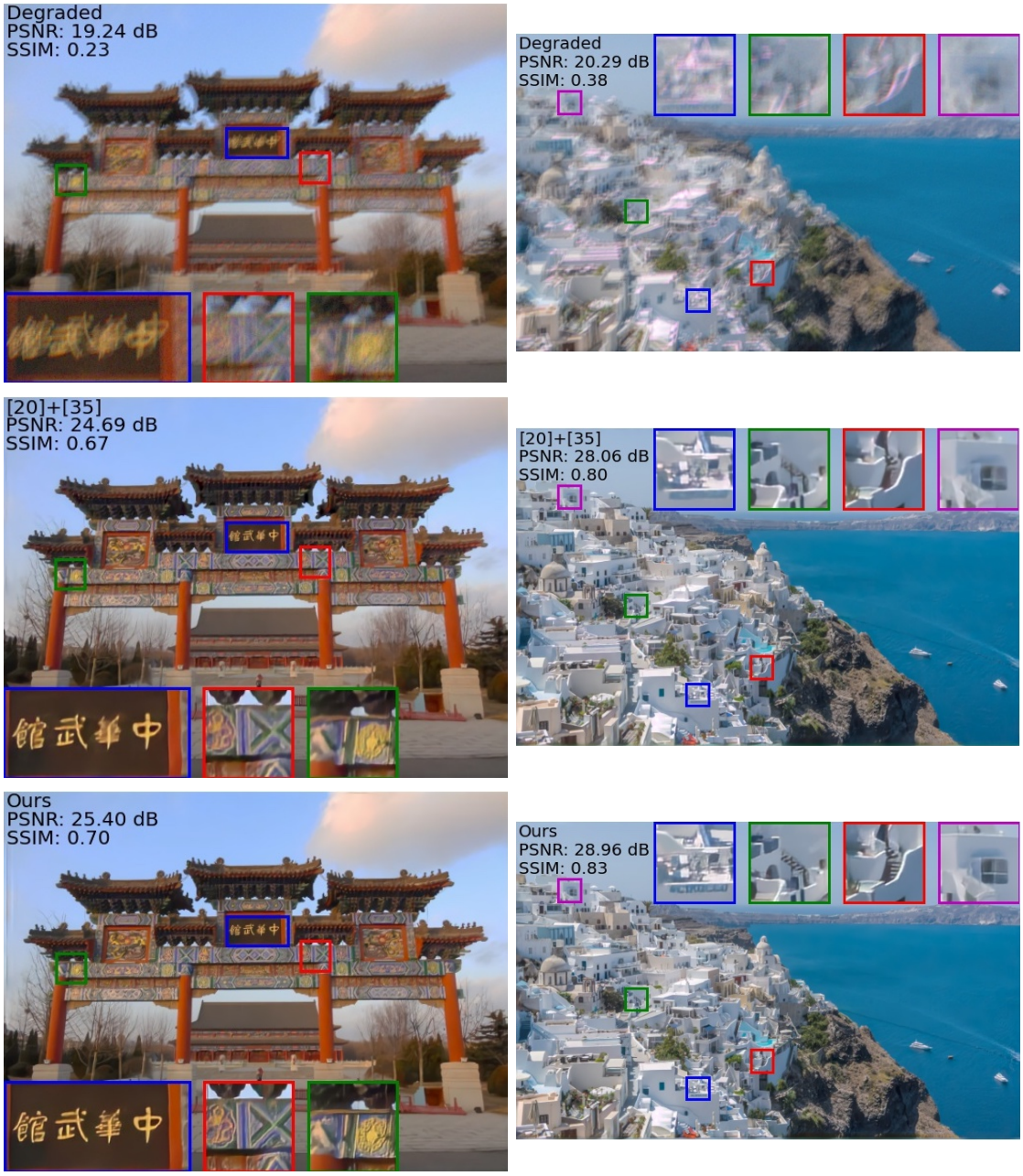


Figure 5-3: Examples of jointly deblurred, demosaicked and denoised images. We show the degraded raw images in the sRGB format. Compared to the two-stage method [84]+ [137], our method restores finer details.

Datasets	Kodak (192 images)				Sun (640 images)			
Noise level	Noiseless		With noise		Noiseless		With noise	
Color space	linRGB	sRGB	linRGB	sRGB	linRGB	sRGB	linRGB	sRGB
[84] + [70]	33.15/0.91	28.03/0.78	30.94/0.79	22.77/0.47	36.55/0.93	29.99/0.82	32.57/0.79	23.59/0.50
[84] + [137] (s/s)	- / -	32.16/0.88	- / -	29.56/0.78	- / -	33.50/0.91	- / -	30.40/0.80
[84] + [137] (lin/lin)	35.92/0.94	31.92/0.89	33.74/0.90	29.42/0.78	39.43/0.96	33.77/0.91	36.27/0.92	30.46/0.81
[84] + [137] (lin/s)	35.28/**0.95**	32.28/0.90	33.28/0.90	29.69/0.79	38.73/0.96	34.21/0.92	35.83/**0.92**	30.66/0.81
[45] + [70]	32.85/0.90	27.89/0.75	32.10/0.87	26.87/0.69	36.07/0.92	29.69/0.80	34.92/0.90	28.26/0.73
[45] + [137] (s/s)	- / -	30.27/0.83	- / -	29.01/0.77	- / -	31.73/0.86	- / -	30.07/0.80
[45] + [137] (lin/lin)	35.83/0.94	31.67/0.88	33.88/0.90	29.48/0.79	38.97/0.96	33.24/0.90	36.11/**0.92**	30.35/0.81
[45] + [137] (lin/s)	35.04/0.94	32.12/0.88	33.12/0.90	29.61/0.78	37.98/0.96	33.61/0.90	35.54/0.91	30.58/0.81
Ours (lin/lin)	**36.48**/**0.95**	32.46/0.90	**34.10**/**0.91**	29.76/**0.80**	**40.10**/**0.97**	34.39/0.93	**36.51**/**0.92**	30.72/**0.82**
Ours (lin/s)	35.72/**0.95**	32.99/0.91	33.52/**0.91**	29.98/**0.80**	39.13/**0.97**	34.90/0.93	35.93/**0.92**	30.86/**0.82**
Ours (lin/s, gray)	35.08/0.94	32.21/0.89	33.26/0.90	29.72/0.79	38.48/0.96	34.15/0.92	35.81/**0.92**	30.71/**0.82**
Ours (lin/s, [45])	35.86/**0.95**	**33.37**/**0.92**	33.53/**0.91**	**30.14**/**0.80**	39.21/**0.97**	**35.19**/**0.94**	35.87/**0.92**	**30.95**/**0.82**

Table 5.1: Joint deblurring, denoising and demosaicking comparison. Best result is in bold font. Second best is underlined.

SSIM margins of 0.01 or 0.02 over the two-stage methods. Methods trained with supervision on linRGB images naturally have the best scores on this color format but are behind the other methods in the sRGB format, suggesting supervision with sRGB sharp images with the approach of [18] is also beneficial for deblurring and demoisacking raw degraded images. Our variant trained only on grayscale kernels is in the ballpark of the best ones in the noisy cases but lags behind them by margins of 1dB in the noiseless case, meaning it cannot restore as fine details as the methods trained with the same blur distribution. The table also shows that initialization matters as the method with initial guess produced by [45] leads to better results, with margins ranging from 0.1 to 0.4dB on sRGB images, compared to the one initialized with a demosaicked image bilinearly interpolated. Figure 5-3 shows two restoration examples of blurry, mosaicked and noisy raw images (displayed as sRGB images) obtained with our best performing method and the best performer from the two-stage techniques. We also provide comparison with the same images but with the classical grayscale kernels of [74] in the supplemental material. The method is as fast as vanilla USRNet [137] since the only modification that might alter running time is the FFT-based module whose computation time is negligible compared to evaluating a CNN. It takes about 1 second to process a 720p image and at most 5 seconds for a 2K image.

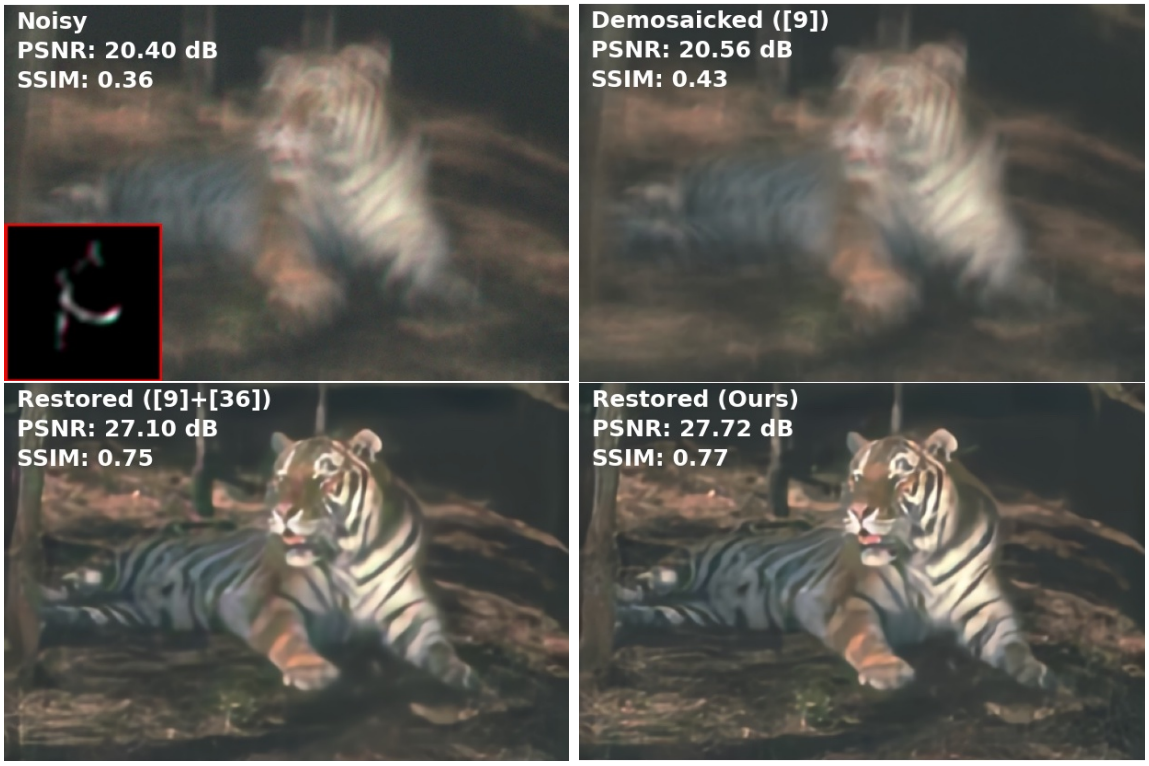


Figure 5-4: Restored exampled with a 65×65 blur kernel and noise parameters set to $\lambda_s = 10^{-3}$ and $\lambda_r = 1.3 \times 10^{-6}$. Better seen on a computer screen. Both quantitatively and visually our method outperforms the two-stage method [45]+ [137].

Robustness to larger kernels. We train the different models with 25×25 kernels but we show in Fig. 5-4 that our method can be used with much larger filters. The image is blurred with a 65×65 kernel from [95]. We compare the two-stage strategy [45]+ [137] to ours, both trained with the "lin/lin" setting. Our method achieves a better PSNR score and visual aspect compared to the two-stage method. This is typical of our observations on other large kernels from [95].

Synthetic optical aberration removal. We further validate our approach on synthetic images blurred with the realistic PSFs of Bauer *et al.* [10].

We build four synthetic datasets from benchmarks typically used for evaluating demosaicking methods. Indeed, our approach replaces a demosaicking algorithm in the camera pipeline of Figure 5-1 and thus must be efficient on images usually prone to artifacts such as Moiré. We use the 24 images of the Kodak dataset, the 18 images of the MacMaster (or McM) datasets and the 1000 images of the Moiré and HDRVDP

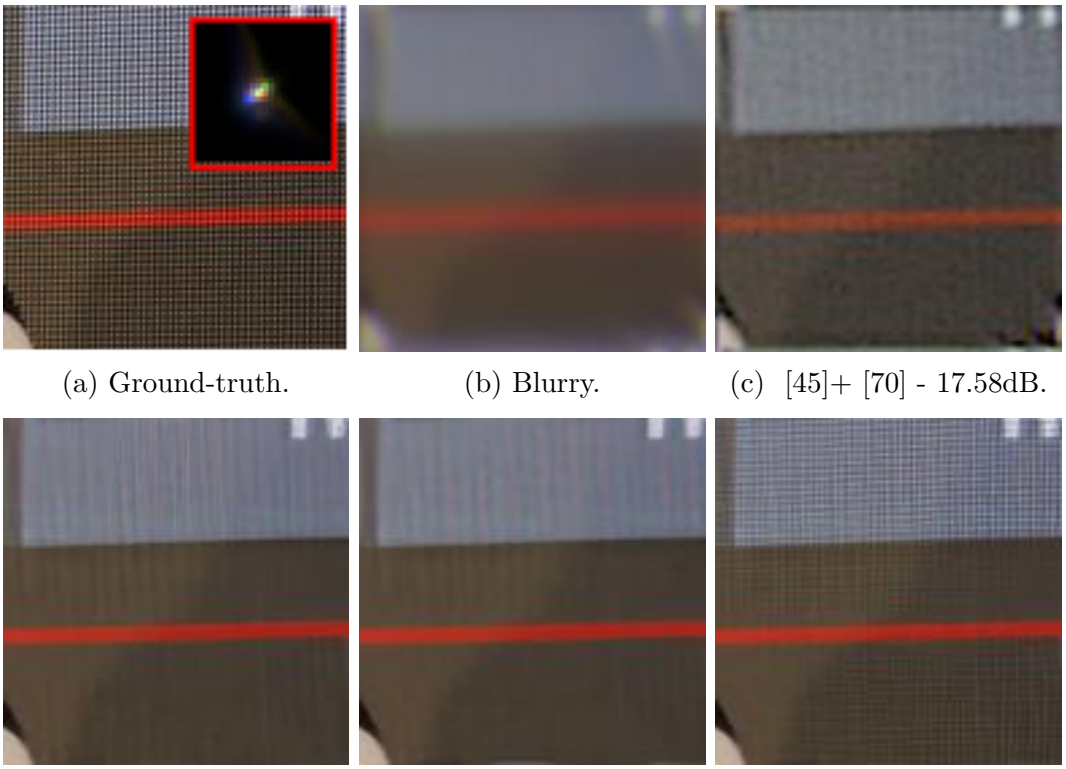
Datasets	Kodak	McM	Moiré	HDRVDP
[75] + [70]	26.89/0.81	26.81/0.82	24.02/0.73	21.47/0.69
[45] + [70]	27.19/0.82	27.00/0.82	24.32/0.70	21.51/0.69
[75] + [137]	<u>30.15</u>/<u>0.84</u>	<u>31.31</u>/<u>0.84</u>	<u>27.43</u>/**0.79**	**25.05**/**0.79**
[45] + [137]	29.56/0.84	31.00/0.83	27.28/<u>0.77</u>	<u>24.97</u>/<u>0.78</u>
Ours	**30.38**/**0.86**	**31.53**/**0.85**	**27.53**/**0.79**	**25.12**/**0.79**

Table 5.2: Quantitative results for optical aberration removal. The four benchmarks are composed of images prone to demosaicking artifacts such as Moiré. First is in bold, second is underlined.

datasets [45] to evaluate the different techniques. The sRGB images undergo an inverse ISP pipeline [18] where the in-camera RGB space is the one of the Canon EOS 5DS R camera. We blur these images with 9 different filters from the Nikon AF-S NIKKOR 50mm $f/1.4$G lens's PSF measurement in [10] and add a Poisson-Gaussian noise whose shot noise variance is set to 1e-4 and read noise variance to 1e-5, corresponding to moderate noise in practice. These transformations yield four benchmarks of respectively 216, 162, 9000 and 9000 blurry and noisy raw images.

Table 5.2 quantitatively compares the sRGB images produced by five approaches removing optical aberrations: Two two-stage approaches in the vein of [109] that concatenate demosaicking algorithms, naive bilinear interpolation [75] and a CNN [45] and followed by the non-blind deblurring approach of Krishnan and Fergus [70], two learning-based approaches whose first stages are bilinear interpolation [75] and demosaicnet [45], both followed by USRNet [137] each time trained with demosaicked blurry images, and the proposed joint deblurring and demosaicking approach. The two-stage model aggregating demosaicnet [45] and USRNet [137] first uses the pre-trained weights of [45] for initializing the demosaicking module and is trained to find an optimal parameter to [137]. The two models are later fine-tuned together, yielding the final method.

The table shows that the performance of our method exceeds the two-stage learning-based techniques by margins of +0.2dB for the two first datasets and is marginally above for the two last ones by +0.1dB and by more than 3dB for the methods using [70], especially for the Moiré and HDRVDP datasets. It suggests that our joint



(a) Ground-truth. (b) Blurry. (c) [45]+ [70] - 17.58dB.

(d) [75]+ [137] - 19.21dB. (e) [45]+ [137] - 19.16dB. (f) Ours - 20.34dB.

Figure 5-5: Example of joint deblurring and demosaicking results. The original sRGB image is prone to Moiré artifacts and is blurred with a realistic optical aberration filter, further corrupted by Poissian-Gaussian noise and mosaicked (we only show the intermediate blurry image here). The two-stage methods cannot reconstruct the finer details such as the grid whereas our approach can.

approach can restore finer details in the context of chromatic aberration removal than two-stage schemes, which is illustrated in Figure 5-5, showing a qualitative example of joint deblurring and demosaicking. The two-stage approaches fail to reconstruct the details of the grid, lost during blurring whereas our technique can. It is explained by the fact that our approach uses the original degraded signal as a starting point for restoration whereas the deblurring technique in the two-step schemes starts from a demosaicked image, containing predictions errors. This observarion encourages the use of a joint approach to restore most of the missing details in real images containing optical aberrations, as suggested by Schuler *et al.* [109].

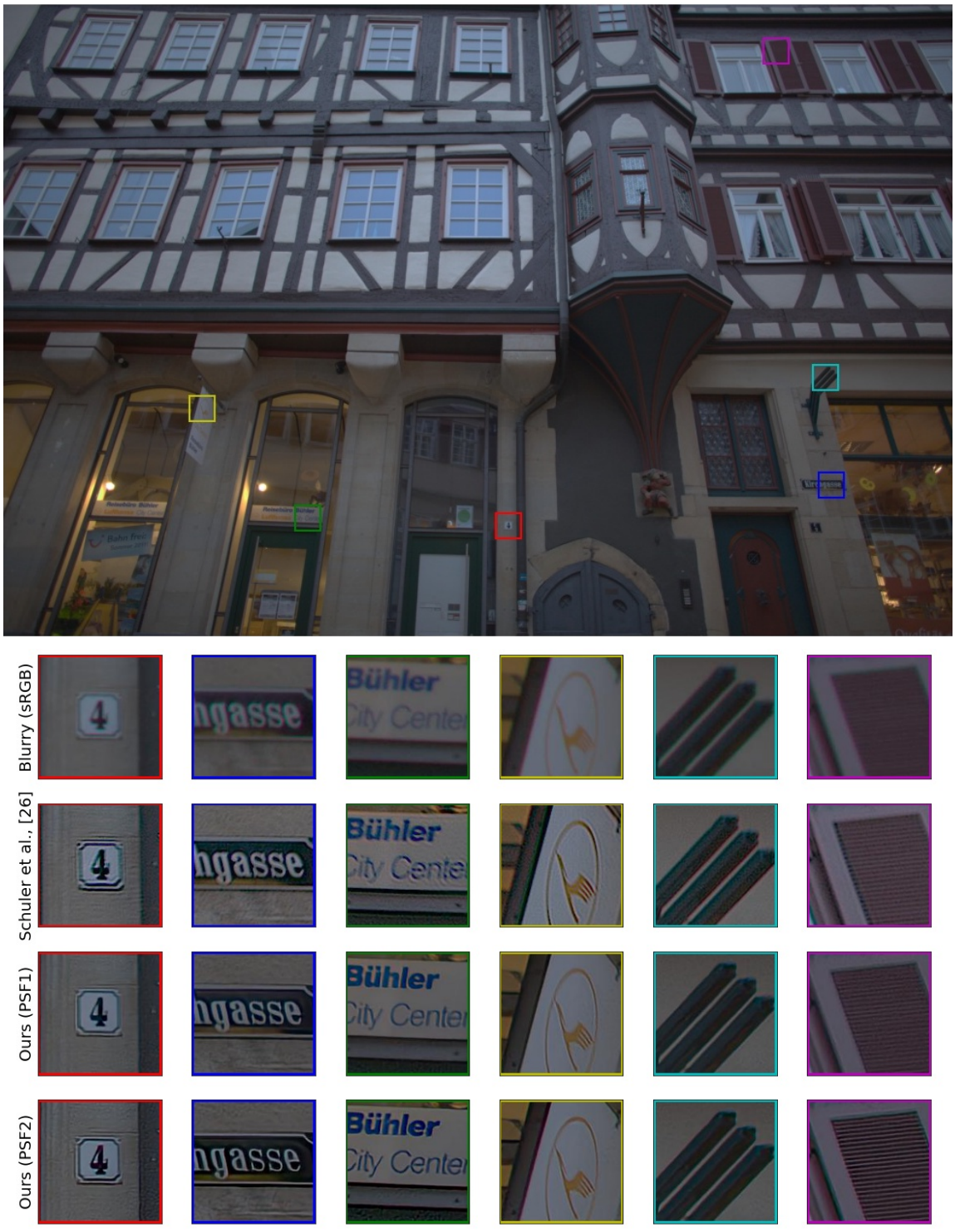


Figure 5-6: The real blurry image of [109] was taken with a modern digital SLR and a zoom lens at maximal aperture, exhibiting chromatic aberrations. Our method efficiently removes the blur caused by the optics, provided either a calibrated or an approximate PSF (best seen on a computer screen).

5.4.3 Optical aberration removal from real images

A practical application of the proposed model is automatic optical aberration removal from raw images taken with smartphones or DSLRs.

We compare our learnt model on raw images with the state-of-the art technique of Schuler *et al.* [109]. We restore a blurry image[2] shot with a Canon Mark II reflex camera and a canon 24mm f/1.4 lens at maximal aperture, whose PSF has been measured by two approaches[3]: a calibration method [109] and a variational approach [110].

We convert the corresponding sRGB real blurry image into a raw image with the camera pipeline of [18]. We follow [109,110] and break the full image into overlapping patches where the PSF boils down to a locally uniform blur kernel. We restore each patch with our model trained for the previous experiment without fine-tuning it with the PSF, stitch them together as detailed in [109] and convert the restored image back into the sRGB format. We show in Figure 5-6 the results for the PSF obtained with camera calibration (PSF1) and the one predicted with a variational method (PSF2). We compare them to the image restored in [109] that also removes blur from the raw image provided with the PSFs. Our methods can restore finer details such as the words on the panels or the closest in Figure 5-6, with both PSFs.

We also restore a 50-megapixel raw image shown in Figure 5-7, taken with a Canon EOS 5DS R camera and equipped with a Canon 35mm $f/1.4$L lens at aperture set to $f/2.8$[4]. We use the corresponding PSF recorded by Bauer *et al.* [10] for the same camera. Our technique processes the corresponding raw image in about 4 minutes, improving the sharpness and removing chromatic aberrations. Qualitative details are shown in Figure 5-8.

Optical aberration removal for saturated images. Saturation is known to introduce ringing artifacts in the context of motion blur removal [27,125] but is also

[2]https://webdav.tuebingen.mpg.de/pixel/lenscorrection/

[3]https://webdav.tuebingen.mpg.de/pixel/blind_lenscorrection/

[4]https://www.dpreview.com/articles/8808135799/1-4-and-more-canon-ef-35mm-f1-4l-ii-comparison

Figure 5-7: Real 5792×8688 image taken with a Canon EOS 5DS R camera with a Canon EF 35mm $f/1.4$ at aperture set to $f/2.8$. The original sRGB image lacks of sharpness and contains chromatic aberrations. Our method takes the corresponding raw image and the measurement of the PSF by Bauer *et al.* [10] as inputs and predicts a sharper image in 4 minutes on a GPU. Details are shown in Figure 5-8.

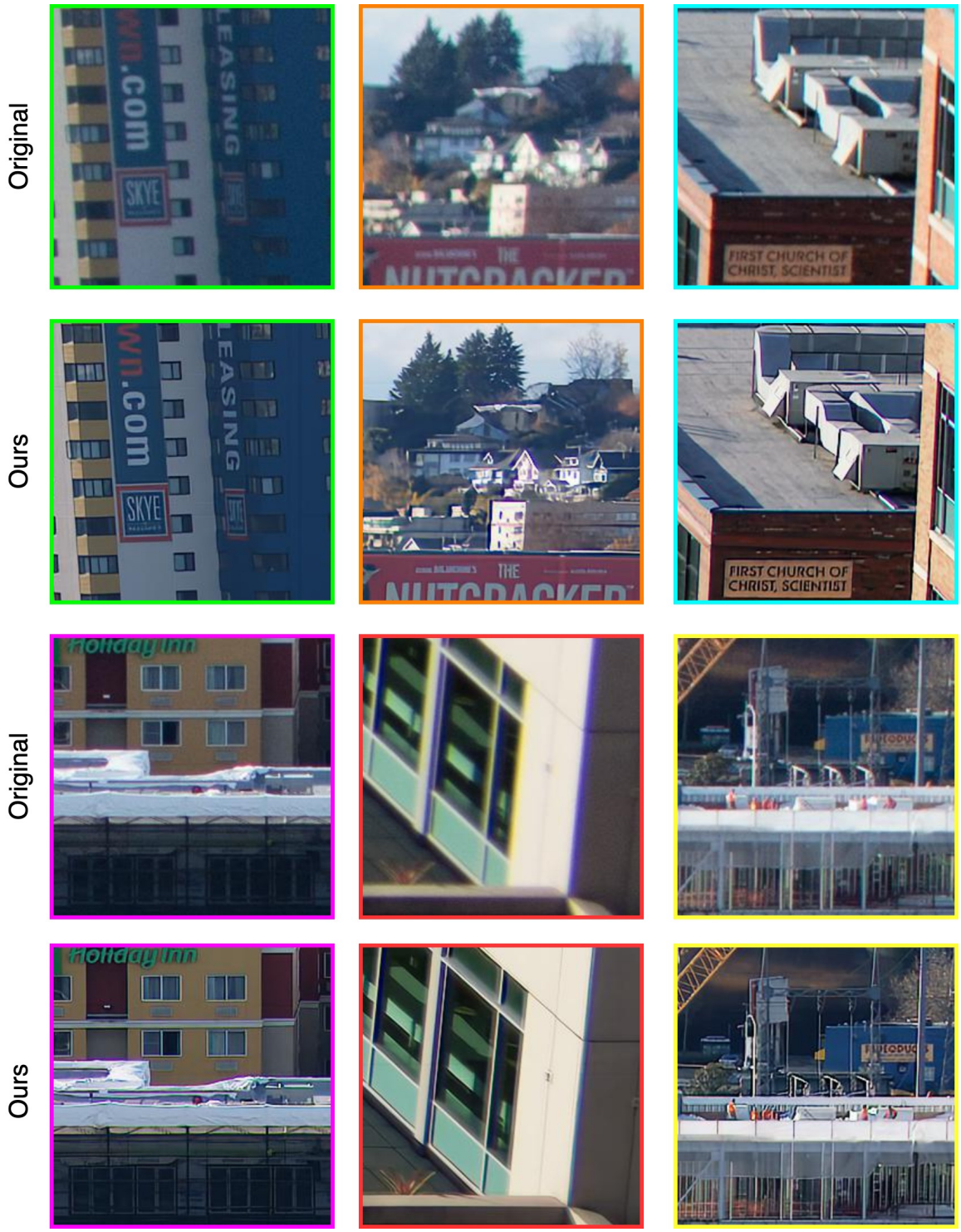
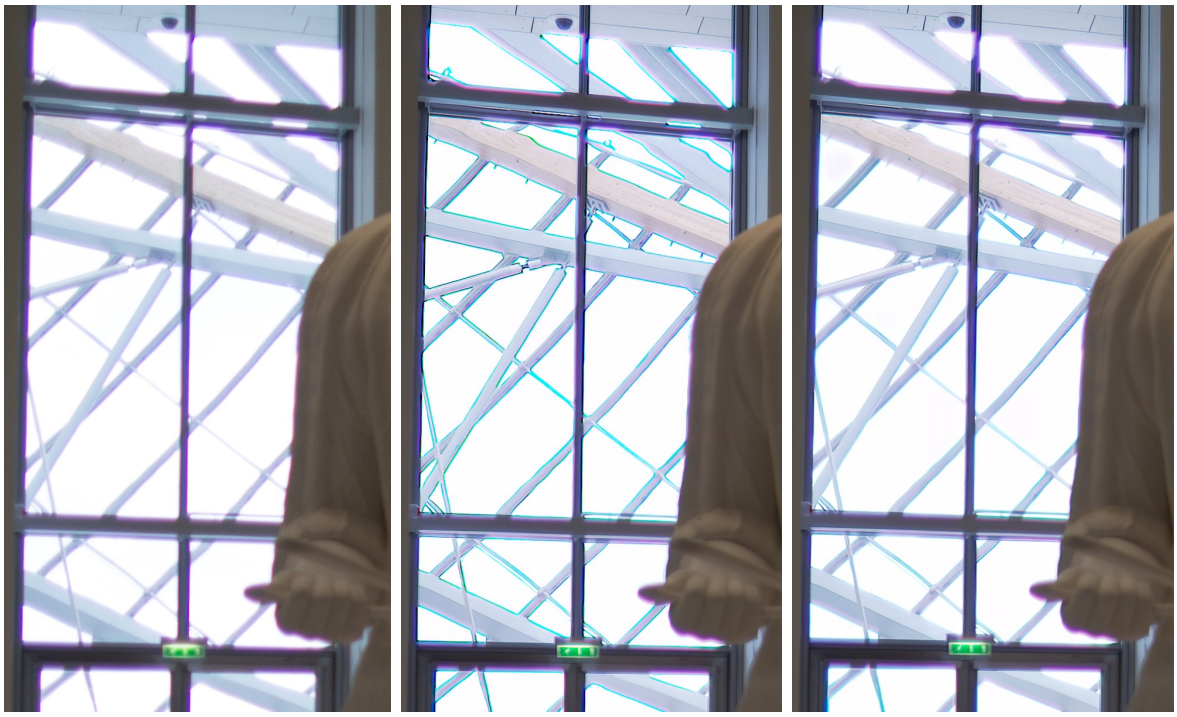


Figure 5-8: Comparisons between the original image and our restored version of the image shown in Figure 5-7. For instance the trees in the orange box or the building in the red box exhibit chromatic aberrations and the panels in the green box and the cyan box are blurred by the monochromatic aberrations. Our method removes these corruptions, resulting in a sharper image (better seen on a computer screen).

problematic for optical aberration removal. The image on the left in Figure 5-9 shows an image with saturated areas trough the window. Chromatic aberrations combined with saturation result in purple fringes [19] next to the saturated areas like the bars through the window.

We make our model more robust to artifacts due to saturation by including synthetic saturated images degraded by optical aberrations. We use the 396 curated high-dynamical range (HDR) images from [77] which are arrays stored in float32 format with entries corresponding to the scene radiance and possibly exceeding 1, the typical maximal value for digital images when converted to the float32 format. Directly using the scene radiance from an HDR image is useful, according to Debevec and Malik [32], for generating images looking like real saturated blurry images. These 396 images are convolved with the PSFs measured in Bauer *et al.* [10], added to a Poissonian-Gaussian random vector, clipped between 0 and 1, and finally mosaicked with the Bayer pattern, resulting in a raw noisy, blurry and saturated image. A sRGB ground-truth version is obtained by clipping the original radiance between 0 and 1, converting the colors to the sRGB format with a random color matrix obtained with the code of Brooks *et al.* [17] and gamma compression.

In Figure 5-9, our model trained only with non-saturated images from the DIV2K and Flickr2K datasets, despite sharpening details such as the ceiling, introduces noticeable green artifacts where the purple fringes were. A second version of our model trained on the same data as previously but combined with the HDR images and the same experimental protocol as before yields an image where most of the green artifacts have disappeared, except at some locations. This suggests that our training data permits to learn a proximal operator ϕ_θ that can efficiently handle saturated images, an in particular purple fringes. Yet, we do not explicitly include the clipping operator s from the forward model of Eq. (3.1) modelling saturation in the blurring process. We embed the approach of Mosleh *et al.* [86] to deal with saturation in our model but green artifacts remain in the image, even this upgrade, suggesting that optical aberration removal on saturated images demands special care. Previous work on non-blind saturated image deblurring, *e.g.,* [27, 125] have shown handling saturation



(a) Blurry. (b) Vanilla data. (c) Vanilla and HDR data.

Figure 5-9: The image on the right is a close-up of a real photograph taken with a Canon EOS 6D camera and a Canon EF 24mm $f/1.4$L USM lens at maximal aperture and known to introduce chromatic aberrations. Combined with the saturated regions trough the window, they yield purple fringes [19]. Our solution trained on the data without saturation introduces green silhouettes around the bars originally exhibiting colored fringes. The same model trained with additional saturated data (detailed in the text) restore the image and avoid most of the green artifacts.

in the data-fitting term is actually challenging and will be the topic of future work.

5.5 Conclusion

We have presented a approximate forward model for joint deblurring, demosaicking and denoising images, derived from a digital camera pipeline. We have proposed a penalized energy based on it, solved with HQS. Iterations of this method are embedded into a parametric function inspired by [137], restoring raw images and supervised with the technique of [18] and the real PSFs recorded by [10]. Our experiments have shown that it outperforms two-stage approaches, decomposing the problem into a demosaicking step followed by non-blind deblurring, quantitatively and visually and

in the presence of affine noise. Our approach have been applied to the removal of chromatic aberrations caused by the optics of a camera from real raw images, when the PSF is estimated beforehand. Future work includes the development of more robust methods for PSF estimation from a single raw image.

Chapter 6

Conclusions

We have presented in this thesis non-blind approaches to remove motion blur and optical aberrations from a single image. For doing so, two hybrid methods combining optimization-based and learning-based techniques in a single parametric function have been proposed.

The first one embeds in a parametric function convolutional preconditioned Richardson iteration to replace the classical FFT-based least-squares solvers used in most non-blind deblurring methods but known to introduce ringing artifacts. Our approach efficiently removes motion blur from both synthetic and real images for which an approximate blur has been estimated beforehand. The second one uses an approximate FFT-based solver for joint deblurring and demosaicking. The model is trained with images convolved with real PSFs and efficiently removes most of optical aberrations on real raw photographs taken with lenses for which we have a measurement of the corresponding PSF. Supervision is carried out on post-processed images after the ISP pipeline for better visual results, following [18]. A limitation of this approach is the presence of colored artifacts next to saturated areas.

Our work on optical aberration removal has shown that jointly inverting demosaicking and deblurring achieves results on par with than two-stage state-of-the-art methods. This suggests considering more degradations from the forward camera image formation model, such as saturation or defocus blur, would lead to superior results at the cost of designing efficient algorithm handling a corresponding least-squares

problem in a penalized energy. For instance, Whyte *et al.* [125] note that deblurring saturated images demands additional care to not propagate saturated pixels in the non-saturated areas, preventing the use of typical linear least-squares solvers for deconvolution such as FFT. One could imagine a hybrid approach based on a preconditioned iterative algorithm for handling saturation, comparable to the Richardson iterations for RGB image deblurring we have introduced in this thesis.

One of the main limitations of our optical aberration removal technique is the need to know or estimate the camera PSF beforehand, which can be obtained by measuring the aberrations in a test photograph a calibrated camera outputs for predefined depth, focal length and aperture parameters or by predicting a non-uniform blur kernel for each new image as the solution of an optimization problem. Both options may lead to inaccurate results and may be even harder than removing aberrations, in addition to be time-consuming for 50-megapixel images taken with recent DSLRs. An interesting research direction would be to build a parametric PSF function, akin to the approach of Joshi *et al.* [111], taking as input the focal length and the aperture and interpolating the corresponding optical aberrations, based on previous measurements for given lens settings, optics theory and possibly annotated data. The combination of such blur functions and our proposed joint deblurring and demosaicking model would result in a blind method. Such a blur model may also be useful for several computer vision and robotics applications for generating training data accounting for the in-camera degradation [20], thus narrowing the domain gap between synthetic and real images.

Blind optical aberration removal, even for saturated images, may also be possible by learning a black-box convolutional neural network with supervisory data, transferring the knowledge of a digital image formation model to the design of reasonable synthetic data. The HDR data and measured PSF by [10] used in Chapter 5 have led to promising results for this direction.

Bibliography

[1] Abdelrahman Abdelhamed, Marcus Brubaker, and Michael S. Brown. Noise flow: Noise modeling with conditional normalizing flows. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 3165–3173, 2019.

[2] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone cameras. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1692–1700, 2018.

[3] George B. Airy. On the diffraction of an object-glass with circular aperture. *Transactions of the Cambridge Philosophical Society*, pages 283–291, 1835.

[4] Raied Aljadaany, Dipan K. Pal, and Marios Savvides. Douglas-rachford networks: Learning both the image prior and data fidelity terms for blind image deconvolution. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 10235–10244, 2019.

[5] David Alleysson, Sabine Süsstrunk, and Jeanny Hérault. Linear demosaicing inspired by the human visual system. *IEEE Transactions on Image Processing (TIP)*, 14(4):439–449, 2005.

[6] Simon R. Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

[7] David A. Atchison and George Smith. *Optics of the Human Eye*. Butterworth-Heinemann, 2000.

[8] Lucio Azzari and Alessandro Foi. Variance stabilization in Poisson image deblurring. In *Proceedings of the Intelligence Symposium on Biomedical Imaging (ISBI)*, pages 728–731. IEEE, 2017.

[9] Simon Baker and Takeo Kanade. Limits on super-resolution and how to break them. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1167–1183, 2002.

[10] Matthias Bauer, Valentin Volchkov, Michael Hirsch, and Bernhard Schölkopf. Automatic estimation of modulation transfer functions. In *Proceedings of the International Conference on Image Processing (ICCP)*, pages 1–12. IEEE Computer Society, 2018.

[11] Bryce E. Bayer. Color imaging array. US patent 3971065, 1976.

[12] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Science*, 2(1):183–202, 2009.

[13] Stephen P. Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[14] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2014.

[15] William S. Boyle and George E. Smith. Charge coupled semiconductor devices. *The Bell System Technical Journal*, 49(4):587–593, 1970.

[16] Kristian Bredies, Karl Kunisch, and Thomas Pock. Total generalized variation. *SIAM Journal on Imaging Science*, 3(3):492–526, 2010.

[17] Tim Brooks and Jonathan T. Barron. Learning to synthesize motion blur. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 6840–6848, 2019.

[18] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned raw denoising. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11036–11045, 2019.

[19] Frédéric Cao, Frédéric Guichard, Hervé Hornung, and Cédric Sibade. Characterization and measurement of color fringing. In *Digital Photography*, volume 6817 of *SPIE Proceedings*, page 68170G. SPIE, 2008.

[20] Alexandra Carlson, Katherine A. Skinner, Ram Vasudevan, and Matthew Johnson-Roberson. Modeling camera effects to improve visual learning from synthetic data. In *ECCV Workshops*, pages 505–520, 2018.

[21] Ayan Chakrabarti. A neural approach to blind motion deblurring. In *Proceedings of the European Conference on Computer Vision*, pages 221–235, 2016.

[22] Ayan Chakrabarti, Todd E. Zickler, and William T. Freeman. Analyzing spatially-varying blur. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2512–2519, 2010.

[23] R. H. Chan, J. G. Nagy, and R. J. Plemmons. Circulant preconditioned toeplitz least squares iterations. *SIAM Journal on Matrix Analysis and Applications*, 15(1):80–97, 1994.

[24] Yunjin Chen and Thomas Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(6):1256–1272, 2017.

[25] Zhixiang Chi, Xiao Shu, and Xiaolin Wu. Joint demosaicking and blind deblurring using deep convolutional neural network. In *Proceedings of the International Conference on Image Processing (ICIP)*, pages 2169–2173. IEEE, 2019.

[26] Sunghyun Cho and Seungyong Lee. Fast motion deblurring. *ACM Transactions on Graphics (ToG)*, 28(5):145, 2009.

[27] Sunghyun Cho, Jue Wang, and Seungyong Lee. Handling outliers in non-blind image deconvolution. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 495–502. IEEE Computer Society, 2011.

[28] Philip E. Ciddor. Refractive index of air: new equations for the visible and near infrared. *OSA Applied Optics*, 35:1566–1573, 1996.

[29] Florent Couzinie-Devy, Julien Mairal, Francis R. Bach, and Jean Ponce. Dictionary learning for deblurring and digital zoom. *Technical report*, arXiv:1110.0957, 2011.

[30] Florent Couzinie-Devy, Jian Sun, Karteek Alahari, and Jean Ponce. Learning to estimate and remove non-uniform image blur. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1075–1082, 2013.

[31] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen O. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing (TIP)*, 16(8):2080–2095, 2007.

[32] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH*, pages 369–378. ACM, 1997.

[33] Mauricio Delbracio, Pablo Musé, Andrés Almansa, and Jean-Michel Morel. The non-parametric sub-pixel local point spread function estimation is a well posed problem. *International Journal Computer Vision (IJCV)*, 96(2):175–194, 2012.

[34] Jiangxin Dong, Stefan Roth, and Bernt Schiele. Deep wiener deconvolution: Wiener meets deep learning for image deblurring. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[35] Brandon Dube, Roger Cicala, Aaron Closz, and Jannick P. Rolland. How good is your lens? assessing performance with MTF full-field displays. *OSA Applied Optics*, 56(20):5661–5667, 2017.

[36] Thomas Eboli, Alex Nowak-Vila, Jian Sun, Francis R. Bach, Jean Ponce, and Alessandro Rudi. Structured and localized image restoration. *Technical report*, arXiv:2006.09261, 2020.

[37] Thomas Eboli, Jian Sun, and Jean Ponce. End-to-end interpretable learning of non-blind image deblurring. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 314–331, 2020.

[38] Michael Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer Publishing Company, Incorporated, 2010.

[39] Robert Fergus, Barun Singh, Aaron Hertzmann, Sam T. Roweis, and William T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics (ToG)*, 25(3):787–794, 2006.

[40] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen O. Egiazarian. Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data. *IEEE Transactions on Image Processing (TIP)*, 17(10):1737–1754, 2008.

[41] Gerald Budge Folland. *Fourier Analysis and its Applications.* Wadsworth, 1992.

[42] Eric R. Fossum and Donald B. Hondongwa. A review of the pinned photodiode for CCD and CMOS image sensors. *IEEE Journal of the Electron Devices Society*, 2(3):33–43, 2014.

[43] Donald Geman and George Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 14(3):367–383, 1992.

[44] Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions in Image Processing*, 4(7):932–946, 1995.

[45] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics (ToG)*, 35(6):191:1–191:12, 2016.

[46] Gene H. Golub and Charles F. Van Loan. *Matrix computations (4th edition).* John Hopkins University Press, 2013.

[47] D. Gong, Z. Zhang, Q. Shi, A. van den Hengel, C. Shen, and Y. Zhang. Learning deep gradient descent optimization for image deconvolution. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2020.

[48] Dong Gong, Jie Yang, Lingqiao Liu, Yanning Zhang, Ian D. Reid, Chunhua Shen, Anton van den Hengel, and Qinfeng Shi. From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 3806–3815, 2017.

[49] Joseph Goodman. *Introduction to Fourier optics.* McGraw-Hill, 1996.

[50] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the International Conference on Machine Learning ICML*, pages 399–406, 2010.

[51] Ankit Gupta, Neel Joshi, C. Lawrence Zitnick, Michael F. Cohen, and Brian Curless. Single image deblurring using motion density functions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 171–184, 2010.

[52] Richard. I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[53] Samuel W. Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T. Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 35(6), 2016.

[54] Felix Heide, Mushfiqur Rouf, Matthias B. Hullin, Björn Labitzke, Wolfgang Heidrich, and Andreas Kolb. High-quality computational imaging through simple lenses. *ACM Transactions on Graphics (ToG)*, 32(5):149:1–149:14, 2013.

[55] Felix Heide, Markus Steinberger, Yun-Ta Tsai, Mushfiqur Rouf, Dawid Pajäk, Dikpal Reddy, Orazio Gallo, Jing Liu, Wolfgang Heidrich, Karen Egiazarian, Jan Kautz, and Kari Pulli. FlexISP: A flexible camera image processing framework. *ACM Transactions on Graphics (ToG)*, 33(6):231:1–231:13, 2014.

[56] Michael Hirsch and Bernhard Schölkopf. Self-calibration of optical lenses. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 612–620. IEEE Computer Society, 2015.

[57] Michael Hirsch, Suvrit Sra, Bernhard Schölkopf, and Stefan Harmeling. Efficient filter flow for space-variant multiframe blind deconvolution. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 607–614, 2010.

[58] Zhe Hu and Ming-Hsuan Yang. Learning good regions to deblur images. *International Journal on Computer Vision (IJCV)*, 115(3):345–362, 2015.

[59] Zhe Hu, Lu Yuan, Stephen Lin, and Ming-Hsuan Yang. Image deblurring using smartphone inertial sensors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1855–1864, 2016.

[60] Neel Joshi, Sing Bing Kang, C. Lawrence Zitnick, and Richard Szeliski. Image deblurring using inertial measurement sensors. *ACM Transaction on Graphics (ToG)*, 29(4):30:1–30:9, 2010.

[61] Neel Joshi, Rick Szeliski, and D.J. Kriegman. PSF estimation using sharp edge prediction. In *Proceedings of the conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[62] Sing Bing Kang. Automatic removal of chromatic aberration from a single image. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[63] Eric Kee, Sylvain Paris, Simon Chen, and Jue Wang. Modeling and removing spatially-varying optical blur. In *Proceedings of the International Conference on Computational Photography (ICCP)*, pages 1–8, 2011.

[64] Tim Kelley. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, 1995.

[65] Tae Hyun Kim and Kyoung Mu Lee. Segmentation-free dynamic scene deblurring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2766–2773, 2014.

[66] Teresa Klatzer, Kerstin Hammernik, Patrick Knöbelreiter, and Thomas Pock. Learning joint demosaicing and denoising based on sequential energy minimization. In *Proceedings of the International Conference on Computational Photography (ICCP)*, pages 1–11, 2016.

[67] Erich Kobler, Teresa Klatzer, Kerstin Hammernik, and Thomas Pock. Variational networks: Connecting variational methods and deep learning. In *Proceedings of the German Conference on Pattern Recognition*, pages 281–293, 2017.

[68] Rolf Köhler, Michael Hirsch, Betty J. Mohler, Bernhard Schölkopf, and Stefan Harmeling. Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. In *Proceedings of the European Conference on Computer Vision*, pages 27–40, 2012.

[69] Filippos Kokkinos and Stamatios Lefkimmiatis. Iterative joint image demosaicking and denoising using a residual denoising network. *IEEE Transactions on Image Processing (TIP)*, 28(8):4177–4188, 2019.

[70] Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-Laplacian priors. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1033–1041, 2009.

[71] Jakob Kruse, Carsten Rother, and Uwe Schmidt. Learning to push the limits of efficient FFT-based image deconvolution. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 4596–4604, 2017.

[72] Karl Kunisch and Thomas Pock. A bilevel optimization approach for parameter learning in variational models. *SIAM Journal on Imaging Science*, 6(2):938–983, 2013.

[73] Anat Levin, Robert Fergus, Frédo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics (ToG)*, 26(3):70, 2007.

[74] Anat Levin, Yair Weiss, Frédo Durand, and William T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1964–1971, 2009.

[75] Xin Li, Bahadir Gunturk, and Lei Zhang. Image demosaicing: a systematic survey. In *Visual Communications and Image Processing*, volume 6822, pages 489 – 503, 2008.

[76] Chih-Hung Liang, Yu-An Chen, Yueh-Cheng Liu, and Winston H. Hsu. Raw image deblurring. *Technical report*, arXiv:2012.04264, 2020.

[77] Yu-Lun Liu, Wei-Sheng Lai, Yu-Sheng Chen, Yi-Lung Kao, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. Single-image HDR reconstruction by learning to reverse the camera pipeline. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1648–1657, 2020.

[78] Leon B. Lucy. An iterative technique for the rectification of observed distributions. *Astronomical Journal*, 79(6):745–754, 1974.

[79] Hiêp Quang Luong, Bart Goossens, Jan Aelterman, Aleksandra Pizurica, and Wilfried Philips. A primal-dual algorithm for joint demosaicking and deconvolution. In *Proceedings of the International Conference on Image Processing ICIP*, pages 2801–2804. IEEE, 2012.

[80] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Non-local sparse models for image restoration. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2272–2279, 2009.

[81] Markku J. Mäkitalo and Alessandro Foi. Optimal inversion of the anscombe transformation in low-count poisson image denoising. *IEEE Transactions on Image Processing (TIP)*, 20(1):99–109, 2011.

[82] Stéphane Mallat. *A Wavelet Tour of Signal Processing, 2nd Edition*. Academic Press, 1999.

[83] Henrique S. Malvar, Li-wei He, and Ross Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 485–488. IEEE, 2004.

[84] Daniele Menon, Stefano Andriani, and Giancarlo Calvagno. Demosaicing with directional filtering and a posteriori decision. *IEEE Transactions on Image Processing (TIP)*, 16(1):132–141, 2007.

[85] Tomer Michaeli and Michal Irani. Blind deblurring using internal patch recurrence. In *Proceedings of the European Conference on Computer Vision*, pages 783–798, 2014.

[86] Ali Mosleh, Yasser Elmi Sola, Farzad Zargari, Emmanuel Onzon, and J. M. Pierre Langlois. Explicit ringing removal in image deblurring. *IEEE Transactions on Image Processing (TIP)*, 27(2):580–593, 2018.

[87] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 257–265, 2017.

[88] Nhat Nguyen, Peyman Milanfar, and Gene H. Golub. A computationally efficient superresolution image reconstruction algorithm. *IEEE Transactions on Image Processing (TIP)*, 10(4):573–583, 2001.

[89] Mehdi Noroozi, Paramanand Chandramouli, and Paolo Favaro. Motion deblurring in the wild. In *Proceedings of the German Conference on Pattern Recognition GCPR*, volume 10496, pages 65–77. Springer, 2017.

[90] Stanley Osher and Leonid I. Rudin. Feature-oriented image enhancement using shock filters. *SIAM Journal on Numerical Analysis*, 27(4):919—-940, 1990.

[91] Dmitriy Paliy, Vladimir Katkovnik, Radu Ciprian Bilcu, Sakari Alenius, and Karen O. Egiazarian. Spatially adaptive color filter array interpolation for noiseless and noisy data. *International Journal on Imaging Systems and Technology (IJIST)*, 17(3):105–122, 2007.

[92] Dmytro Paliy, Alessandro Foi, Radu Ciprian Bilcu, Vladimir Katkovnik, and Karen O. Egiazarian. Joint deblurring and demosaicing of poissonian bayer-data based on local adaptivity. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE, 2008.

[93] Russ Palum. Anti-aliasingf filter analysis for digital cameras. In *Proceedings of the International Congress of Imaging Science (ICIS)*, pages 25–28, 2006.

[94] Jinshan Pan, Zhe Hu, Zhixun Su, and Ming-Hsuan Yang. Deblurring text images via l0-regularized intensity and gradient prior. In *CVPR*, pages 2901–2908. IEEE Computer Society, 2014.

[95] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang. Deblurring images via dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(10):2315–2328, 2018.

[96] Neal Parikh and Stephen P. Boyd. Proximal algorithms. *Foundations and Trends on Optimization*, 1(3):127–239, 2014.

[97] Bruce H. Pillman. Impact of CCD size, pixel pitch, and anti-aliasing filter design on sharpness of digital camera print. In *Image Processing, Image Quality, Image Capture, Systems Conference (PICS)*, pages 216–220, 2000.

[98] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2750–2759, 2017.

[99] Michael Potmesil and Indranil Chakravarty. Synthetic image generation with a lens and aperture camera model. *ACM Transactions on Graphics (ToG)*, 1(2):85–108, 1982.

[100] Michael Potmesil and Indranil Chakravarty. Modeling motion blur in computer-generated images. In *SIGGRAPH*, pages 389–399. ACM, 1983.

[101] Erik Reinhard, Greg Ward, Sumanta N. Pattanaik, Paul E. Debevec, and Wolfgang Heidrich. *High Dynamic Range Imaging - Acquisition, Display, and Image-Based Lighting, 2nd edition).* Academic Press, 2010.

[102] William Hadley Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62(1):55–59, 1972.

[103] Max J. Riedl. *Optical Design Fundamentals for Infrared Systems.* SPIE Press, 2001.

[104] Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (RED). *SIAM Journal on Imaging Science*, 10(4):1804–1844, 2017.

[105] Stefan Roth and Michael J. Black. Fields of experts. *International Journal on Computer Vision*, 82(2), 2009.

[106] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[107] Uwe Schmidt and Stefan Roth. Shrinkage fields for effective image restoration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2774–2784, 2014.

[108] Christian J. Schuler, Harold Christopher Burger, Stefan Harmeling, and Bernhard Schölkopf. A machine learning approach for non-blind image deconvolution. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1067–1074, 2013.

[109] Christian J. Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Non-stationary correction of optical aberrations. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 659–666, 2011.

[110] Christian J. Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Blind correction of optical aberrations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 187–200, 2012.

[111] Yichang Shih, Brian Guenter, and Neel Joshi. Image enhancement using calibrated lens simulations. In *Proceedings of the European Conference on Computer Vision*, pages 42–56, 2012.

[112] Ferréol Soulez and Éric Thiébaut. Joint deconvolution and demosaicing. In *Proceedings of the International Conference on Image Processing (ICIP)*, pages 145–148, 2009.

[113] Jean-Luc Starck and Fionn Murtagh. *Astronomical Image and Data Analysis, Second Edition*. Astronomy and Astrophysics Library. Springer, 2006.

[114] Michael Stokes, Matthew Anderson, Srinivasan Chandrasekar, and Ricardo Motta. A standard default color space for the internet - sRGB. `https://www.w3.org/Graphics/Color/sRGB.html`, 1996.

[115] Rob C. Sumner. Processing RAW images in MATLAB. *Department of Electrical Engineering, University of California Santa Cruz*, 2014.

[116] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 769–777, 2015.

[117] Jian Sun, Zongben Xu, and Heung-Yeung Shum. Image super-resolution using gradient profile prior. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[118] Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Edge-based blur kernel estimation using patch priors. In *Proceedings of the International Conference on Computational Photography (ICCP)*, pages 1–8, 2013.

[119] Tiancheng Sun, Yifan Peng, and Wolfgang Heidrich. Revisiting cross-channel information transfer for chromatic aberration correction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 3268–3276. IEEE Computer Society, 2017.

[120] Yu-Wing Tai, Ping Tan, and Michael S. Brown. Richardson-Lucy deblurring for scenes under a projective motion path. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1603–1618, 2011.

[121] Hiroyuki Takeda, Sina Farsiu, and Peyman Milanfar. Deblurring using regularized locally adaptive kernel regression. *IEEE Transactions on Image Processing*, 17(4):550–563, 2008.

[122] Huixuan Tang and Kiriakos N. Kutulakos. What does an aberrated photo tell us about the lens and the scene? In *ICCP*, pages 1–10. IEEE Computer Society, 2013.

[123] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8174–8182, 2018.

[124] Yilun Wang, Junfeng Yang, Wotao Yin, and Yin Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Science*, 1(3):248–272, 2008.

[125] Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *International Journal of Compututer Vision (IJCV)*, 110(2):185–201, 2014.

[126] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. *International Journal on Computer Vision (IJCV)*, 98(2):168–186, 2012.

[127] Norbert Wiener. *The Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. John Wiley & Sons, Inc., 1949.

[128] Bartlomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. Handheld multi-frame super-resolution. *ACM Transactions on Graphics (ToG)*, 38(4):28:1–28:18, 2019.

[129] Li Xu and Jiaya Jia. Two-phase kernel estimation for robust motion deblurring. In *Proceedings of the European Conference on Computer Vision ECCV*, volume 6311, pages 157–170, 2010.

[130] Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia. Image smoothing via L_0 gradient minimization. *ACM Transactions on Graphics*, 30(6):174, 2011.

[131] Li Xu, Jimmy S. J. Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *Advances in Neural Information Processing Systems*, pages 1790–1798, 2014.

[132] Li Xu, Xin Tao, and Jiaya Jia. Inverse kernels for fast spatial deconvolution. In *Proceedings of the European Conference on Computer Vision*, pages 33–48, 2014.

[133] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural L_0 sparse representation for natural image deblurring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1107–1114, 2013.

[134] Li Xu, Shicheng Zheng, and Jiaya Jia. Unnatural L0 sparse representation for natural image deblurring. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1107–1114, 2013.

[135] Tao Yue, Jin-Li Suo, Jue Wang, Xun Cao, and Qionghai Dai. Blind optical aberration correction by exploring geometric and visual priors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1684–1692, 2015.

[136] Jiawei Zhang, Jinshan Pan, Wei-Sheng Lai, Rynson W. H. Lau, and Ming-Hsuan Yang. Learning fully convolutional networks for iterative non-blind deconvolution. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 6969–6977, 2017.

[137] Kai Zhang, Luc Van Gool, and Radu Timofte. Deep unfolding network for image super-resolution. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3214–3223, 2020.

[138] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep CNN denoiser prior for image restoration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition CVPR*, pages 2808–2817. IEEE Computer Society, 2017.

[139] Xuaner Zhang, Qifeng Chen, Ren Ng, and Vladlen Koltun. Zoom to learn, learn to zoom. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3762–3770. Computer Vision Foundation / IEEE, 2019.

[140] Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouame, and Jean-Yves Tourneret. Fast single image super-resolution using a new analytical solution for ℓ_2-ℓ_2 problems. *IEEE Transactions on Image Processing (TIP)*, 25(8):3683–3697, 2016.

[141] Song Chun Zhu and David Mumford. Learning generic prior models for visual computation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 463–469. IEEE Computer Society, 1997.

[142] Song Chun Zhu and David Mumford. Prior learning and Gibbs reaction-diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 19(11):1236–1250, 1997.

[143] Xiang Zhu, Scott Cohen, Stephen Schiller, and Peyman Milanfar. Estimating spatially varying defocus blur from a single image. *IEEE Transactions on Image Processing (TIP)*, 22(12):4879–4891, 2013.

[144] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 479–486, 2011.

Chapter A

Inverse Filter Computation

Computing the filters $c_0, \ldots, c_n$ in closed-form

In this section, we explain with more details how we compute the filters $c_0, \ldots, c_n$ in practice. We seek to minimize the ridge regression problem

$$\begin{aligned} C &= \underset{C}{\operatorname{argmin}} \, ||\delta - C \star \widehat{K}||_F^2 + \rho \sum_{i=0}^{n} ||c_i||_F^2, && \text{(A.1)} \\ &= \underset{c_0,\ldots,c_n}{\operatorname{argmin}} \, ||\delta - \sum_{i=0}^{n} c_i \star k_i||_F^2 + \rho \sum_{i=0}^{n} ||c_i||_F^2, && \text{(A.2)} \\ &= \underset{c_0,\ldots,c_n}{\operatorname{argmin}} \, ||\delta - \sum_{i=0}^{n} k_i \star c_i||_F^2 + \rho \sum_{i=0}^{n} ||c_i||_F^2. && \text{(A.3)} \end{aligned}$$

By introducing $\widehat{c} = (\widehat{c}_0^\top, \ldots, \widehat{c}_n^\top)^\top$, we solve

$$\widehat{c} = \underset{c_0,\ldots,c_n}{\operatorname{argmin}} \, ||\widehat{\delta} - \sum_{i=0}^{n} K_i \widehat{c}_i||_F^2 + \rho \sum_{i=0}^{n} ||\widehat{c}_i||_F^2. \quad \text{(A.4)}$$

With $Q = [K_0^\top, \ldots, K_n^\top]$, the solution if this ridge regression problem is

$$\widehat{c} = Q^\top (QQ^\top + \rho \mathrm{Id})^{-1} \widehat{\delta}, \quad \text{(A.5)}$$

where $Q^\dagger$ is the right pseudo-inverse of Q. Therefore each $\widehat{c}_i$ is given by

$$\widehat{c}_i = K_i(QQ^\top + \rho \mathrm{Id})^{-1}\widehat{\delta}, \quad \text{for } i = 0, \ldots, n. \tag{A.6}$$

The main computational cost is the inversion of the $w^2 \times w^2$ matrix $QQ^\top$, which is the much smaller than the $(n+1)^2 w_c^2 \times (n+1)^2 w_c^2$ matrix involved in the left pseudo-inverse of Q.

In all the experiments, we generate the filters $c_0, \ldots, c_n$ with 0.01 as regularizer weight ρ.

RÉSUMÉ

La netteté est un critère important lorsque l'on souhaite prendre de bonnes photographies. Plusieurs facteurs tels que les paramètres de l'appareil photo, mouvement et defocus peuvent réduire la netteté d'une photographie et causer du flou qui détruit des details de l'image. Ce contenu peut être retrouvé par le défloutage, un problème inverse mal posé qui utilise la photographie floue, et le flou si connu, pour estimer une version nette. Les techniques de traitement d'image habituelles reposent sur l'optimisation utilisant des priors empiriques sur les images ou des approches d'apprentissage automatique exploitant des paires d'images nettes et floue en guise de supervision. Dans cette thèse, nous suivons une tendance récente visant à combiner les deux types de techniques présentés précédemment et produisant l'état de l'art pour le défloutage. Nous présentons d'abord une fonction paramétriques pour déflouter des images RVB, en incorporant des itérations de point-fixe de Richardson préconditionnées pour remplacer la classique transformée de Fourier rapide (TFR), sujette aux artefacts d'ondulation, en particulier aux bords d'une image. Dans une seconde contribution, nous proposons un modèle pour le dématriçage et le défloutage non-aveugle simultanés pour les images "raw".

MOTS CLÉS

Traitement d'images, défloutage d'images, problème inverse, apprentissage profond.

ABSTRACT

Sharpness is an important criterion for shooting acceptable photographs. Several factors such as the camera settings, motion or defocus may decrease image sharpness and lead to blur, resulting in the loss of details. Typical image processing techniques rely either on optimization algorithms using a handcrafted image prior or machine learning approaches leveraging supervisory pairs of sharp and synthetic blurry images. In this thesis, we follow a recent trend that combines the two sorts of techniques previously detailed and achieving the state-of-the-art image deblurring results. We first present a parametric function for RGB image deblurring, embedding preconditioned Richardson fixed-point iterations to replace the classical fast Fourier transform (FFT) algorithm prone to ringing artifacts, especially at the image boundaries. In a second contribution, we propose a model for joint non-blind deblurring and demosaicking of raw images.

KEYWORDS

Image processing, image deblurring, inverse problem, deep learning.