POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
DOTTORATO DI RICERCA IN INGEGNERIA DELL'INFORMAZIONE

Motion Blur: Analysis and Restoration

Tesi di dottorato di:
Giacomo Boracchi

Relatore:
Prof. Vincenzo Caglioti
Tutore:
Prof. Marco Colombetti
Coordinatore del programma di dottorato:
Prof. Patrizio Colaneri

XX ciclo

POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
Piazza Leonardo da Vinci 32 I 20133 — Milano

POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
DOTTORATO DI RICERCA IN INGEGNERIA DELL'INFORMAZIONE

Motion Blur: Analysis and Restoration

Doctoral Dissertation of:
Giacomo Boracchi

Advisor:
Prof. Vincenzo Caglioti
Tutor:
Prof. Marco Colombetti
Supervisor of the Doctoral Program:
Prof. Patrizio Colaneri

XX - 2008

Dedication

Acknowledgements

Vorrei ringraziare Diana che mi è sempre stata pazientemente accanto in questi anni di dottorato. Un ringraziamento anche ai miei genitori e a mio fratello per il loro aiuto e supporto.

Devo molto a tutto il gruppo di Tampere che mi ha offerto l'opportunità di fare ricerca e mi ha fatto appassionare a questo mestiere, ringrazio il professor Vladimir Katkovnik ed il Professore Karen Egiazarian. In particolare ringrazio Alessandro Foi per tutto quello che mi ha trasmesso e per i suoi consigli, spero un giorno di poter essere anche io altrettanto di aiuto per chi ne avrà bisogno.

Un grazie al mio relatore, il professor Vincenzo Caglioti, che mi ha saputo propormi innovativi argomenti di ricerca per il mio dottorato.

Un ringraziamento ai miei colleghi di dottorato ed in particolare a Simone, Pierluigi ed Alessandro, per tutto quello che abbiamo fatto assieme e anche per quello che siamo solo riusciti ad abbozzare.

Giacomo Boracchi

Things which we see are not by themselves what we see
It remains completely unknown to us what the objects may be by themselves
and apart from the receptivity of our senses.
We know nothing but our manner of perceiving them

I. Kant

Abstract

Motion blur is a phenomenon which is corrupting images when, any motion occurs between the camera viewpoint and the captured scene during the acquisition. Rarely this can be described with a shift invariant operator although this is a common assumption in the literature.

In a motion blurred image, the Point Spread Function (PSF) of each pixel is determined by the relative motion between the camera viewpoint and the imaged scene point. Therefore the PSF of each pixel may typically vary according to the camera motion and the depths of the imaged scene points. Both the blur analysis (estimation) and the image restoration, become much more challenging issues in case of a shift variant blur operator, than in case of a shift invariant blur operator. As a matter of fact, only few works in literature have considered the shift variant blur.

This thesis concerns the analysis and the restoration of single blurred images when the blur is due to a specific camera motions. In particular the focus is on the blur produced by a camera translation. We show that assuming shift variant blur allows us to describe the degradation process more accurately. We derive two descriptions of the degradation process due to camera translation and camera rotation, where the blur is modeled as shift variant and parametric operators.

The thesis is divided into two parts. The first part deals with local blur estimation, and presents algorithms devised for estimating blur direction and extent in small image regions containing a corner. The proposed algorithms estimate blur parameters in corner regions where other blur parameters estimation methods typically fail. We devised also a procedure for detecting blurred corners and adaptively select a region where to perform blur estimation.

In the second part of the thesis we consider the blurred image as a whole and we address two different issues: the estimation of the camera motion and the image restoration. This part is mostly dedicated to images corrupted by blur due to a pure camera translation. We prove that, although this situation has been always treated assuming the blur shift invariant, the blur becomes shift variant as the camera translation has an essential component perpendicular to the image plane. We devise a single image algorithm for estimating both the camera 3D motion direction and the PSF parameters in every image pixel. We

also introduce a restoration algorithm for these kind of images (radial blurred images), which is based on two steps: the blur inversion and the noise removal. The blur is inverted exploiting polar to Cartesian coordinate transformations. We study how the coordinate transformations and the blur inversion affect the noise in order to use a non-linear spatially adaptive filter, the Pointwise Shape-Adaptive DCT to exploit the image structures and attenuate noise and artifacts. Since in radially blurred images, the PSF extent at any image pixel can be related to the depth of the corresponding point in the scene, we also investigate and discuss the capabilities of estimating the scene depth from a single motion blurred image.
The blur produced by a camera rotation is also considered in the second part of the thesis. We devise an algorithm for estimating the 3D rotation axis of a camera by analyzing a single blurred image. Contrary to the existing methods, we treat the more general case where the rotation axis is not necessarily orthogonal to the image plane, taking into account the perspective effects that affect the smears.

All the proposed algorithms have been tested on synthetically blurred images as well as camera images.

Riassunto

Il *Motion Blur* nelle immagini è causato da un movimento della camera, della scena o di parte di questa durante il tempo di esposizione. Il motion blur comporta la perdita di dettagli delle immagini, con un conseguente degrado della qualità visiva: la ricostruzione di queste immagini è di grande interesse applicativo ed è un problema ancora lontano dall'essere definitivamente risolto. Un problema strettamente collegato alla ricostruzione dell'immagine è la stima del moto della camera e delle caratteristiche del blur, sempre a partire da una sola immagine mossa. Questa tesi presenta algoritmi per estrarre dell'informazione da una singola immagine motion blurred, qualora il blur fosse dovuto ad un particolare moto tra la camera e la scena. Con il termine informazione facciamo riferimento alle caratteristiche del blur, ai parametri del moto della camera e all'immagine "ideale" che si otterrebbe in assenza di movimento e con una corretta esposizione.

In questa tesi motion blur è stato descritto da operatori definiti dal moto 3D della camera. Questa modellazione ad-hoc ha permesso di ricavare algoritmi innovativi per stimare il blur e per ricostruire l'immagine, a partire da una sola immagine mossa. In particolare abbiamo trattato il blur prodotto da due tipi di moto della camera la traslazione e la rotazione della camera. Questa scelta è stata fatta per frequenza con la quale queste situazioni occorrono e per la facilità di collegare il blur al moto della camera.

Il motion blur risultante dalla traslazione della camera è stato sempre trattato nella letteratura assumendo la traslazione parallela al piano immagine. Sotto questa ipotesi il blur risulta essere spazio invariante, la Point Spread Function (PSF) è quindi fissata per tutti i pixel dell'immagine. In alcuni casti, tra cui quello in cui la traslazione presenta una componente ortogonale al piano immagine, il blur diventa spazio variane e la PSF varia da pixel a pixel. Questa situazione, che è più difficile da gestire sia per l'analisi del blur che per la ricostruzione dell'immagine, è stata quasi trascurata nella letteratura.

La tesi è suddivisa in due parti. La prima parte è dedicata ad algoritmi per stimare la PSF localmente. In questa parte sono presentati algoritmi per stimare, da una sola immagine mossa, i parametri del blur in una regione dove l'immagine presenta un punto saliente (corner). I metodi presenti prima d'ora nella letteratura non sono in grado di stimare la PSF in regioni contenenti un

corner, sebbene queste offrano una chiara interpretazione del blur. E' stato anche proposto una procedura per selezionare automaticamente ed in maniera adattativa la regione del corner.

Nella seconda parte della tesi consideriamo l'intera immagine mossa per la stima del moto della camera e la ricostruzione dell'immagine stessa. La maggior parte del lavoro è stato rivolto ad immagini acquisite durante la traslazione della telecamera. A partire da una descrizione geometrica tridimensionale è stato ricavato un modello per descrivere la formazione di un'immagine acquisita durante una traslazione generica della camera. In questo modello il blur viene rappresentato da un operatore parametrico e spazio variante. E' stato poi proposto un algoritmo per stimare la dirzione 3D del moto della camera, data una sola immagine mossa e quindi la PSF ad ogni pixel.

E' stato quindi ricavato un algoritmo per ricostruire immagini corrotte dal blur dovuto ad una traslazione generica (blur radiale), che è quindi spazio variante. L'algoritmo assume noti i parametri del blur ed è basato sull'inversione del blur e la rimozione del rumore, alterato dall'inversione del blur. Abbiamo anche presentato un metodo per stimare la profondità della scena a partire da un'immagine mossa acquisita durante la traslazione della camera.

E' stato poi affrontato il caso di motion blur prodotto da una rotazione della camera. A differenza di tutte le soluzioni esistenti, è stato trattato senza approssimazioni il caso generale in cui asse di rotazione della camera non è ortogonale al piano immagine. E' stato proposto quindi un algoritmo per stimare, da una sola immagine corrotta da blur rotazionale, la posizione 3D dell'asse di rotazione e la velocità angolare della camera.

Contents

Part I.

Motion Blur Analysis at Corners

The first part of this thesis presents novel algorithms for analyzing the blur in a single image acquired during a camera motion. Often, the blur due to camera motion (motion blur) can not be considered as spatially invariant (i.e. shift invariant). For example, when the camera is translating one can observe smears in resulting image which have varying direction and extent between image pixels, as shown in Figure 0.1.**a** and Figure 0.1.**b**.

Spatially variant blur estimation is a challenging problem which is often simplified by assuming the blur as locally spatially invariant [4], i.e. for each image pixel, one assumes that there exist a region where the blurring process is modeled as a convolution against a Point Spread Function (PSF). This simplification has been made also in the first part of this thesis, which focuses on the estimation of the *local blur parameters* and the detection of regions where the blur can be considered as spatially invariant. We further assume that each PSF is described by two parameters, its direction and its extent. While this assumption is too restrictive when combined with the spatially invariant blur assumption, this allows a faithful interpretation of motion blur when assuming spatially invariant blur.

The algorithms presented in Chapter 1 are meant for regions containing an image corner: this image content is exploited as a prior for estimating the blur parameters. To the author's knowledge, the issue of estimating blur parameters within a small region containing a corner has never been addressed before. The use of small regions for estimating the blur parameters is crucial for obtaining a reliable approximation of spatially variant blur (a region is considered small if its sizes are about two, three times the blur extent).

There are three main reasons why it is worth estimating the blur parameters in a region containing a corner. First, within a corner region, the blur parameters can be determined as the *aperture problem* does not hold. The term aperture problem, has been introduced in studies on vision systems [38], to express the ambiguity that moving homogeneous contours present. Within a region (an aperture) of a blurred image, different physical motions may be indistinguishable: this happens for example in smooth areas or at blurred edges, where infinite pairs (direction , extent) of blur parameters can describe the blur. Figure 0.1.c shows some of the motions that correctly interpret the blur in the highlighted region. On the other hand, thanks to the aperture problem, pixels belonging to blurred edges can be used for estimating the blur in a corner region, as at these pixels the blur parameters can be assumed equal to those at the corner. Second, corners are ubiquitous in images and often carry the most relevant information for scene understanding: as a matter of fact, corners are widely used in Computer Vision and Pattern Recognition for extracting features in images. Analyzing the blur at corner regions is therefore useful both for scene and motion understanding. Third, there are no other blur parameters estimation algorithms able to perform on small regions containing a corner. Although several algorithms for estimating blur parameters have been proposed for blind deconvolution, all these algorithms work on the whole image, assuming the blur spatially invariant, and they do not perform adequately when applied on a small region containing a corner.

Most of motion blur PSF parameters estimation algorithms analyzes the Fourier power spectrum of the image, see [15, 24, 49, 67, 68], and references therein. Fourier-domain methods exploit the convolution theorem [61] which claims that the Fourier transform of the convolution of two periodic signals is the point-wise product of the Fourier transforms of the two signals. Since the Fourier transform of these PSFs present characteristic zero patterns, which can be directly related to the PSF parameters, these zero-patterns should be distinguishable also in the Fourier transform of the blurred image. Figure 0.2 **b** shows how the zero patterns are clearly distinguishable in the Fourier transform of blurred white noise. However, this approach fails at small regions containing a corner because of two reasons: first, in a small image a region the image can be roughly approximated as periodic and second, the Fourier transform of the blurred image is mostly influenced by blurred edges rather than from the PSF as shown in Figure 0.2 **d**.

A different approach to PSF parameters estimation has been introduced by Yitzhaky [93, 94, 92] that proposed to estimate the blur direction as the direction of the derivative filters having minimum energy response. The blur extent is estimated consequently detecting the minima of the autocorrelation function of image derivatives along blur direction. This method does not present the restrictions on regions sizes of Fourier transform based methods but does not perform correctly at blurred corners. In fact, typically an edge direction may represent the direction having minimum derivative energy, as illustrated in Figure 0.3 **d**.

Chapter 2 addresses two complementary issues: the detection of blurred corners and the corner region selection. Corner regions are adaptively selected in order to improve the proposed algorithm performance.

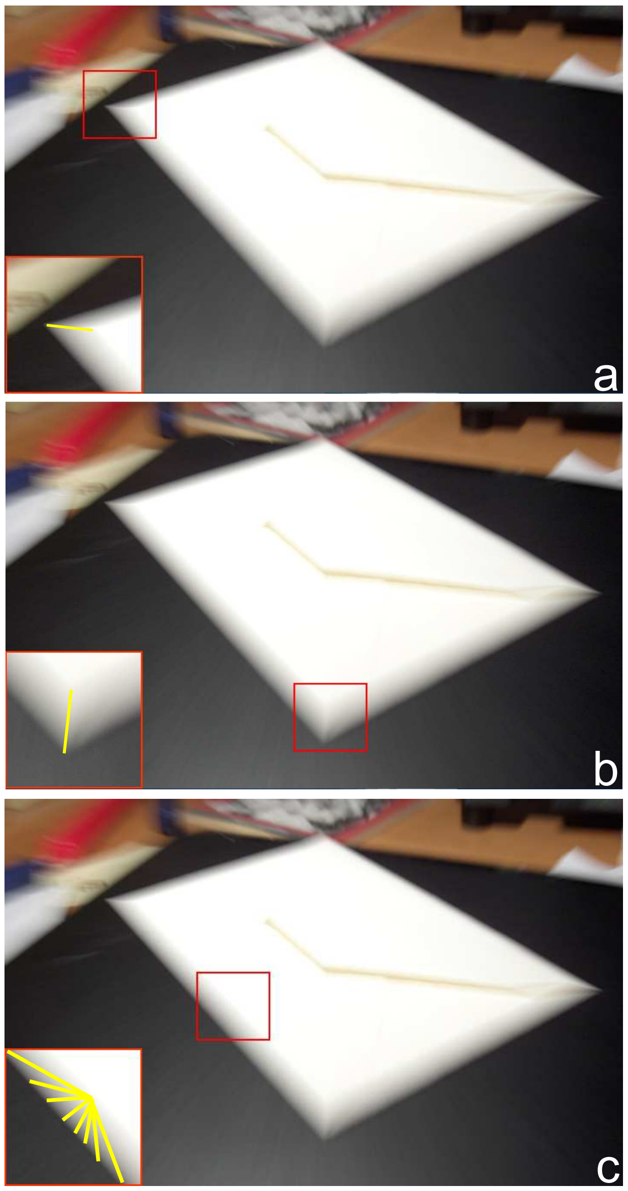


Figure 0.1.: Image corrupted by spatially variant blur. Blur is spatially variant as the highlighted regions in **a** and **b** show. At blurred edges, there are infinite blur direction/ blur extent pairs that represent the blur, see **c**.

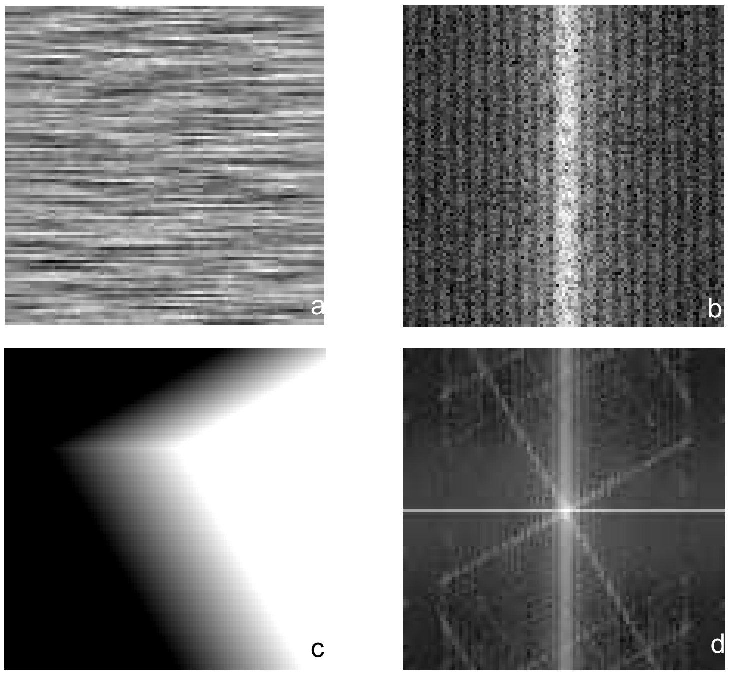


Figure 0.2.: Blur parameters estimation using Fourier power spectrum. The Fourier power spectrum of motion blur PSFs presents zero-patterns, which are clearly distinguishable on blurred white Gaussian noise, Figure 0.2**a,b**. The direction of these zero patterns is related to the blur direction. However, the Fourier power spectrum of a blurred corner is mostly influenced by the image steps due to edges than from the PSF parameters, Figure 0.2 **c,d**.

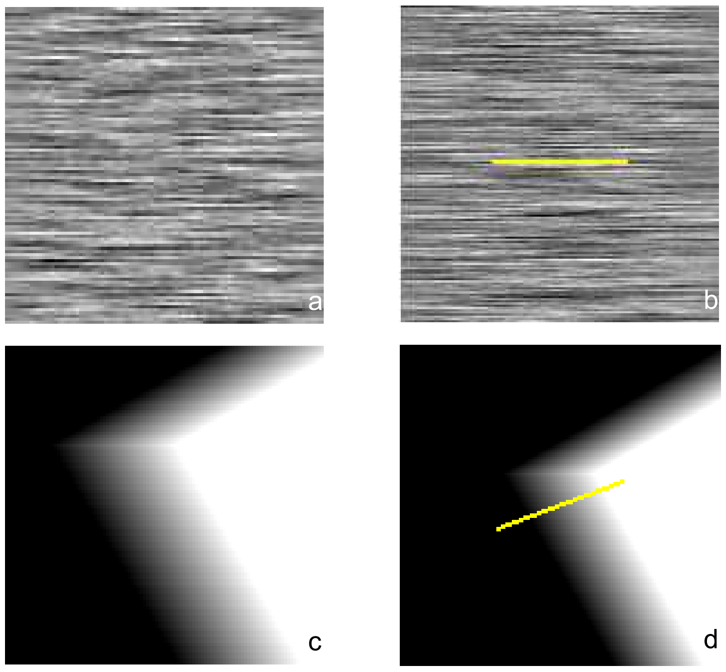


Figure 0.3.: Blur Parameters Estimation using derivative filters. The direction having minimum energy response on white Gaussian noise is the blur direction, Figure 0.3**a,b**. However, at blurred corners, the minimum energy direction is typically the one of the steepest blurred edge, Figure 0.3**c,d**.

1. Estimation of Blur Parameters at Corners

This chapter presents an observation model for motion blurred images and in particular focuses on regions containing image corners. The proposed model considers both blur and noise. We assume that each corner is blurred by a convolution against a Point Spread Function (PSF) having vector-like support.

We address the issue of estimating the blur direction and extent in a region containing a corner, and we derive three solutions.

Experimental results both on synthetic and camera images, show accurate estimates of the PSF in small corner regions. The algorithms described in this chapter have been published in two papers in conference proceedings, see [5, 6].

1.1. Motivation

This chapter presents novel algorithms to estimate spatially varying blur from a single image, assuming that the relative motion between the camera and the scene produces "rectilinear smears" in the blurred image. The image is therefore blurred along line segments whose direction and extent are varying between the image pixels.

This observation model is particularly suited for images acquired by agile cameras in indoor environments [14, 19]. These images, when captured at reduced lightning conditions are often motion blurred: as the exposure time is (automatically) increased in order to acquire normally exposed frames, the camera motion results in significant blur in the recorded frame. Moreover, when the scene presents various depths, the resulting blur is strongly spatially varying.

The proposed algorithm analyzes image blur specifically at regions that contains a corner as at blurred corners the *aperture problem* [38] does not hold, contrarily to blurred edges. Figures 0.1 and 1.1 show a naive description of what is the aperture problem is when : the blur direction and the blur extent can be clearly perceived within regions A and D of Figure 1.1, while it is not possible to give a unique interpretation to the blur direction and the blur extent in regions B and C. The same situation is faced in regions where the image is smooth: the aperture problem still holds as there are potentially infinite pairs blur direction/blur extent, that could have caused the same blur.

Corners, instead, offer a clear interpretation of motion direction and extent and that's the reason why we design an algorithm to estimate motion blur specifically at corners. At the same time, regions containing an image corner can be easily detected and the image content easily modeled. Moreover, corners often correspond to boundaries between scene objects, and therefore they are relevant for motion understanding.

1.2. Related Works

Pixel motion estimation is a relevant issue for both image processing and computer vision, as it is often required as a preprocessing step in several algorithms. When a significant displacement occurs between the camera and the scene during the exposure, this results in a blur in the acquired image. The blur heavily corrupts image quality and the estimation of the blurring process is a challenging problem.

Sometimes, the observer can exploit few images capturing the same scene [2, 48, 74] or images produced by hybrid imaging systems that, for example, employ simultaneously two cameras [3] or acquire sequentially two images varying the exposure [62, 96]. Clearly, when a single image is available, the blur analysis becomes more complicated and it is typically handled by introducing simplifying assumptions on image blur and exploiting *a priori* information on the original image, when available.

In most of cases the blur is assumed a linear and spatially invariant system on the image. Thus it corresponds to a convolution of the ideal, original image (representing the captured scene without any artifact introduced by the acquisition device such as blur or noise) with an unknown kernel, the Point Spread Function (PSF) of the blurring process.

Algorithms that pair blur estimation and image restoration from a single image (blind deblurring) have been widely studied in the last decades, [11, 12, 52]. Recently, Fergus *et al.* [25] showed good performance in camera shake removal from a single blurred photograph by using a Gaussian mixture prior for the distribution of gradient norms in natural images. They assume spatially invariant blur on the image, as most of deconvolution based algorithms do. They also assume non parametric PSFs which seem to describe faithfully the blur produced by camera shake in camera images. Levin [54] relaxed the spatially invariant blur assumption and devised a blind deblurring algorithm combined with a segmentation of the image in (few) areas having the same blur extent, but again, the blur is assumed to have the same direction in every image pixel.

Most of the literature algorithms for parametric PSF estimation from a single blurred image are based on the well known property that a convolution in space domain becomes a pixel-wise product in Fourier Domain. In particular motion blur PSFs are typically assumed as kernels having vector-like support and constant value on it. In this case the Fourier power spectrum presents particular patters which are related to PSF parameters that can be thus estimated with several analysis methods [15, 24, 49, 52, 67, 68, 75, 76].

A different approach is based on high-pass filtering the image in space domain [94, 95] with directional filters and taking the direction having minimum energy response.

Also the wavelet transform [18, 61] has been used in order to estimate the PSF from a single blurred image. Rooms *et al.* [26] exploits the sparsity of the wavelet subbands for estimating the PSF parameter. They restrict to Gaussian PSF described by one parameter, even if the method could be extended to other parametric PSFs families.

Blur estimation from a single blurred image has been addressed for several purposes other than deblurring: optical flow estimation [75, 76], its integration in a tracking system [49], the measurement of vehicles [56] and balls speed [57] or scene depth [53, 58]. Klein [51] recently developed a gyroscope by measuring the rotational blur in each video frame of a robot-mounted camera.

The most interesting and straightforward application for the algorithms proposed in this chapter is the optical flow estimation from a single motion blurred image. The optical flow in the blurred image shows the motion that the camera underwent during the exposure. This issue has been initially addressed by Rekleitis [75, 76]. He proposed a Fourier transform based algorithm for handling spatially variant blurred images, treating the blur as locally shift invariant. He defines a tessellation on the image and then analyzes the Fourier power spectrum of each of the tessellation regions separately. However, all the frequency-domain based algorithms does not allow correct blur parameters estimation in image regions having small sizes, as pointed out in the introduction of this part and as illustrated in Figure 0.2. Finally, a considerable drawbacks of using a fixed fixed tessellation is that blur estimate could be strongly biased by the image content. This happens for example when a corner is divided into two different regions of the tessellation. None of the two regions allows a correct interpretation

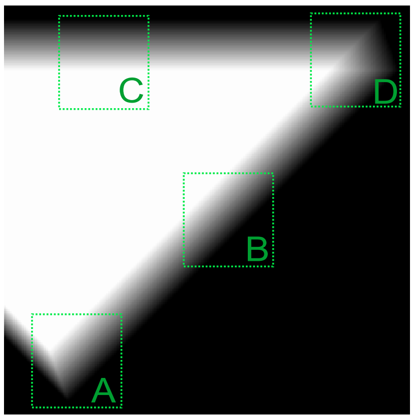


Figure 1.1.: Synthetic blurred corner

1.3. Problem Formulation

The observation z, corrupted by spatially varying blur and noise is modeled as

$$z(x) = K\left(y\right)(x) + \eta(x)\,, \quad x \in \mathcal{X} \tag{1.1}$$

x being a vector representing image coordinates varying on the discrete domain $\mathcal{X}$, y is the original (and unknown) image and K the blur operator. The term η models quantization errors and the electronic and thermal noise, which are together modeled as white noise η.

1.3.1. The Blur Model

We assume K as a linear blur operator and therefore, in its more general form, is [4]

$$K\left(y\right)(x) = \int_{\mathcal{X}} k(x,s)y(s)ds\,,\ \forall x \in \mathcal{X}\,, \tag{1.2}$$

where $k(x, \bullet)$ represent the Point Spread Function (PSF) at x, i.e. the response of the blur process to a point source at x in the original image. It thus describes how the intensity of a pixel x in the original image, $y(x)$, is spread "or mixed" with the neighboring pixels in the blurred observation.

Typically K is assumed spatially invariant, i.e. $k(x,s) = k(x-s)$, therefore Equation (1.2) becomes a convolution against a PSF h:

$$K\left(y\right)(x) = \int_{\mathcal{X}} h(x-s)y(s)\,ds \;= (h \circledast y)(x)\,,\ \forall x \in \mathcal{X}\,, \tag{1.3}$$

where $\circledast$ denotes the convolution operator.

The assumption about spatially invariant blur is too restrictive as scene points usually follow different trajectories with respect to the camera viewpoint and this results in spatially variant blur in the image. In other words each pixel has been processed with a different PSFs. In a broad scenario, spatially invariant blur of Equation (1.3) does not describe, for instance, scenes containing several moving objects, scenes with a moving target on a still background or non planar scenes captured by a moving camera.

On the other hand, solving (1.2) is an extremely challenging inverse problem. To reduce its complexity the blur operator K is *locally* approximated as shift invariant blur, i.e.

$\forall x_0 \in \mathcal{X}, \exists U_0 \subset \mathcal{X}\ ,\ x_0 \in U_0$ and a PSF v_0 such that

$$K\left(y\right)\left(x\right) \approx \int_{\mathcal{X}} v_0(x-s)y(s)ds \; \forall x \in U_0 \, . \tag{1.4}$$

Furthermore, we consider only parametric PSFs defined over an 1-D linear support. These PSFs can be written as

$$v_0 = R_{(\theta)}\left(s_l\right)\left(x\right) \; \theta \in [0, 2\pi], l \in \mathbb{N}, x \in U_0 \, . \tag{1.5}$$

$$s_l(x_1, x_2) = \begin{cases} 1/(2l+1), & -l \leq x_1 \leq l \\ & x_2 = 0 \\ 0, & else \end{cases}$$

where θ and l are motion direction and extent respectively and $R_{(\theta)}\left(s_l\right)$ is s_l rotated by θ degrees on $\mathcal{X}$.

1.3.2. The Corner Model

Let y be a gray scale image or, equivalently, a color plane in a color image and let $A \subset \mathcal{X}$ be a region containing a corner. The image contains a *binary corner* if $y(A) = \{b, c\}$, where b and c are image intensity values for the background and the corner, respectively. Moreover, $B = y^{-1}(\{b\})$ and $C = y^{-1}(\{c\})$, the sets of pixels belonging to the background and to the corner, have to be separated by two straight segments, having a common endpoint. An example of binary corner is shown in Figure 1.2.

Let us define $\tilde{v}$ as the corner displacement vector: this vector has the origin at the location of the corner pixel on the image grid and the direction θ and the length l corresponding to the direction and the length of v_0: the PSF which locally approximates the blur operator.

We further assume that $\tilde{v}$ direction belongs to an interval determined by the edges direction. Let introduce a reference axis in the image and let α be the amplitude of the corner angle, let θ be the direction of $\tilde{v}$ w.r.t. the reference

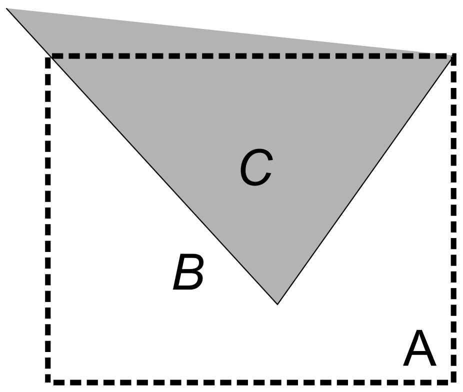


Figure 1.2.: The Binary Corner model

axis and let γ be the orientation of the corner bisector, as illustrated in Figure 1.3. Thus, we initially restrict ourseves to corners having displacement vector satisfying the following assumption

$$\theta \in [\gamma - \alpha/2, \gamma + \alpha/2] \; + k\pi \quad k \in \mathbb{N}. \tag{1.6}$$

Figure 1.3.**a** shows a corner displacement vector satisfying this assumption, while Figure 1.3.**b** shows a corner that does not.

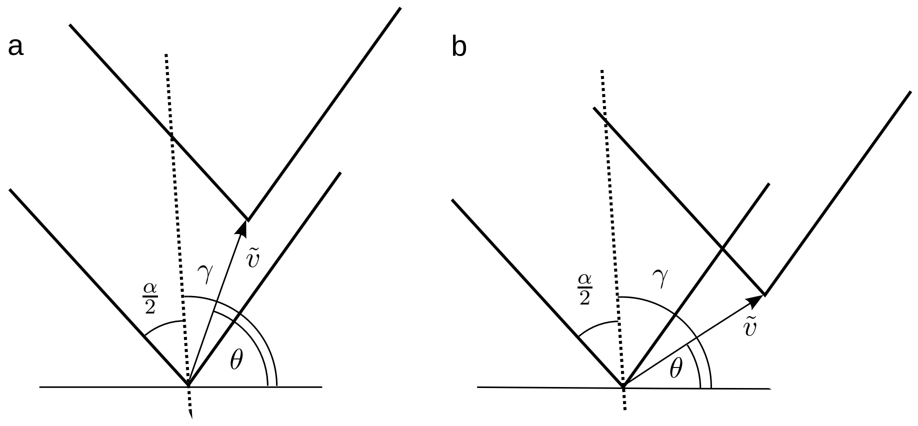


Figure 1.3.: Two possible cases for corner displacements, **a** satisfy Equation (1.6), while **b** does not.

In real images corners are, in general, not binary. It is reasonable to expect corners to be distinguishable from their background, but hardly they would be uniform. Often their intensities are varying due, for example, to texture, shading or details presented.

Therefore, in order to take into account corners with few, unstructured details, we introduce in Equation (1.1), another source of white noise ξ. This is done only for corner regions where the image content is being modeled. Thus, within a corner region A, the observation is described as follows

$$z(x) = K\left(y + \xi\right)(x) + \eta(x)\,, \quad x \in A \tag{1.7}$$

ξ being white noise.

By Equation (1.7), the result of blurring of non-binary corner is considered equivalent to the result of blurring a binary corner contaminated by white noise.

1.3.3. Problem Formulation

After having specified the corner and the blur model, we formulate the addressed problem.

Given an image region A that contains a blurred corner, satisfying the requirements stated in Sections 1.3.1 and Section 1.3.2, our goal is to estimate the corner displacement vector $\tilde{v}$, and thus the direction θ and extent l of the PSF representing the blur in A.

1.4. Least Squares Solution

In this section we derive the core equations for estimating the corner displacement vector $\tilde{v}$ (and thus the blur parameters) within a region A that contains a blurred corner. First, we present the proposed least squares solution assuming binary corners, and then we consider how the estimated displacement vectors change when corners are not binary.

1.4.1. Binary Corners, Least Squares Solution

Let us examine an image region containing a binary corner, like the one depicted in Figure 1.2. Let d_1 and d_2 be the first order derivative filters w.r.t. horizontal and vertical direction respectively. The image gradient is defined as

$$\nabla z(x) = \begin{bmatrix} z_1(x) \\ z_2(x) \end{bmatrix} = \nabla K\left(y\right)(x) + \nabla\,\eta(x)\,, \forall x \in \mathcal{X}\,,$$

where $z_1 = (z \circledast d_1)$ and $z_2 = (z \circledast d_2)$.

It follows that, denoting by $\Delta = |c - b|$ the difference between the image intensities at corner and at the background as shown in Figure 1.4,

$$\Delta = \tilde{v}\,\cdot\,\nabla K\left(y\right)\,(x)\,, \quad \forall x \in A_0\,, \tag{1.8}$$

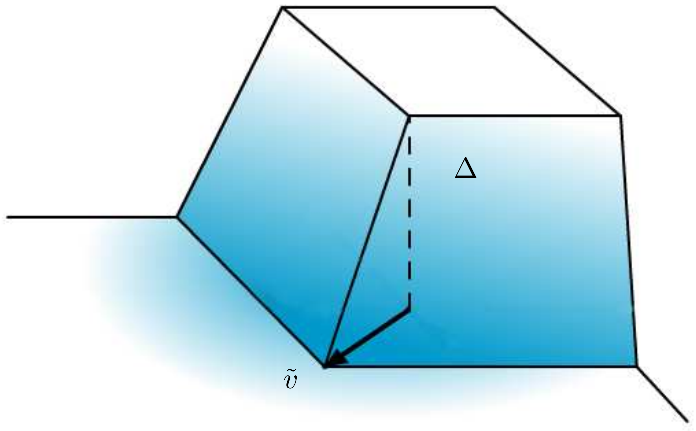


Figure 1.4.: Mesh of pixel intensities within region A of Figure 1.1. The displacement vector $\tilde{v}$ and Δ , the difference between corner and background colors are shown.

where $A_0 = \{x \in A \,|\, \nabla K\,(y)\;(x) \neq [0,0]^T\}$, $\cdot$ denotes the scalar product and $\tilde{v}$ is the corner displacement vector (one column vector).

Note that the scalar product in right hand side of Equation (1.8) is positive. This follows from the assumption stated in Equation (1.6), since $\tilde{v}$ forms with the gradient $\nabla K\,(y)\,(x)$ an angle smaller than $\pi/2$.

Equation (1.8) has no unique solution as Δ and $K\,(y)$ are unknown and only z, which is however corrupted by noise η, is known. Similar uncertainties are typically resolved by considering several instances of Equation (1.8), each one evaluated in a different pixel in A_0. In fact, Equation (1.8) gives the same solution $\forall x \in A_0$.

We call w a window described by its weight w_i , $-n < i < n$, and we solve the following system

$$M(x)\,\tilde{v} = \,\Delta\,[w_{-n},...,w_0,...,w_n]^T\;\forall x \in A_0\,, \tag{1.9}$$

where the matrix M is defined as

$$M(x) = \begin{bmatrix} w_{-n}\,\nabla z(x_{-n})^T \\ ... \\ w_0\,\nabla z(x)^T \\ ... \\ w_n \nabla z(x_n)^T \end{bmatrix}.$$

In our experiment we choose w as a squared window having Gaussian distributed weights.

The least square solution of Equation (1.9), $\tilde{v}$ is

$$\tilde{v} = \arg\min_{v} \left\| M(x)\, v - \Delta \left[w_{-n}, ..., w_0, ..., w_n\right]^T \right\|_2 , \forall x \in A_0 , \qquad (1.10)$$

which yields

$$\tilde{v} = H^{-1}(x)\, M^T(x)\ \left[w_{-n}, ..., w_0, ..., w_n\right]^T\ \forall x \in A_0 , \qquad (1.11)$$

$$H = \begin{bmatrix} \sum\limits_i w_i^2 z_1(x_i)^2 & \sum\limits_i w_i^2 z_2(x_i) z_1(x_i) \\ \sum\limits_i w_i^2 z_2(x_i) z_1(x_i) & \sum\limits_i w_i^2 z_2(x_i)^2 \end{bmatrix} .$$

Note that H corresponds to the Harris Matrix [35], whose determinant and trace are used as corner detectors in many feature extraction algorithms, see [65] and references therein.

Note also that when A_0 does not contain any image corner, the determinant of H is close to zero and when $z_1(x) = k$ and $z_2(x) = k'$, $\forall x \in A_0$, k and $k' \in \mathbb{R}$, H is singular and consequently the system (1.11) has no a unique solution. This happens when the corner region intersects only one blurred edge (like regions B and C in Figure 1.1). Then the system (1.11) has an infinite number of solutions and the motion parameters cannot be estimated.

On the contrary, H is nonsingular when w intersects two blurred edges (like region A in Figure 1.1) and in this case the system (1.11) has an unique solution.

The least squares solution (1.11) performs optimally in case of Gaussian white noise (here we assume that η is white noise, without specifying any distribution). However, in low-noise regions (i.e. regions where noise standard deviation is significantly lower than Δ), Equation (1.11) represent a suboptimal solution.

Equation (1.11) gives a solution which depends on the considered pixel $x \in A_0$: let us denote this solution as $\tilde{v}(x)$. When a different pixel x' is taken in A_0, the solution $\tilde{v}(x')$ may be different, as the window w is centered at x'. Therefore a procedure is required to determine the most reliable solution $\tilde{v}$ from the set of solutions $\{\tilde{v}(x_i)\}_i$, $\forall x_i \in A_0$.

A solution consists of taking the corner displacement vector $\tilde{v}(x)$, x being the center of A. A less naive solution can be based on statistical analysis on the set $\{\tilde{v}(x_i)\}_i$, $\forall x_i \in A_0$. For example, $\tilde{v}$ can be taken as the mode of vectors in $\{\tilde{v}(x_i)\}_i$, $\forall x_i \in A_0$ or by analyzing separately directions or extents of the estimated displacement vectors with a median or a weighted median. The weights can be determined as proportional to $|det(H)(x)|$ as it determines "how much" there is an image corner at x, according to Harris corner detector, [35].

1.4.2. Non Binary Corners

The algorithm proposed in Section 1.4.1 assumes that the region A corresponds to a region in the original image where $y(A) = \{b, c\}$, i.e. there are only two intensity values in y. These "cartoon world" corners are far from being similar to corners of camera images, as noted in Section 1.3.2.

Let then consider how Equation (1.8) changes if it presents also the noise ξ,

$$\nabla z(x) = \nabla K\,(y)\,(x) + \nabla K\,(\xi)(x)\,,\ \forall x \in \mathcal{X}\,. \tag{1.12}$$

Note that Equation (1.8) holds for $\nabla K\,(y)\,(x)$, while it does not for $\nabla K\,(\xi)\,(x)$, the second term in the right-hand side of Equation (1.12). However, the blur operator $K\,(\xi)$, which is locally a convolution with a PSF, produces a correlation of ξ samples along the motion direction [94], so that we have

$$\nabla K\,(\xi)\,(x)\,\cdot\,\tilde{v}\,, \approx 0\,,\ \forall x \in A_0\,, \tag{1.13}$$

which means that the larger the blur extent is in the considered region, and consequently the correlation among random values of ξ increases, the more our algorithm will work for regions where corners are not binary.

1.5. Robust Solution

Although Equation (1.13) assures that the proposed algorithm would work even in presence of noise ξ, we expect that outliers would heavily influence the solution of Equation (1.11), since it comes from the ℓ^2 error norm minimization of Equation (1.10).

Beside pixels where $\nabla K\,(\xi)\,(x)\,\cdot\,\tilde{v} \neq 0$ there could be several other noise factors that are not considered in our model but that we should be aware of. For example compressed images often present artifacts at edges such as ringing or blocking, corners on y are usually smoothed and edges are not perfectly straight lines.

However, if we assume that outliers are a relatively small percentage of pixels, we can still obtain a reliable corner displacement vector estimate using a robust technique to solve (1.9). We do not look for a vector $\tilde{v}$ which satisfies Equation (1.8) at each pixel, or which minimizes the ℓ^2 error norm (like in Equation (1.10)): rather we look for a value of $\tilde{v}$ which satisfies a significant percentage of equations in System (1.9), disregarding how $\tilde{v}$ is far from the solution of the remaining equations.

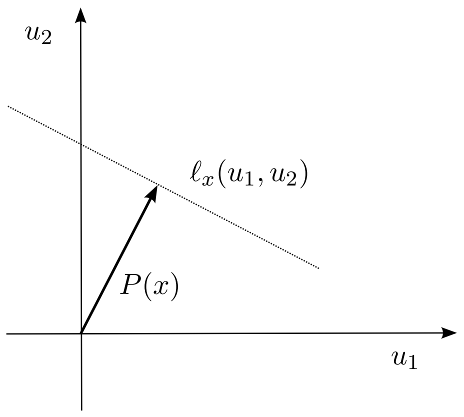


Figure 1.5.: $\ell_x(u_1, u_2)$ set of possible endpoint for $\tilde{v}(x)$

1.5.1. The Voting Approach

For every pixel $x \in A_0$, we define $P(x)$ as the projection of the displacement vector along the gradient direction

$$P(x) = \left[\tilde{v} \cdot \frac{\nabla z(x)}{||\nabla z(x)||}\right] \frac{\nabla z(x)}{||\nabla z(x)||}\,, \forall x \in A_0\,, \tag{1.14}$$

where $\cdot$ denotes the scalar product. It follows that, when the assumption of Equation (1.6) holds, from Equation (1.8),

$$P(x) = \frac{\nabla z(x)}{||\nabla z(x)||^2}\,\Delta\,, \tag{1.15}$$

we have that $P(x)$ corresponds to the $\tilde{v}$ component along $\nabla z(x)$ direction, $\forall x \in A_0$.

The endpoint of any vector $\tilde{v}$, solution of (1.8), lies on the straight line perpendicular to $P(x)$, going through its endpoint.

As in usual Hough approaches, we consider a (2-D) parameter space $\mathcal{U}$, containing all the possible location for the endpoints of $\tilde{v}$. The parameter space is subdivided into cells of suitable size (e.g. 1 pixel) and indexed by two coordinates, (u_1, u_2). Let define $\ell_x(u_1, u_2)$ as the locus of the possible endpoints of $\tilde{v}$, compatible with a given datum $\nabla z(x)$: it follows that $\ell_x(u_1, u_2)$, is a line as shown in Figure 1.5. A vote is then assigned to each cell that contains a value of $\tilde{v}$ that satisfies an instance of Equation (1.8), i.e. ℓ_x. This vote is summed to the votes coming from the other data. The most voted cells represent values of $\tilde{v}$ that satisfy a significant number of equations (1.9).

1.5.2. Neighborhood Construction

In order to reduce the approximation errors due to the discrete parameter space and to take into account $\nabla\eta$, we construct an ad-hoc neighborhood ℓ_x for assigning votes in the parameter space. We build ℓ_x in order to assign a full vote (e.g 1) to each parameter pair which solves Equation (1.8) (the line of Figure 1.5), and a fraction of vote to the neighboring parameter pairs. We define thus the following weight function

$$\ell(u_1, u_2) = exp\Big[- \Big(\frac{u_2}{1 + k|u_1|\sigma_{\nabla\eta}} \Big)^2 \Big] \,, \tag{1.16}$$

where $\sigma_{\nabla\eta}$ is $\nabla\eta$ standard deviation and k is a tuning parameter. The weight function ℓ has the following properties: it is constant and equal to 1 on u_1 axis, (i.e. $\ell(u_1, 0) = 1$), and when evaluated on a vertical line, $(u_1 = const)$, it is a Gaussian function having standard deviation that is proportional to $|u_1|$, i.e.

$$\ell(u_1, u_2) \sim N(0, 1 + k|u_1|\sigma_{\nabla\eta})(u_2) \,,\ \forall (u_1, u_2) \in \mathcal{U}$$

We select this weight function as a prototype of the vote map: given $\nabla z(x)$, the votes are distributed in the parameter space as the values of ℓ opportunely translated and rotated. The straight line of Figure 1.5, $\ell_x(u_1, u_2)$, is therefore replaced by function ℓ rotated by $(\frac{\pi}{2} - \theta)$ degrees and translated so that its origin is in $P(x)$ endpoint, i.e.

$$\ell_x(u_1, u_2) = R_{(\frac{\pi}{2} - \theta)}\,(\ell)\,([u_1, u_2]^T - P(x))\,,\ \forall (u_1, u_2) \in \mathcal{U}\,,\ \ ,\ \forall x \in A_0\,. \tag{1.17}$$

where θ is $\nabla z(x)$ direction and $R_{(\frac{\pi}{2} - \theta)}$ is the rotation of $(\frac{\pi}{2} - \theta)$ degrees.

In such a way, we give a full vote to parameter pairs which are exact solutions of (1.8) and we increase the spread of votes as the distance from $P(x)$ endpoint increases.

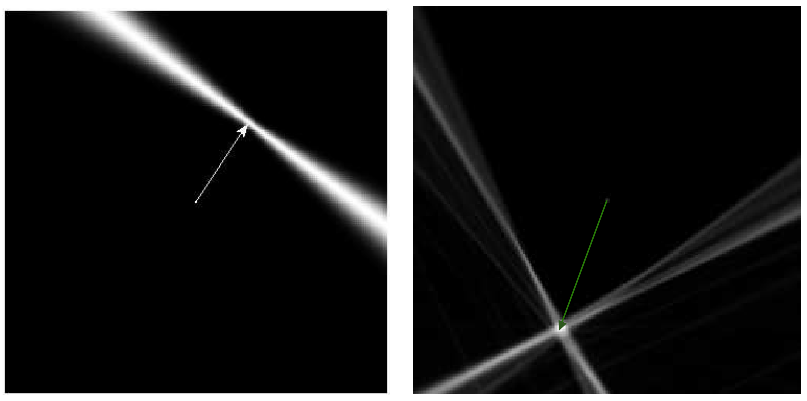
Figure $1.6(a)$ shows how votes are distributed in parameter space for a vector $P(x)$. Figure 1.6(b) shows parameter space after having considered all data, the arrow indicates the vector $\tilde{v}$ estimated.

1.6. Performance of the Algorithm Based on the Hough Transform

1.6.1. Implementation Details

Given a region A containing a blurred corner, we proceed as follows

- Estimate σ_η on the whole image, using the linear filtering procedure proposed in [42] or any other method presented in Appendix 7.



(a) (b)

Figure 1.6.: Votes in Parameter Space. (a) An illustration of the weight function $\ell_x(u_1, u_2)$, used to spread the votes in the parameter space. The vector represents the projection vector $P(x)$. (b) The sum of votes in the parameter space, after having considered all data. The green vector represents the estimated corner displacement vector $\tilde{v}$.

- Define A_0, the set of considered pixels as

$$A_0 = \{x \in A \mid ||\nabla z(x)|| > T\}$$

  $T > 0$ being a fixed threshold. In such a way we exclude image pixels where the value is constant but gradient is non zero because of ξ and η. The threshold T is typically defined as $T = n_t \sigma_\eta$, n_t being a tuning parameter.

- Estimate Δ as $\Delta = |max(A_0) - min(A_0)| + n_t \sigma_\eta$.

- Voting: $\forall x \in A_0$ distribute votes in parameter space computing $\ell_x(u_1, u_2)$ and adding them to the previous votes. The k parameter used in (3.9) is chosen between $[0.02, 0.04]$.

- The solution of System (1.9), $\tilde{v}$, is the vector having endpoint in the most voted coordinates pair. Whenever several parameter pairs receive the maximum vote, their center of mass is selected as $\tilde{v}$ endpoint.

- To speed up the algorithm, we eventually consider gradient values only at even coordinate pairs.

1.6.2. Experiments

We validate the proposed algorithm with tests on synthetic images, on test images synthetically blurred, and on camera images blurred because of camera motion.

Synthetic Images

We generate synthetic images according to Equation (1.7), using a binary corner (according to the model of Section $1.3.2$), taking y constantly equal to 0 at background and equal to 1 at corner pixels and we add η and ξ with Gaussian distribution.

Corner displacement vectors have been estimated on several synthetically generated images with values of the standard deviations $\sigma_\eta \in [0, 0.02]$ and $\sigma_\xi \in [0, 0.08]$. Blur has been produced by a convolution with a PSF h having direction 10 degrees and length 20 pixels in the first case and 70 degrees and 30 pixels in the second case.

Figure 1.7 and Figure 1.8 show some test images and Table 1.1 and Table 1.2 present algorithm performance in terms of distance, in pixel unit, between the endpoints of the estimated, $\tilde{v}$, and the true displacement vector v, expressed as a percentage w.r.t PSF length.

Comparing the first two rows of Table 1.1 with those of Table 1.2, we note how stronger blurs introduce higher correlation between ξ samples and gives more accurate results, as expressed in Equation (1.13).

When $\sigma_\eta = 0.02$ the algorithm accuracy is significantly decreased.

$\sigma_\eta \mid \sigma_\xi$	0	0.02	0.04	0.06	0.08
0	1.94%	2.37%	1.67%	3.26%	5.40%
0.01	6.54%	2.98%	1.67%	4.21%	1.68%
0.02	4.14%	7.57%	5.40%	3.97%	3.35%

Table 1.1.: Result on synthetic images: v has direction 10 degrees and length 20 pixels, $\sigma_\eta \in [0, 0.02]$ and $\sigma_\xi \in [0, 0.08]$.

$\sigma_\eta \mid \sigma_\xi$	0	0.02	0.04	0.06	0.08
0	1.95%	1.08%	1.95%	2.23%	0.98%
0.01	3.04%	0.31%	3.99%	1.43%	2.54%
0.02	9.39%	10.11%	6.55%	7.65%	7.50%

Table 1.2.: Result on synthetic images: v has direction 70 degrees and length 30 pixels, $\sigma_\eta \in [0, 0.02]$ and $\sigma_\xi \in [0, 0.08]$.

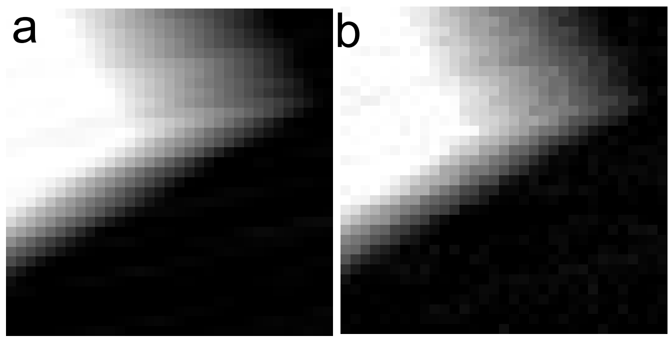


Figure 1.7.: Synthetic test images used PSF directed 10 degrees and length 20 pixels, in **a** $\sigma_\eta = 0$ and $\sigma_\xi = 0.08$, while in **b** $\sigma_\eta = 0.02$ and $\sigma_\xi = 0$.

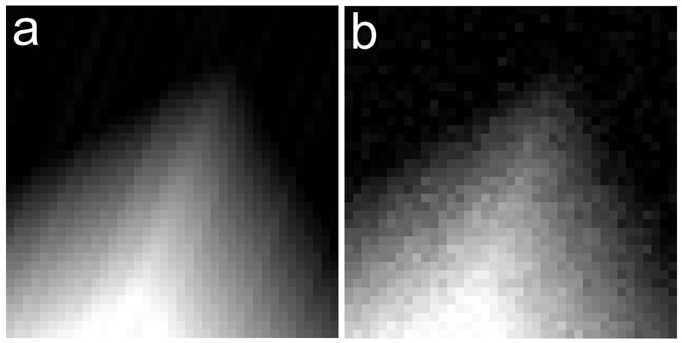


Figure 1.8.: Example of synthetic test images used, PSF was directed 70 degrees and length 30 pixels, in **a** $\sigma_\eta = 0$ and $\sigma_\xi = 0.08$, while in **b** $\sigma_\eta = 0.02$ and $\sigma_\xi = 0$.

Synthetically Blurred Real Test Image

We replace the original image at corners, i.e. $y + \xi$ with a common test image and we blur it using a convolution with a PSF on the whole image. We finally add white Gaussian noise η and analyze the blurred noisy image within some regions containing a corner.

We take *house* as the original image and we manually select five squared windows of side 30 pixels at some corners. Figure 1.9 shows the original and the blurred *house* image (using PSF with direction 30 degrees and length 25 pixels) and the analyzed regions. Figure 1.10 shows two vectors in pixel coordinates, the estimated $\tilde{v}$ (dashed line) and the vector having true motion parameters (solid line), for each selected region. Table 1.3 shows distance

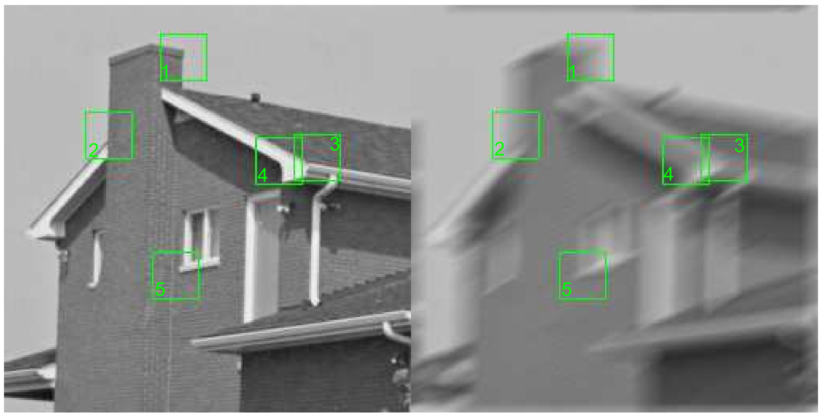


Figure 1.9.: Original and blurred house image. Blur has direction of 30 degrees and 25 pixels length, analyzed regions are numbered.

between the endpoints of the two vectors.

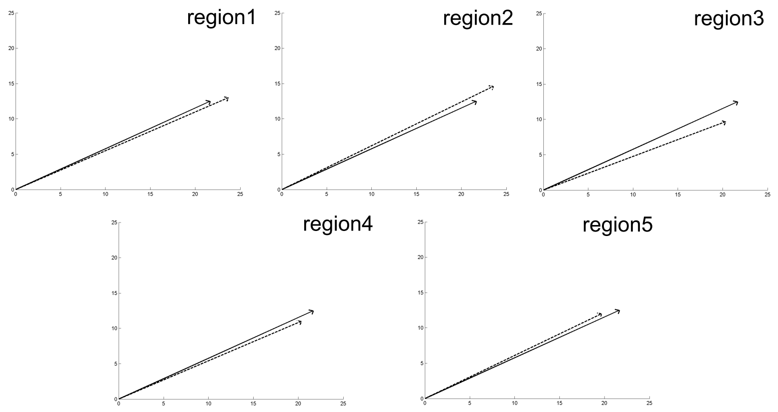


Figure 1.10.: Displacement vectors $\tilde{v}$ estimated in selected regions of synthetically blurred *house* test image. The solid line is the true displacement vector, while the dotted line represents the estimated displacement vector $\tilde{v}$.

1.6.3. Camera Images

We perform a second experiment using a sequence of camera images, captured according to the following scheme

- a still image, at the initial camera position.

σ_η	R 1	R 2	R 3	R 4	R 5
0	2.07	2.75	3.19	1.87	2.04
0.01	0.32	6.91	3.52	2.64	4.58

Table 1.3.: Estimation error: distance between $\tilde{v}$ endpoint and displacement vector, expressed in pixels, on each image region R

- a blurred image, captured while the camera was moving.
- a still image, at the final camera position.

We estimated the blur parameters at some manually selected corner regions in the blurred image and we compare these results with the ground truth, given by matching corner found by the Harris detector in the images taken at the initial and at the final camera position. Clearly, the accuracy obtained in motion estimation from a single blurred image is lower than that obtained with methods based on two well focused views. However preliminary results show good accuracy. For example, motion parameters estimated in region 2 of Figure 1.17 are particularly close to the computed ground truth, even if the corner is considerably smooth, as it is taken from a common swivel chair.

Figures 1.12 - 1.16 illustrates the algorithm results at the selected image regions. As Figure 1.13 shows, the votes in parameter space are more spread around the solution than in Figure 1.12, where the corner is close to the model of Section 1.3.2. Table 1.4 shows result using the same criteria of Table 1.3.

Results are less accurate than in previous experiments because according to experimental settings, motion PSF could not be perfectly straight or perfectly uniform, because of camera movement. This affects the algorithm performance.

R 1	R 2	R 3	R 4	R 5
0.44	1.90	1.09	3.95	3.75

Table 1.4.: Estimation error expressed in pixel unit on each image region R.

1.7. Remarks

The results obtained from the experiments, performed both on synthetic and camera images, show that the image at blurred corners has been suitably modeled and that the solution proposed is robust enough to cope with artificial noise and to deal with camera images.

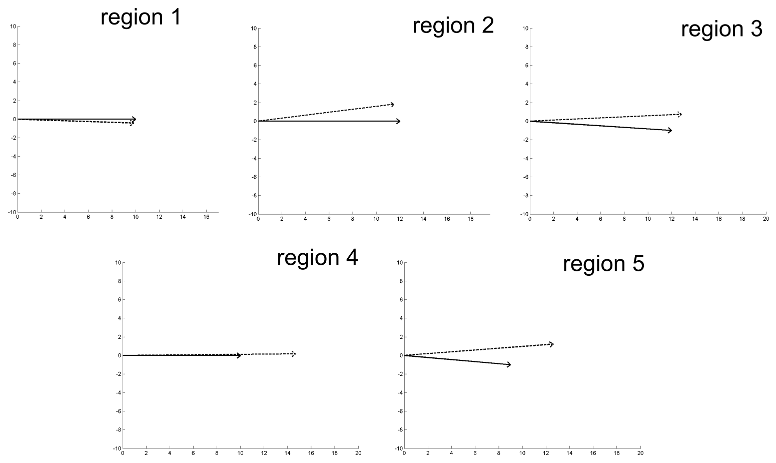


Figure 1.11.: Displacement vectors $\tilde{v}$ estimated in camera images. In each plot, the solid line indicates the true displacement vector obtained by matching corners of pictures at initial and final camera position. Dotted line represents the estimated displacement vector $\tilde{v}$.

The results show also that there are only a few useful corners in camera images. This is mostly due to background and corner non uniformity caused by shadows, occlusions or because the original image itself shows significant intensity variations. Another motivation is that only corners satisfying assumption of Equation (1.6) are considered, while the others are discarded.

In the next section, we will present an extension of this algorithm which is able to estimate the displacement vector at corners which are moving like Figure 1.3 **b**, i.e. corners that do not satisfy assumption (1.6). This will be possible thanks to a procedure to discriminate whether the corner has a "self-intersection" because of image blur (like those in Figure 1.3 **a** and Figure 1.1, region D), or not (like Figure 1.3 **b** and Figure 1.1 region A).

1.8. The Orientation Ambiguity

Let now consider a squared region A centered in a local maximum of Harris measure [35], see Figure 1.1. The motion of the image corner during the exposure is described by the corner displacement vector $\tilde{v}$. However, since there is no way to determine which is the initial and which is the final corner position from a single blurred image, the corner displacement vector can be determined only up to its orientation. Therefore its direction θ has to be considered up to π: in what follows we assume $\theta \in [0, \pi]$.

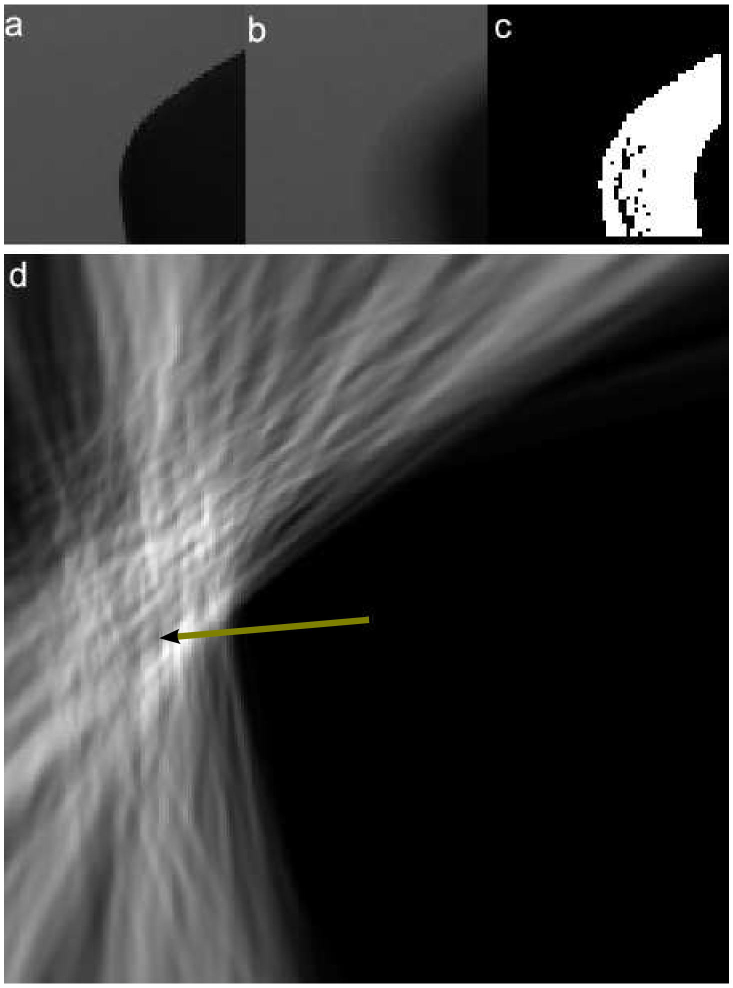


Figure 1.12.: Figure **a** Original corner in image **b** blurred corner, **c** set A_0 of considered pixels and **d** votes in the space parameter

The same orientation ambiguity holds for the projections P of vector $\tilde{v}$ on directions orthogonal to the corner edges (1.14). Let $x_a \in A_0$ be a pixel belonging to a blurred edge, see Figure 1.21. Let denote by $P(x_a)$ the projection of $\tilde{v}$ on the gradient vector at x_a, i.e.

$$P(x_a) = \left[\tilde{v} \cdot \frac{\nabla z(x_a)}{||\nabla z(x_a)||} \right] \frac{\nabla z(x_a)}{||\nabla z(x_a)||}\,, \tag{1.18}$$

where $\cdot$ denotes the scalar product. Figure 1.19 shows the projection vectors in the blurred corner model.

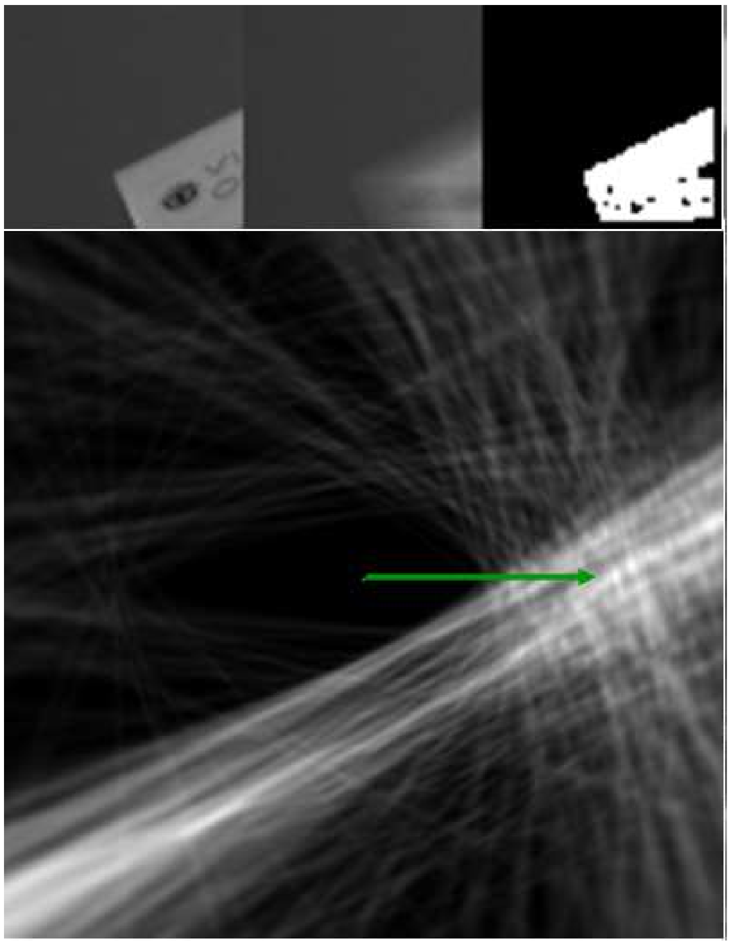


Figure 1.13.: Figure **a** Original corner in image **b** blurred corner, **c** set A_0 of considered pixels and **d** votes in the space parameter

By relaxing assumption (1.6), also corners like those of Figure 1.3**b** has to be taken into account. For these corners, the angle between the gradient at blurred edges $\nabla K\left(y\right)\left(x\right)$ and the displacement vector $\tilde{v}$ is larger than $\pi/2$, as shown in Figure 1.20. Therefore the scalar product $\nabla K\left(y\right)\left(x\right)\cdot\tilde{v}$ could assume negative values. Therefore Equation (1.8) becomes

$$\nabla K\big(y(x)\big)\cdot\tilde{v} = \begin{cases} 0, & \text{if} \quad \nabla K\left(y\right)\left(x\right) = 0 \\ \pm\Delta, & \text{otherwise} \end{cases} \quad ,\forall x \in A\,. \tag{1.19}$$

Assuming that $x_a \in A_0$ (and thus $\nabla K\left(y\right)\left(x\right) \neq 0$) and substituting Equation (1.19) into the right-hand side of Equation (1.18), we have that the projec-

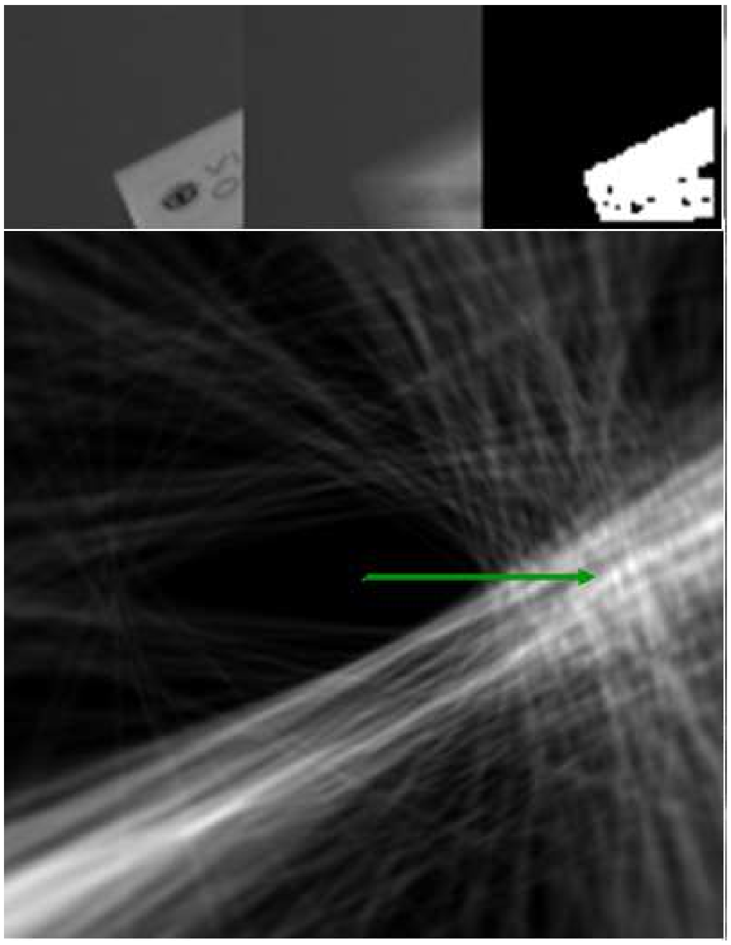


Figure 1.14.: Figure **a** Original corner in image **b** blurred corner, **c** set A_0 of considered pixels and **d** votes in the space parameter

tion of $\tilde{v}$ along $\nabla z(x_a)$ direction is

$$P(x) = \pm \frac{\nabla z(x)}{||\nabla z(x)||^2} \Delta \,, \quad \forall x \in A_0 \,. \tag{1.20}$$

It follows that $\tilde{v}$ cannot be trivially computed whenever two projections along two different directions are available, like in the previous case. In fact let $x_b \in A_0 \,,\; x_b \neq x_a$, then if $\nabla z(x_a)$ and $\nabla z(x_b)$ are linearly independent, four displacement vectors $\pm\tilde{v}_1$ and $\pm\tilde{v}_2$ are identified by back-projection of the four vectors $\pm P(x_a)$ and $\pm P(x_b)$, as illustrated in Figure 1.22. These four displacements vectors $\pm\tilde{v}_1$ and $\pm\tilde{v}_2$, are indeed two pairs having the same directions and opposite orientations. As pointed out before, there is no way to

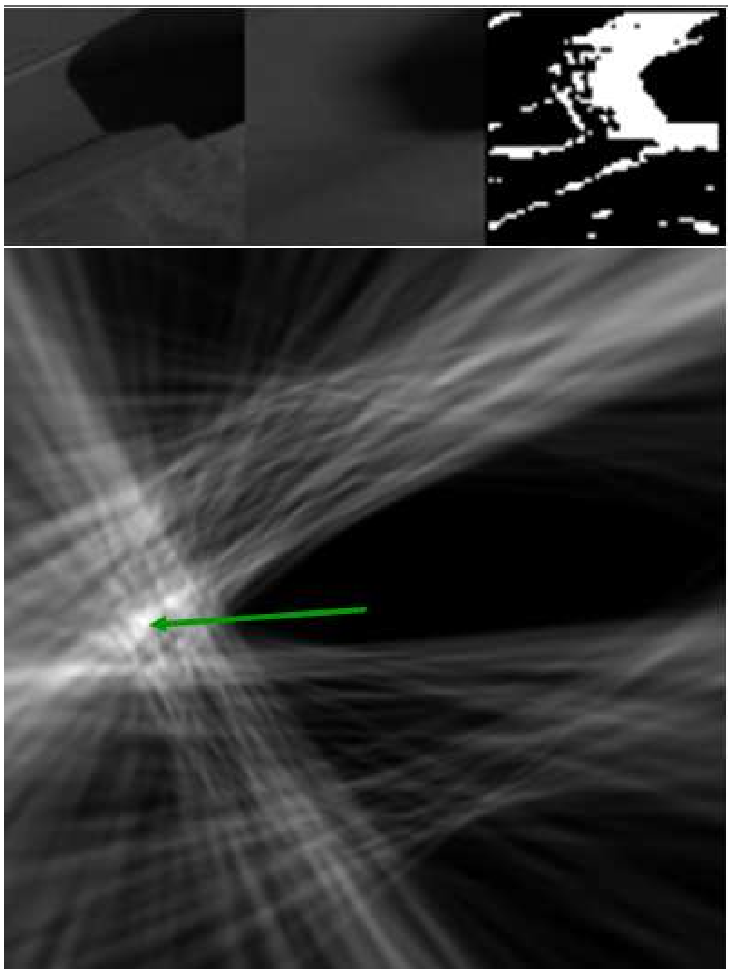


Figure 1.15.: Figure **a** Original corner in image **b** blurred corner, **c** set A_0 of considered pixels and **d** votes in the space parameter

exploit motion orientation from a single image so that we can assume the corner displacement direction $\theta \in [0, \pi]$ and reduce to consider only $\tilde{v}_1$ and $\tilde{v}_2$, whose directions $\theta_1, \theta_2 \in [0, \pi]$.

When $\nabla z(x_a)$ and $\nabla z(x_b)$ are linearly dependent, the motion direction cannot be estimated. This happens when both x_a and x_b belong to the same blurred edge, where all gradient vectors have the same direction. This situation happens at blurred edges, as illustrated in Figure 1.1, regions B and C.

Therefore in a blurred corner region, up to an orientation swap, there are two admissible displacement vector $\tilde{v}_1$ and $\tilde{v}_2$. As Figure 1.23 shows, a single

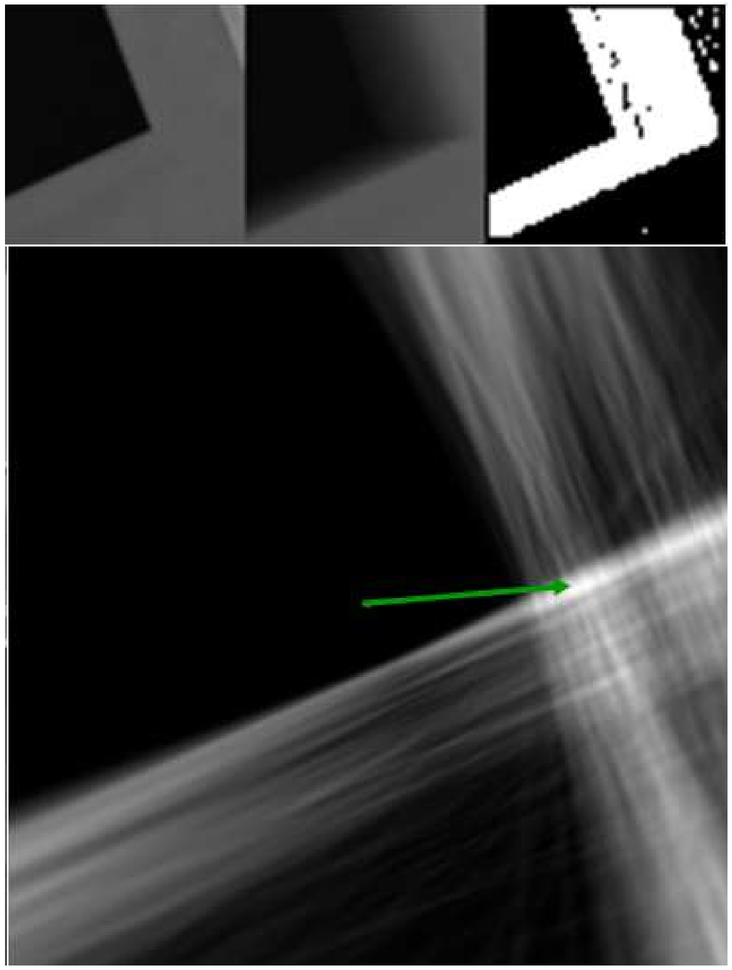


Figure 1.16.: Figure **a** Original corner in image **b** blurred corner, **c** set A_0 of considered pixels and **d** votes in the space parameter

binary corner, blurred with two different PSFs (whose parameters are given by $\tilde{v}_1$ and $\tilde{v}_2$), may present the same blurred edges. In order to disambiguate which is the correct corner displacement vector then, we devised a *decision function*, presented in Section 1.10.

1.9. Selection of Best Projection Vectors

Consider an image region A containing a binary corner and assume no noise, i.e. $\eta = 0$ and $\xi = 0$. Denote by $\nabla K\big(y\big)(x_a)$ and $\nabla K\big(y\big)(x_b)$ the gradient

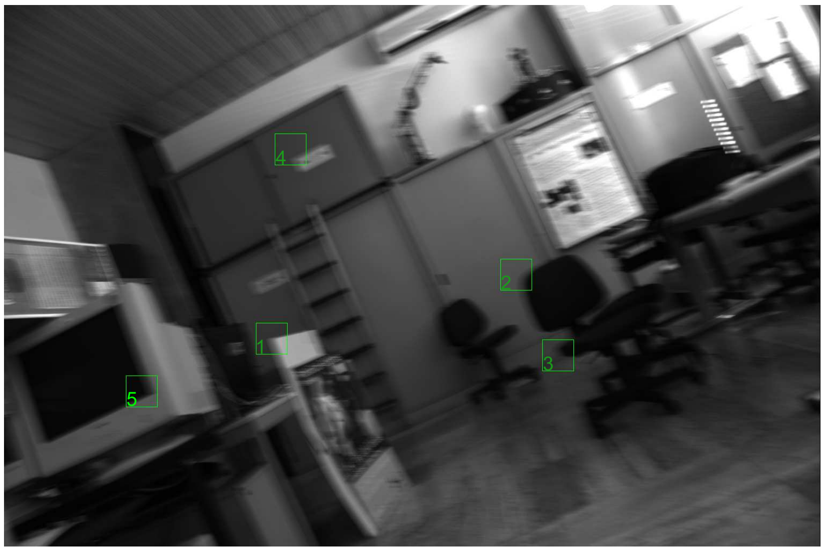


Figure 1.17.: Laboratory Image and selected regions

vectors at pixels x_a and x_b belonging to the two blurred edges, as shown in Figure 1.21, and denote by $P(x_a)$ and $P(x_b)$ the projections of $\tilde{v}$ along their directions. When the considered image region A is large enough, the highest peaks in the 2-D histogram of $\{\nabla z(x_i)\}_{x_i \in A_0}$ represent the end points of $\nabla K\big(y\big)(x_a)$ and $\nabla K\big(y\big)(x_b)$, then $P(x_a)$ and $P(x_b)$ are promptly obtained from Equation (1.19).

Let now examine how η and ξ affects vectors $\nabla z(x)$. If η and ξ are white noise, according to (1.7) we have that

$$\nabla z(x) = \begin{bmatrix} (z \circledast d_1)(x) \\ (z \circledast d_2)(x) \end{bmatrix} = \begin{bmatrix} ((K\ (y+\xi)+\eta) \circledast d_1)(x) \\ ((K\ (y+\xi)+\eta) \circledast d_2)(x) \end{bmatrix} \forall x \in \mathcal{X}. \tag{1.21}$$

Let h be the PSF which is approximating the blur in the selected corner region. The noises η and ξ are transformed by the blur operator and by the image gradient in the following way,

$$\nabla \eta(x) = \begin{bmatrix} (\eta \circledast d_1)(x) \\ (\eta \circledast d_2)(x) \end{bmatrix}, \nabla \xi(x) = \begin{bmatrix} ((\xi \circledast h) \circledast d_1)(x) \\ ((\xi \circledast h) \circledast d_2)(x) \end{bmatrix}, \forall x \in \mathcal{X}. \tag{1.22}$$

Therefore it follows that $E[\nabla \eta + \nabla \xi] = 0$ and thus the mean of all gradient vectors for pixels *belonging to the same blurred edge* is an unbiased estimator

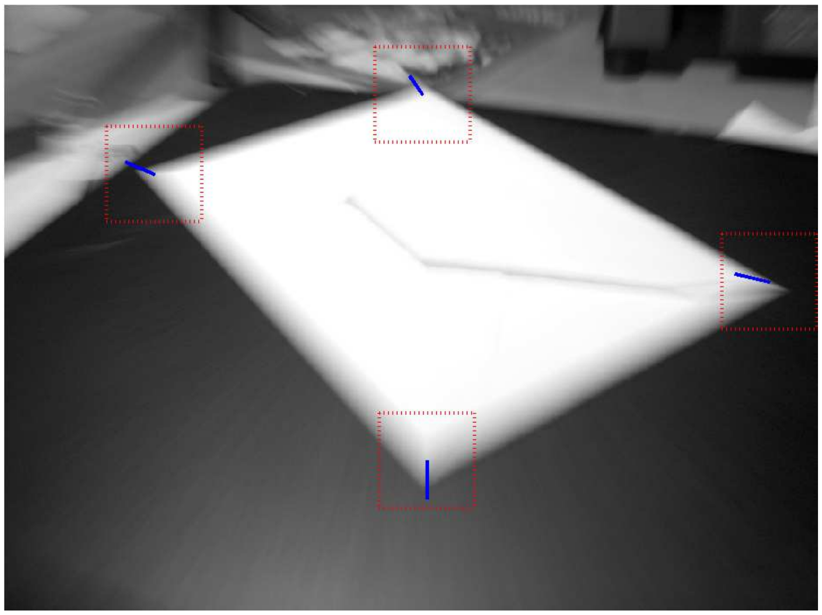


Figure 1.18.: Algorithm results on a picture taken from a hand held camera

for $\nabla K\left(y\right)\left(x_a\right)$ or $\nabla K\left(y\right)\left(x_b\right)$.

Figure 1.24 presents the 2D histogram of $\nabla z(x), x \in A_0$. There are two clusters, clearly distinguishable, as the gradient vectors are orthogonal to the corner edges. If we separate the gradient vectors in these two clusters, and then we average the gradient vectors on each cluster, we obtain unbiased estimates of $\nabla K\left(y\right)\left(x_a\right)$ and $\nabla K\left(y\right)\left(x_b\right)$. A review on clustering can be found in [43].

Moreover, when η and ξ are Gaussian distributed, also $\nabla\eta$ and $\nabla\xi$ are Gaussian distributed. Then, the two most frequent gradient vectors on each cluster of the 2D histogram can be taken as in the Gaussian case these are also unbiased estimates of $\nabla K\left(y\right)\left(x_a\right)$ and $\nabla K\left(y\right)\left(x_b\right)$. It is therefore possible to avoid clustering by imposing a minimum angular distance between the two highest peaks in the 2D histogram of all the gradients (as the angle between the two projections of $\tilde{v}$ along edges directions is proportional to the angle at corner).

Once $\nabla K\left(y\right)\left(x_a\right)$ and $\nabla K\left(y\right)\left(x_b\right)$ have been estimated, Equation (1.19) gives $\pm P(x_a)$ and $\pm P(x_b)$, and thus $\tilde{v}_1$ and $\tilde{v}_2$ are obtained, see Figure 1.22.

For corners such as the one of region D in Figure 1.1 there is a third cluster of vectors in the histogram, corresponding to the gradient in the triangular shaped area between the corner blurred edges. When the considered corner

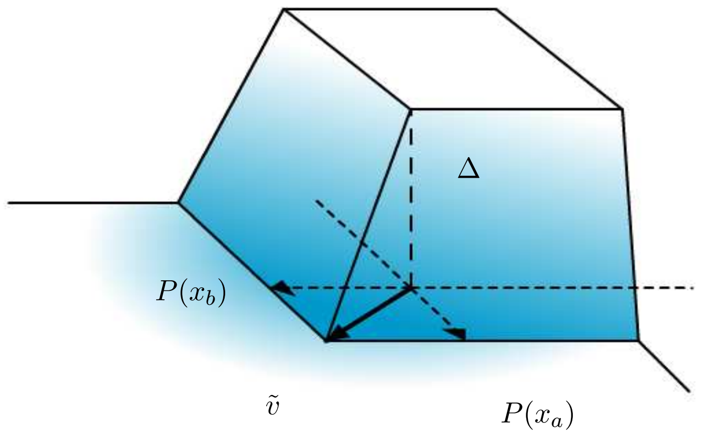


Figure 1.19.: Blurred Corner Model: Mesh of pixel intensities within region A of Figure 1.1. The projection vectors $P(x_a)$ and $P(x_b)$ are also shown.

region is large enough, the number of pixels belonging to each of the two blurred edges is larger than the number of pixels in the triangular area. This should not be taken as a projection vector. However, when the corner region has not been accurately located around the corner, this triangular area may yield uncorrect estimates.

1.10. Decision Function

Due to the fact that $\tilde{v}$ orientation is unknown, both orientations of $P(x_a)$ and $P(x_b)$ have to be considered so that there are two possible solutions, $\tilde{v}_1$ and $\tilde{v}_2$ (see Figure 1.22). The decision function determines which one, between $\tilde{v}_1$ and $\tilde{v}_2$, is, up to its orientation, the true displacement vector.

Loosely speaking, blurred corners can be divided in two classes, according to the presence of an area where gradients are orthogonal to the true displacement vector. The first class contains corners like the one represented in Figure 1.1 region D and in Figure 1.23 **b**. This class of corners shows an area whose pixels belong to the set $Z_i = \{x \in A_0 \, , \, \nabla z(x) \perp \tilde{v}_i\}$. An example of corners of the second class is reported in Figure 1.1 region A and in Figure 1.20 **a**. For a binary corner of the second class $Z_i = \emptyset \, , i = 1$ or $i = 2$ holds and therefore the number of pixels having gradient orthogonal to each candidate displacement vector (i.e $\#Z_i$) is taken as a discriminant between the two classes.

For a binary corner of the first class, $\#Z_i$ corresponds to the surface of a triangle between the two blurred edges, whose value in the ideal case S_1 (resp. S_2) can be calculated from $\tilde{v}_1$ (resp. $\tilde{v}_2$) and $P(x_a)$ and $P(x_b)$. If $\#Z_1$

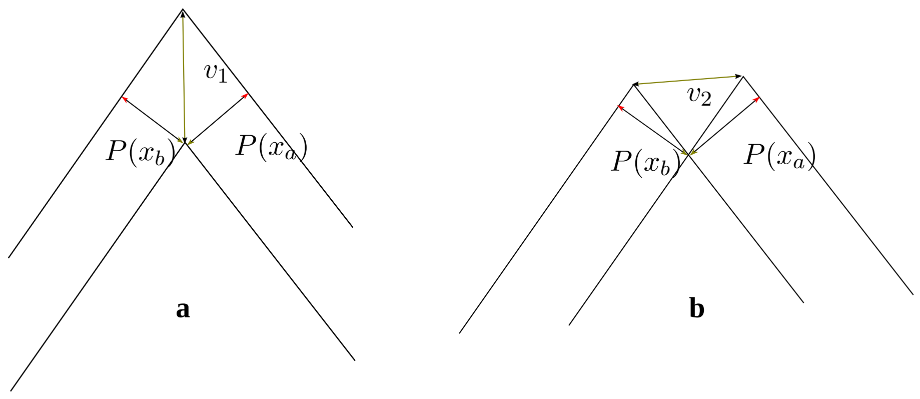


Figure 1.20.: Two different kind of corners: in the first case (**a**), the inner product in Equation (1.19) gives $+\Delta$, while in the second case (**b**) gives $-\Delta$. In fact in case **a** the angle between $\tilde{v}$ and $P(x)$ is smaller than $\pi/2$ while in case **b** this angle is larger than $\pi/2$.

(resp. $\#Z_2$) corresponds to S_1 (resp S_2), then $\tilde{v}_1$ (resp. $\tilde{v}_2$) is taken as the true displacement vector.

The condition $\nabla z(x) \perp \tilde{v}_i$, in the definition of the set Z_i is relaxed in order to manage camera images and is replaced by

$$\# \left\{ x \in A_0 \, , \, \frac{\nabla z(x) \cdot \tilde{v}}{|\nabla z(x)||\tilde{v}|} < t \right\} < S_i \quad i = 1, 2 \, , \tag{1.23}$$

where t represent the cosine of a threshold angle between the two vectors. Whenever both $\tilde{v}_1$ and $\tilde{v}_2$ satisfy (1.23), the one having the largest value in left side of (1.23) is taken.

Whenever both $\tilde{v}_1$ and $\tilde{v}_2$ do not satisfy (1.23), the corner belongs to the second class. In this case, the derivative along motion direction is constant in A_0, i.e $\nabla z \cdot \tilde{v} = const \ \forall x \in A_0$. This yields $+\Delta$, or $-\Delta$ in Equation (1.19), and the signum does not change in the region. Then, the histograms of directional derivatives along directions of $\tilde{v}_1$ and $\tilde{v}_2$ are computed and the more peaked one is selected. The sample kurtosis is taken as peakedness measure.

Finally, in order to obtain a reliable estimate of the motion extent, an accurate estimate of $\Delta = |b - c|$ is required, as Δ scales both $P(x_a), P(x_b)$, as illustrated Figure 3.5(a). The value of Δ is computed as the intensity difference between the two highest local maxima in the histogram of image intensities in the corner region. Since there should be a clear difference between b and c, a minimum distance of half of the intensity range in the region is required.

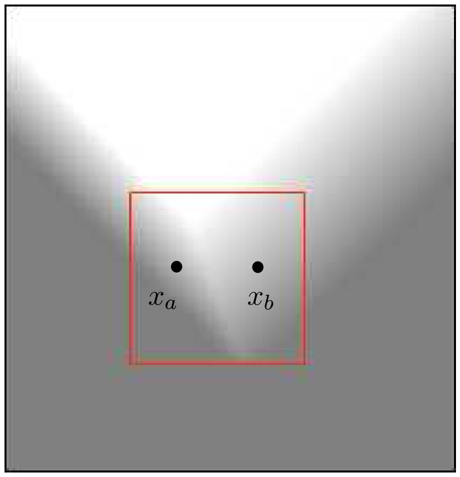


Figure 1.21.: Blurred corner region: pixels x_a and x_b belong to two different blurred edges.

1.11. Experiments

The algorithm based on the procedure for select the best projection vectors described in Section 1.9 and the decision function described in Section 1.10 has been tested on synthetic images, on a test image synthetically blurred and on a camera image.

1.11.1. Synthetic Images

Synthetic images have been generated according to Equation (1.7), with an original image y according to the binary corner model of Section $1.3.2$ having an angle of $90, 60$, and 45 degrees. The original image is constantly 0 at background and 255 at corner pixels. Blur is produced by a convolution against a PSF having extent $l \in \{20, 30, 40\}$ pixels, and direction $\theta \in \{0, 15, 75, 90\}$ or $\theta \in \{0, 20, 60, 80\}$ according to corners edges orientation.

For each value of blur direction and extent a squared region of 100 pixels, taken around the Harris measure maximum (see Figure 1.25), has been analyzed. Images have been corrupted by noise ξ with standard deviation $\sigma_\xi \in \{4, 8, 12, 16\}$ and by η with standard deviation $\sigma_\eta \in \{1, 2, 3, 4\}$, according to Equation (1.7).

Values reported Tables 1.11.1 - 1.7 are $\|v - \tilde{v}\| / \|v\|$ i.e. the distance, in pixels, between the estimated displacement vector $\tilde{v}$, and the ground truth v, expressed as a percentage with respect to true motion extent. Results have

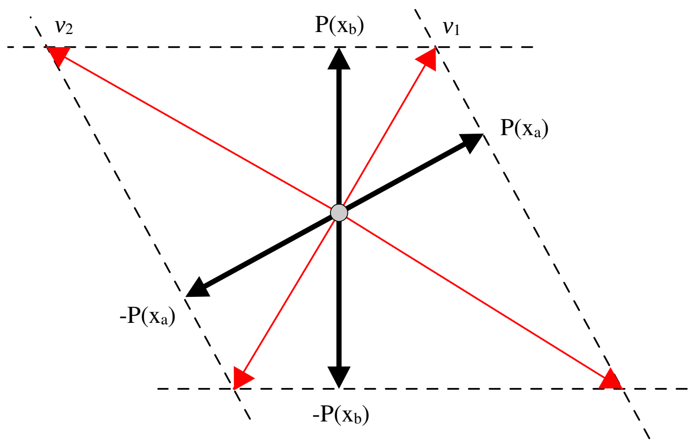


Figure 1.22.: $\pm P(x_a)$ and $\pm P(x_b)$ individuate via back-projection four corner displacement vectors $\pm\tilde{v}_1$ and $\pm\tilde{v}_2$.

been averaged on 10 realization of η for each value of σ_η and on all directions and extents.

As σ_η, and σ_ξ increases, the decision function may fail to select the true displacement vector: this occurred in about 2.3% of cases.

1.11.2. Experiment on a Test Image

The well known *cameraman* test image has been synthetically blurred by a convolution against a PSF direction 35 degrees and length 15 pixels. Squared regions of 40 pixels centered in every corner selected by Harris corner detector have been analyzed with our method.

Figure 1.26 shows the blurred cameraman image and the corner displacement vectors estimated. The dashed regions surrounding some of them are the regions where the estimated displacement vector $\tilde{v}$, satisfies $|\tilde{v} - v| < 2$ in pixel unit, being v the ground truth identified by the PSF parameters. The average error in the correct matches is 0.71 pixels. The algorithm results are accurate in regions containing a corner satisfy the model presented in Section $1.3.2$. The regions where the algorithm fails do not contain a binary corner.

1.11.3. Experiment on Camera Images

A triplet of camera images have been captured according to the following scheme. First a still image at the initial camera position is taken, followed by a blurred image captured while moving the camera during the exposure. At the end of the exposure, another still image at the final camera position, is

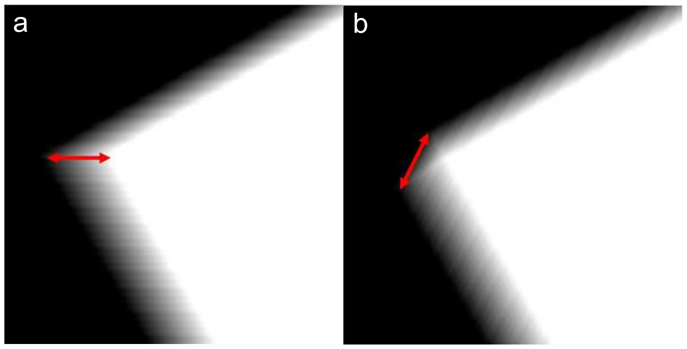


Figure 1.23.: The same binary corner blurred with two different displacement vectors, $\tilde{v}_1$ and $\tilde{v}_2$. Their blurred edges coincide.

taken. In this way the algorithm performance on a real motion blurred image is be compared with the ground truth obtained by matching the two still images. Again, the corners have been selected by local maxima of Harris measure and a region of 50 pixels around each of them have been analyzed.

Figure 1.27 shows the blurred camera picture and the regions where $|\tilde{v} - v| < 7$ in pixel unit, being v the ground truth computed by feature matching [86] in the corresponding region in the two still images. The error $|\tilde{v} - v|$ averaged on all these regions is 4.84 pixels. The regions where the displacement vectors are marked in red, represent regions where the decision function discards the displacement vector closer to the true displacement vector (this happens 4 times over 17).

1.12. Conclusions

The experiments show that the blurred corners have been suitably modeled and that is possible to estimate the blur even in a small image region containing a corner. The algorithm can be used for estimating the optical flow from a single blurred image as in [75, 76]. Estimating the optical flow at corners is advantageous as the blur is analyzed only at some significant region, and not on a fixed image tessellation that covers the whole image. On a fixed tessellation, blur estimates may be biased by the image content. Moreover, we believe that spatial domain algorithms are more suited to blur parameters estimation as they do not impose restriction on region size with respect to blur extent, while Fourier transform based methods do.

In the next chapter we will address corner detection and region selection

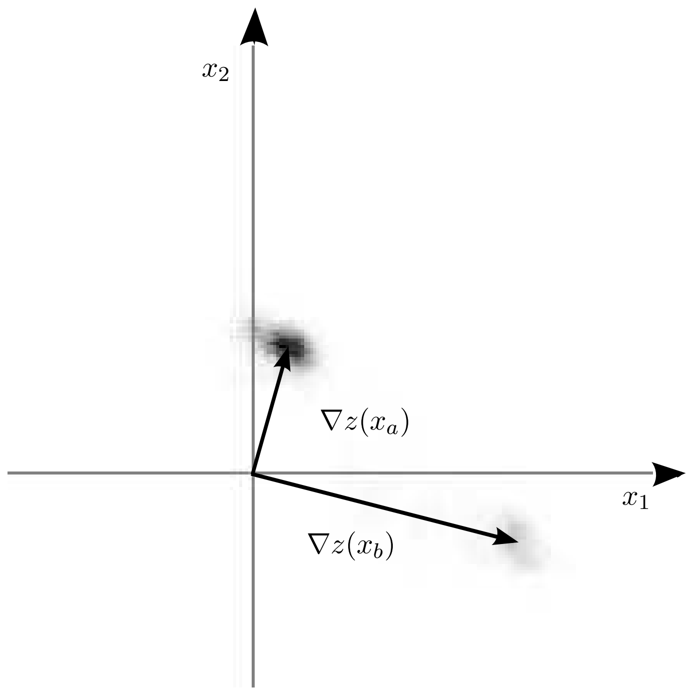


Figure 1.24.: 2d histogram of gradient.

issues, presenting a preliminary solution. The blur estimates obtained with the proposed algorithms can be used in initialization of deblurring algorithms that treat spatially variant blur, such as [89], which requires user supervision during initialization.

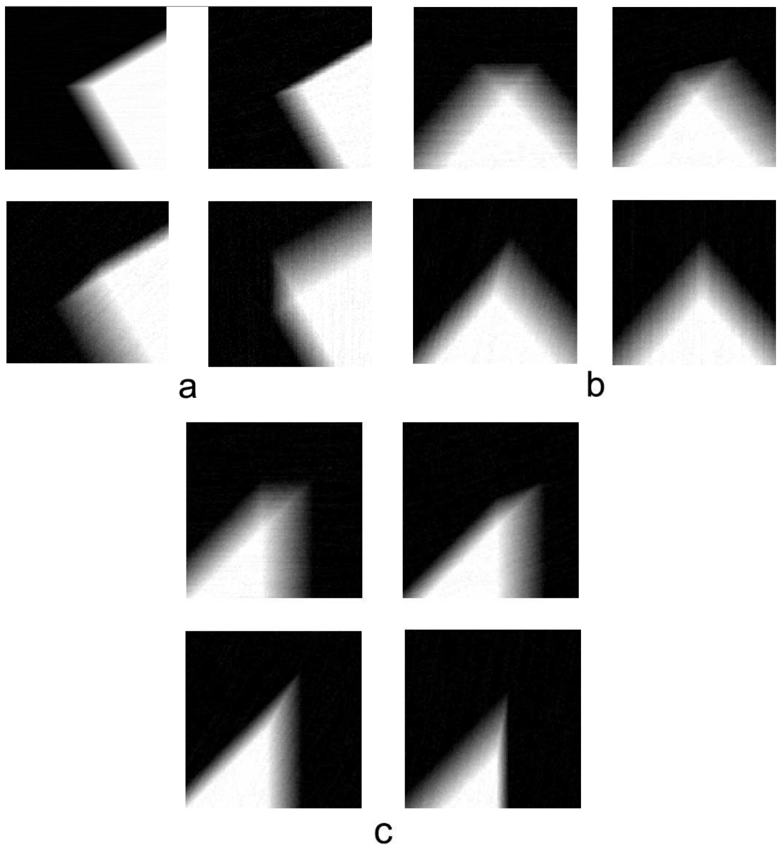


Figure 1.25.: Examples of Synthetic Test Images from dataset.

$\sigma_\eta \mid \sigma_\xi$	4	8	12	16
1	1.77 %	1.73 %	1.53 %	1.48 %
2	1.97 %	2.00 %	1.91 %	2.05 %
3	2.52 %	2.76 %	2.58 %	2.61 %
4	3.35 %	3.64 %	3.60 %	3.69 %

Table 1.5.: Result on corner of Figure 1.25**a**, $l \in \{20, 30, 40\}$ pixels, $\theta \in \{0, 15, 75, 90\}$

$\sigma_\eta \mid \sigma_\xi$	4	8	12	16
1	2.50 %	2.33 %	2.31 %	2.66 %
2	2.71 %	2.77 %	2.86 %	3.13 %
3	3.44 %	3.61 %	3.82 %	3.75 %
4	4.89 %	4.39 %	4.84 %	5.03 %

Table 1.6.: Result on corner of Figure 1.25**b**, $l \in \{20, 30, 40\}$ pixels, $\theta \in \{0, 15, 75, 90\}$

$\sigma_\eta \mid \sigma_\xi$	4	8	12	16
1	1.87 %	1.9 %	1.83 %	1.86 %
2	1.97 %	1.9 %	1.88 %	1.94 %
3	2.53 %	2.3 %	2.27 %	2.29 %
4	3.18 %	3.1 %	2.99 %	3.12 %

Table 1.7.: Result on corner of Figure 1.25**c**, $l \in \{20, 30, 40\}$ pixels, $\theta \in \{0, 20, 60, 80\}$

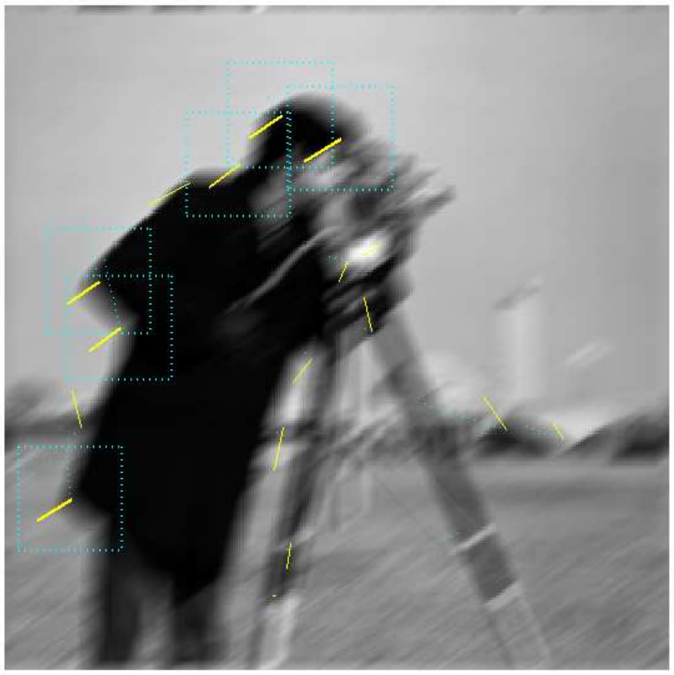


Figure 1.26.: Cameraman synthetically blurred.

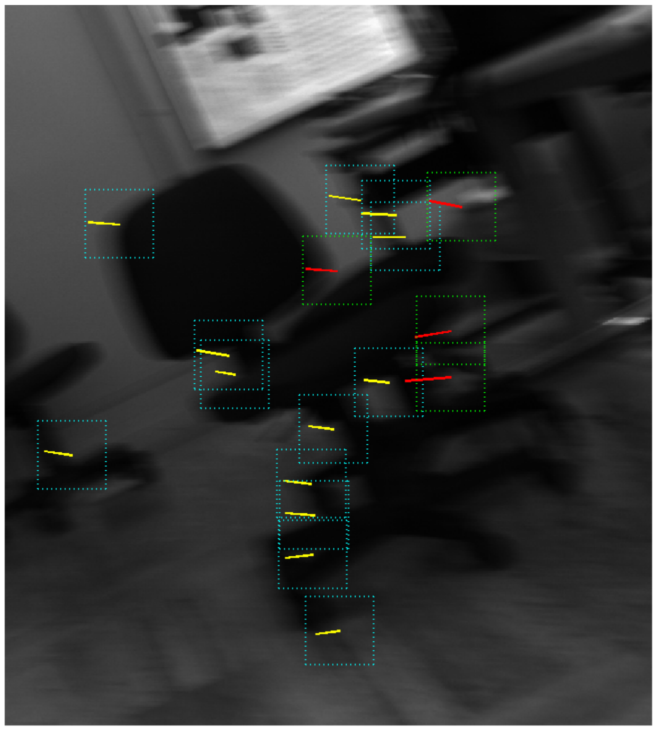


Figure 1.27.: Test on Camera Image.

2. Corner Detection and Adaptive Region Selection

In the previous chapter we presented algorithms which estimate blur parameters in regions containing corners. In this chapter we address two related issues: the detection of blurred corners and the selection of a region around each of them.

We propose two "naive" solutions, based on two intuitive ideas: blurred corners are detected combining the well known Harris measure [35] with a mask obtained by thresholding the gradient magnitude. The corner region selection procedure, figures out the blurred edges of each detected corner. In fact, the algorithm of Section 1.9 exploits only pixels belonging to the blurred edges for determining the projection vectors and then the corner displacement vector. Thus, the selected corner region may possibly contain only pixels belonging to the corner blurred edges. Other pixels, for example those on the background or pixels on blurred areas not belonging to the corner, have not be considered, as these could bias the estimators of Chapter 1.

Blurred corner detector and adaptive region selection procedure, combined with the algorithm of the previous chapter allow to process a single blurred image without any user interaction, and to estimate the optical flow by exploiting blurred corners.

2.1. Blurred Corner Detection

Salient points in images are often extracted from the local maxima of the Harris measure [35, 59, 65]. At pixels having large Harris measure, the Hessian matrix of the sum of square differences function [35] has two large eigenvalues. Therefore the image in a patch of these pixels is significantly different from any neighboring patch. On the contrary, the Harris measure is zero when a directional derivative is zero.

In a blurred corner region, the Harris measure is larger on the corner smears than on the blurred edges. By corner smears we indicate the set of pixels between the two blurred edges, corner smears have been shaded in Figure 2.1

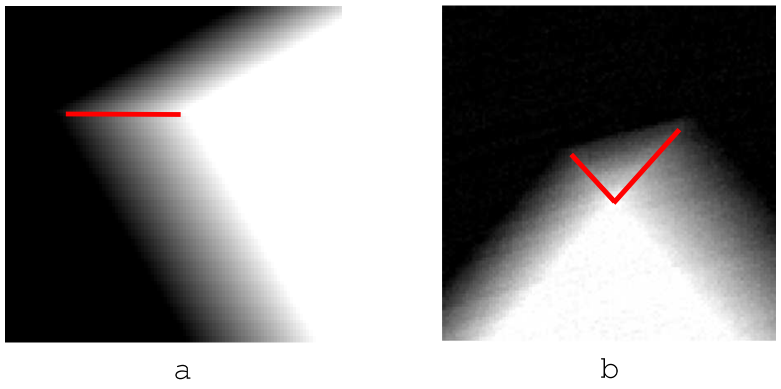


Figure 2.1.: The corner Smears, pixels shaded belongs to the corner smears.

for both an example of corner of the first class and of the second class. At corner smears the image changes in any neighboring patch, and thus the Harris matrix has two nonzero eigenvalues. On the blurred edges the Harris measure is zero as the derivative along the edge direction is everywhere null. Therefore, provided that in the original image the corner is binary [5] (i.e. the image is constant in the corner and in the background area), the Harris measure has a local maximum that belongs to each corner smear. This maximum could be any pixel of the corner smear but the adaptive corner region selection procedure does not require higher accuracy. Figure 2.3 shows the Harris measure of the image depicted in Figure 2.2, containing some blurred corners.

In order to extract blurred corners and exclude possible details in still areas (the image may be not uniformly blurred), we consider only the local maxima of Harris measure belonging to the mask Γ defined as follows

$$\Gamma = \{x \; s.t. \; ||\nabla I(x)|| > T\}, \tag{2.1}$$

where T is a threshold that has to be tuned considering the minimum acceptable slope for the blurred edge or the noise standard deviation. Image noise can be estimated using any method described in Appendix 7. Then Γ is post processed by using ordinary morphological operators [32] in order to remove isolated points, small areas, thin lines and for widening larger areas. Figure 2.4(b) shows the mask Γ for the blurred corner of Figure 2.4(a). Therefore we have $\Gamma = 1$ where the image contains blurred edges, while $\Gamma = 0$ at still areas close to e.

Corners are detected by computing the Harris measure on the blurred image and by selecting the local maxima. This is a standard procedure which is also exploited in several feature detection algorithms [64]. Figure 2.5(a) show the

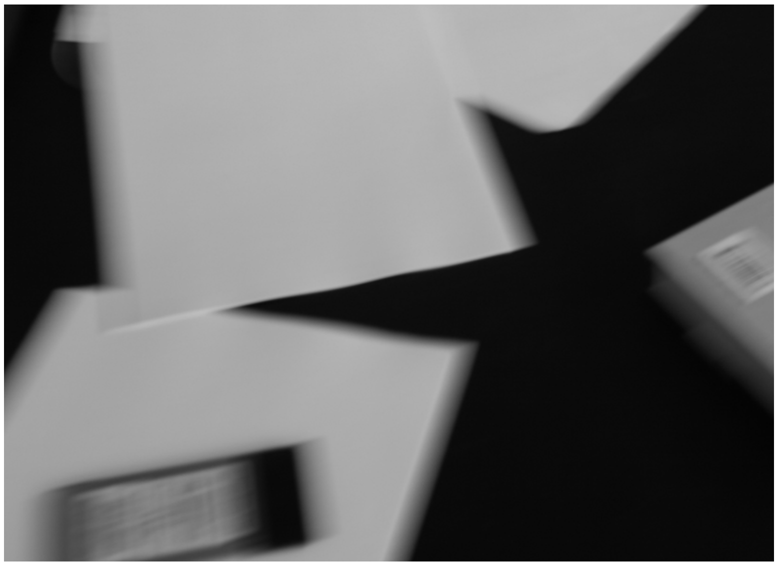


Figure 2.2.: Zoom on an area containing some blurred corners of Figure 2.4(a)
.

Harris measure for Figure 2.4(a). We estimate then, $\{x_i\}_{i=1,..,m}$ as the location of the maxima of Harris measure belonging to Γ; these pixels are supposed to be blurred corners and we run the corner region selection procedure presented in Section 2.2, determining thus the regions where to estimate the blur direction with algorithm [5]. Figure 2.5(b) shows corners selected.

The corner detection procedure is tuned by the parameter that determines the minimum distance between local maxima of Harris measure and by the morphological operators used for widening and eroding the blurred edge masks. In such a way it is possible to increase the number of detected corners in a given blurred image. Finally, it is possible to run the procedure for selecting adaptively the corner regions (described in Section 2.2) and discarding those corners presenting too small regions. Thus in the remiaining corners, the blur parameters are estimated using one of the methods presented in Chapter 1 on the selected corner regions.

2.2. Corner Region Selection Procedure

Image corners are characterized by the blurred edges, areas where the image gradient vectors are approximatively constant. We describe an iterative procedure, which is used both for selecting a data-adaptive corner region and to

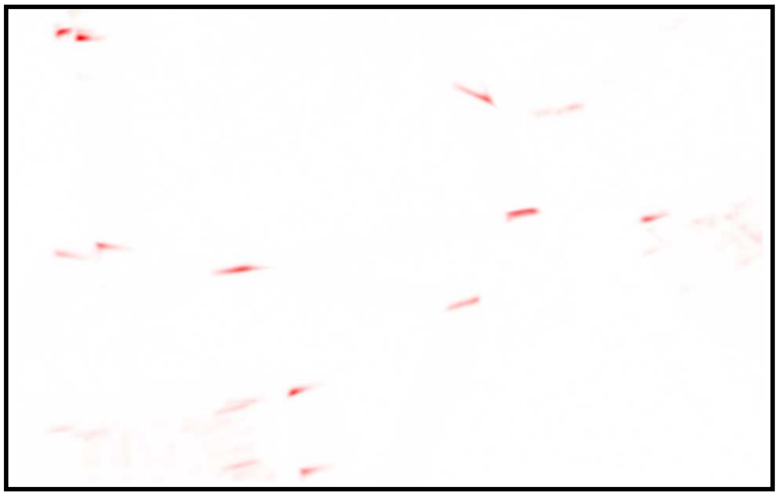


Figure 2.3.: In Red the Harris measure of Figure 2.2. The Harris measure on the corner smears is larger than in neighboring pixels.

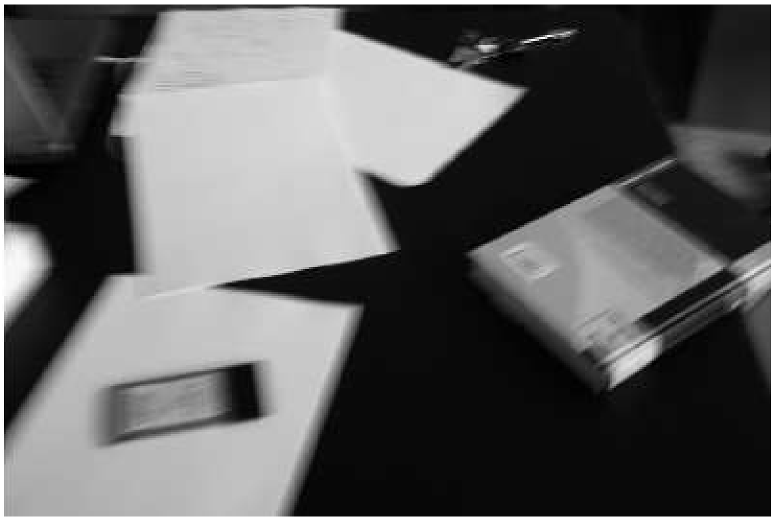
test if the selected corner location x_i $i = 1, .., N$ corresponds to a blurred corner. The basic idea is that within pixels belonging to blurred edges the image gradient are constant. Since the algorithms proposed in the Chapter 1 exploits only image values within blurred edges, in particular the last one based on projection vectors selection (Section 1.9), the adaptive corner region will contain only blurred edge area.

The adaptive corner region is build as the union of wedge shaped binary masks $W_{j,\alpha}$, where $j \in \{j_0, .., J\}$ represents the wedge sizes and $\alpha \in \{2i\pi/A\}_{i=0,..,A}$ the wedge orientation. All the masks have a vertex in the i-*th* corner, moreover masks having the same orientation are nested, i.e. $W_{j,\alpha} \subset W_{j+1,\alpha}, \forall\alpha, \forall j = 1, .., J-1$, see Figure 2.6. These wedge shaped mask has been built from the supports of Local Polynomial Approximation kernels widely exploited in [45, 47] and reference therein.

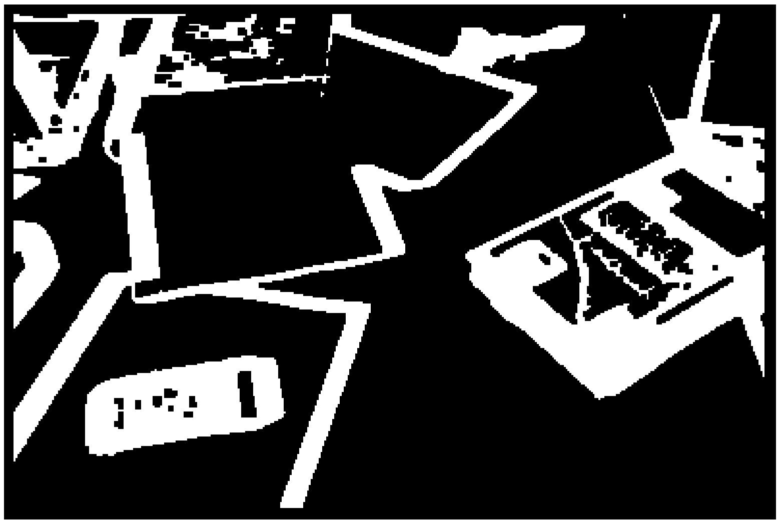
Let denote by $\nabla I = [\nabla I_1, \nabla I_2]' = [I \circledast d_1, I \circledast d_1]'$ the gradient image, where d_1 and d_2 are derivative filters along horizontal and vertical directions respectively. The procedure is repeated on each gradient component ∇I_1 and ∇I_2 separately.

The adaptive region selection is a three step iterative procedure which is repeated for each direction of wedge masks $\alpha \in \{2i\pi/A\}_{i=0,..,A}$, and starts from $j = 3$.

- Compute $w_{j,\alpha}$, the average of ∇I_1 on the support of $W_{j,\alpha}$, i.e. $w_{j,\alpha} = \sum_{x \in W_{j,\alpha}} \nabla I_1(x)/\#W_{j,\alpha}$ where by $x \in W_{j,\alpha}$ we mean that



(a) Blurred image acquired when moving a Canon EOS 400D.

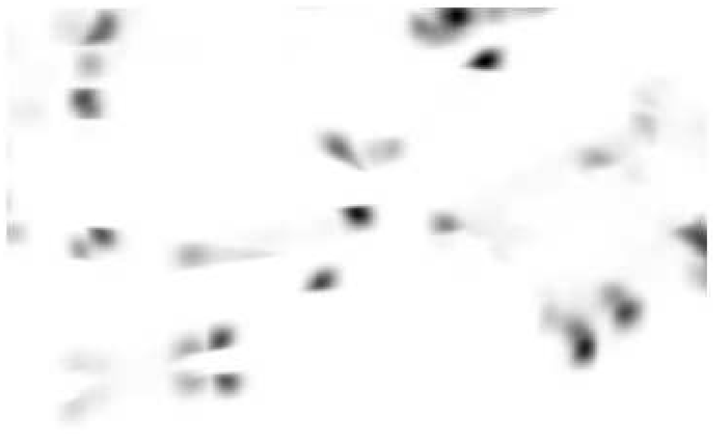


(b) Gradient mask Γ to extract blurred edges.

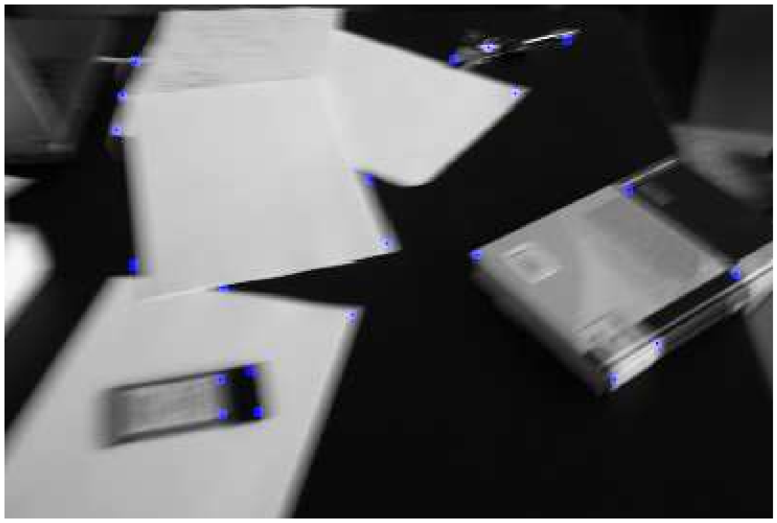
Figure 2.4.: Blurred image and blurred edge mask.

$W_{j,\alpha}(x) = 1$ and where $\#W_{j,\alpha}$ denotes the number of nonzero element in $W_{j,\alpha}$.

- Compute $d_{j+1,\alpha}$, the average of ∇I_1 on $D_{j+1,\alpha} = W_{j+1,\alpha} - W_{j,\alpha}$, the area within the two consecutively nested masks $W_{j+1,\alpha}$ and $W_{j,\alpha}$; $d_{j+1,\alpha} = \sum_{x \in D_{j+1,\alpha}} \nabla I_1(x)/\#D_{j+1,\alpha}$.



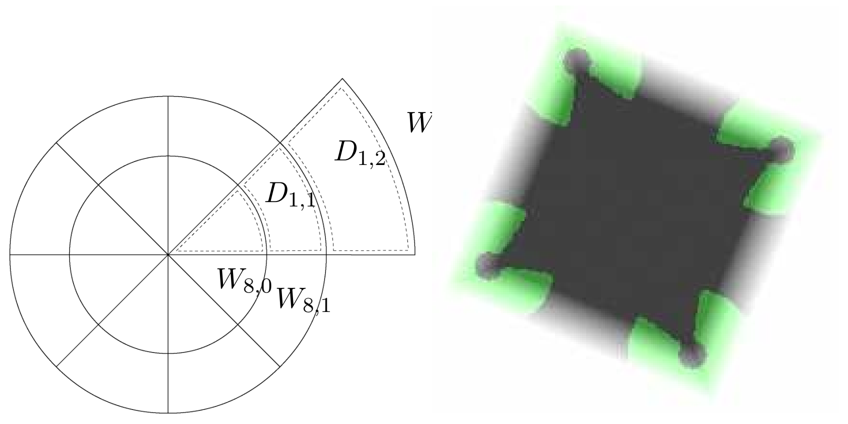
(a) Harris corner measure (darkest area indicates highest values).



(b) Corners detected combining the Harris measure with the gradient mask.

Figure 2.5.: Harris measure and detected corners.

- If $d_{j+1,\alpha}$ and $w_{j,\alpha}$ “do not differ too much”, i.e $|w_j - d_{j+1}| < M_1 \sigma_\eta$, if $d_{j+1} > M_2 \sigma_\eta$, and finally, if the largest scale has not been considered jet, the procedure is iterated from the mask $W_{j+1,\alpha}$. Otherwise the scale j is taken as $\overline{j}_\alpha$, the largest wedge size containing blurred edges along direction α.



(a) Lines delimiting the supports of wedge masks $W_{j,\alpha}$ (b) Example of Corner Regions Selected

Figure 2.6.: Corner Region Selection

Here M_1, M_2 are tuning parameters, σ_η is estimated using [20], see Appendix 7.

After having considered all directions in A, we obtain

$$U_{i,1} = \bigcup_{\alpha \in A} W_{\overline{j}_\alpha,\alpha}\,, \tag{2.2}$$

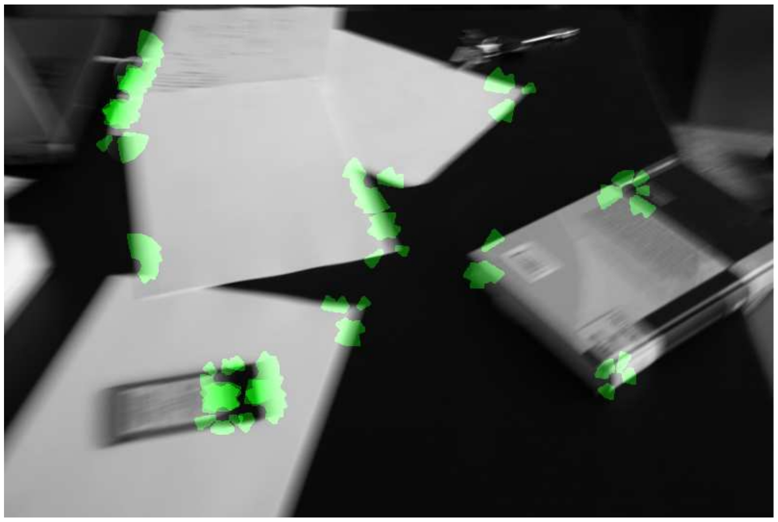
and, repeating the whole procedure on ∇I_2, we obtain $U_{i,2}$, so that the adaptive corner region selected for i-*th* corner is

$$U_i = U_{i,1} \cup U_{i,2}\,. \tag{2.3}$$

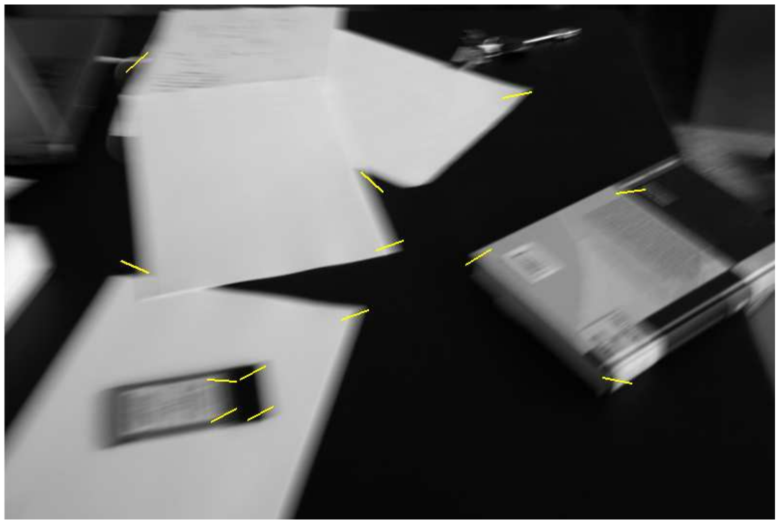
Figure 2.6(b) shows example of adaptive region selection on a synthetic image, while Figure 2.7(a) shows the adaptive regions selected around the corners detected in the blurred image of Figure 2.4(a). Figure 2.7(b) shows the displacement vectors estimated with the algorithm presented in Section 1.9.

Note that pixels in $W_{j,\alpha}, \forall\alpha, j < 3$, which are the closest to the estimated corner location, are not considered as at these pixels the gradient is typically discontinuous. Since the corner displacement vector $\tilde{v}$ is estimated only from the blurred edges, we do not affect the blur direction estimates by excluding these pixels. This results in a "hole" in the selected corner region, see Figure 2.6(b).

Figure 2.7(b) shows the corner displacement vector estimation performances on the camera image of Figure 2.4(a)



(a) Adaptive regions selected.



(b) Corner displacement vectors estimated.

Figure 2.7.: Adaptive corner region selection and corner displacement vectors estimation.

Part II.

Blur due to Camera Translation and Camera Rotation

The focus in the second part of the thesis is on the blur due to a pure camera translation or a pure camera rotation. We introduce an observation model where the blur is represented by a spatially variant operator. This allows us to correctly handle motion blurred images.

Although the blur produced by a camera translation is typically assumed spatially invariant, some blurred images acquired during camera translation show smears which proves that are PSFs are pixel-wise varying, as one can see from Figures 3.12 and 3.13. We therefore consider the most general case of a translating camera in the 3D space and we propose an observation model based on a spatially variant operator for the blur. This model has been derived by analyzing how a 3D translation of a pin-hole camera corrupts the image being acquired. In Chapter 3 and in Appendix 7 we prove that in images taken from a translating camera, independently on the captured scene, the blur PSF in each pixel has a 1D rectilinear support, and a uniform value on it. Moreover, the PSFs directions are pointing to a particular point on the image plane, the epipole e. In Chapter 3 we prove that e corresponds to the vanishing point of the camera translation direction. Then as e goes to infinity, the blur direction does not vary and the spatially invariant blur correctly describe motion blurred images when the capture scene is planar and parallel to the image plane, i.e. when the depth does not influence the image. However, when e does not go to infinity, a sort of zooming effect, the radial blur, is observed. In this case, we also refer to e as the blur center.

We address two main issues for these images. The first issue is the estimation of the coordinates of e, given a single blurred image (Chapter 3). The second issue is the restoration of radial blurred images, i.e. images acquired during camera translation toward a planar scene, parallel to the image plane (Chapter 4). Radial blur admits a global parametric description, as the location of e and a blur extent parameter ℓ determine the PSF in every image pixel. However, radial blur admits also a local parametric description, as the PSF at any pixel x_i can be expressed as a function of its direction θ_i and its extent l_i. When the scene is not planar and parallel to the image plane the PSF direction at each pixel is still determined by the epipole e. However there is no an analogous formulation for the PSFs supports as these vary according to the scene depth. The relation between the PSFs support and the scene depth can be used to reconstruct the scene depth from a single blurred image. In Chapter 5 we present some considerations about the capabilities of recovering the scene depth from a single radial blurred image and we point out some remarks about extending similar approaches to more general motions.

Finally, the blur produced by a purely rotating camera is considered in Chapter 6. This blur is also modeled similarly to the blur produced by a translating camera, with two parametric description. In fact the PSF at each image pixel is described by the parameters of an arc of conic section. These parameters are related to the camera rotation axis and the camera angular speed, which thus constitute the global blur parameters. In Chapter 6 we also present a novel algorithm for estimating the camera 3D rotation axis and its angular speed, by analyzing a single (blurred) image. Contrary to the existing methods, we treat the more general case where the rotation axis is not necessarily orthogonal to the image plane, taking into account the perspective effects that affect the smears.

3. Estimation of 3D Camera Translation Direction

Often the blur corrupting an image cannot be treated as spatially invariant because its characteristics vary considerably between different image pixels. Motion, one of the most important blurring factors, typically corrupts the images in a non-uniform way. This chapter focuses on images captured by a perspective camera that translates at constant speed in a 3-D scene. The blur in the resulting images is characterized by smears, whose directions and extents are varying. However, in this particular case, the directions of blur smears are determined by the camera displacement direction.

We devise an algorithm for estimating the camera displacement direction, given a single blurred image, exploiting these smears as a hint. The estimation is based on blur analysis of small image regions. These regions are automatically detected and two different methods for analyzing blur within them are used, according to their content. The choice of the blur analysis method improves the overall estimation accuracy. A voting procedure combines the multiple local estimates, increasing the robustness. The algorithm has been successfully tested, both on synthetic and camera images.

3.1. Motivations

Motion blurred images embody information about the motion that the camera undergoes during the acquisition. Estimating the camera motion when a single blurred image is available is very challenging, especially when the image content is unknown. We present a novel algorithm for estimating the 3D direction of a translating camera by analyzing a single blurred image, acquired during the camera motion. We derive a description of this blurring process by studing the 3D camera motion: it turns out that the resulting blur is spatially variant and characterized by rectilinear smears. Blur smears are segments whose directions are determined by the coordinates (on the image plane) of the epipole, i.e. the vanishing point of camera motion direction, from now on e. The proposed algorithm estimates the smear directions at some automatically selected image regions, using two sorts of blur direction estimators. These directions are combined in a voting procedure for determining the coordinates of e. Whenever the camera is calibrated it is then immediate to determine the viewing ray through e and thus, the camera motion direction.

As a matter of fact, blurred images often occur in vision systems, especially at reduced lightning condition and in indoor environments. The core idea of our algorithm is to exploit the blur as a cue for estimating the camera ego-motion. Thus, our motivations are similar to those in [51], where an algorithmic gyroscope based on the analysis of a rotationally blurred image is presented (other recent works on rotational blur estimation are [7, 72]). The estimation of camera ego-motion exploiting the blur is particularly attractive in case of camera translation: this motion in fact cannot be sensed by accelerometers, whereas other motions (such as the shake or the rotation) could also be handled combining these sensors. Recently, two algorithms for restoring radial blurred images have been proposed [10, 88]. Radial blur occurs when the camera translates toward a planar scene, which is assumed parallel to the image plane. The proposed algorithm can be also used for estimating the radial blur center, given a single image.

3.1.1. Related Works

The estimation of the blur Point Spread Function (PSF), assuming the blur spatially invariant, has been widely studied in the last decades [11, 52]. Recently, Fergus *et al.* [25] proposed an algorithm for photographs corrupted by camera shake, that first estimates the PSF and then deblur the image. Levin [54] deblurring algorithm is meant for images blurred by a PSF having rectilinear support. It is based on a segmentation of the image into regions having the same the blur extent, assuming the PSF direction constant on the whole image. Jia [44] introduced the use of transparency maps for estimating the PSF. All these methods are meant for deblurring, however blur estimation has been addressed for other purposes such as the integration of the optical flow for tracking system [49] , the measurement of vehicle [56] and ball speed [57] or the measurement of planar scene distance [58].

Whereas the estimation of 3D camera translation direction from a single image has never been addressed before, other works considering spatially variant blur have to be mentioned. Rekleitis [76] used the blur for computing the optical flow from a single image. This algorithm estimates the blur parameters on an tessellation, without adaptively selecting the image regions. Blur PSFs are estimated in Fourier domain, thus the block sizes have to be significantly larger than the PSF extent. Finally, the camera ego-motion is not estimated. Another algorithm that exploits spatially variant blur is [53], where the PSF parameters determines the depth map of a static scene. Klapp *et al.* [50] modeled the blur produced by an angular motion of the camera while Nagy *et al.* proposed a restoration algorithm [69] for astronomical images corrupted by spatially varying blur. Recently, it has been shown how to facilitate the PSF estimation task [62, 85, 96] capturing a short-exposure image, paired to a blurred one. Other multiple images methods are [22, 37].

The rest of the chapter is organized as follows: in Section 3.2 we present the blur model showing that e determines the blur direction at any pixel. Section 3.3.1 illustrates the local blur direction estimation methods and the procedure adopted for selecting the image regions. The voting algorithm for estimating e is described in Section 3.4, while the experiments are presented in Section 3.5.

3.2. Problem Formulation

Figure 3.1 illustrates the camera motion we are considering. When the shutter opens, the camera viewpoint is in O, the origin of a 3D coordinate system. We assume that the camera is translating with constant speed during the whole exposure interval T. The point F represents the final position, that the camera reaches as the shutter closes.

Our goal is to estimate, by only analyzing the resulting blurred image z, the 3D direction of camera translation $\overline{OF}$. When the camera calibration matrix is known, this is equivalent to estimating the coordinates on the image plane of the epipole e, i.e. the vanishing point of $\overline{OF}$.

The proposed algorithm analyzes the blur "smears" in z as these, in case of camera translation, are pointing to e. In what follows we describe the image formation process, note that e may not fall into the image boundaries.

3.2.1. Blurred Image Formation

Any image z acquired during a camera motion can be represented as the integration of an infinite number of still images y_t, each captured when the camera viewpoint is in a different position in the space. Thus, we consider the following image formation model:

$$z(x) = \int_0^T y_t(x)dt + \eta(x), \quad x = (x_1, x_2) \in X\,. \tag{3.1}$$

Here, x is a point on the 2D image grid $X \subseteq \mathbb{Z}$, x_1 and x_2 indicate the coordinates of X, $y_t : X \to \mathbb{R}$ represents the light intensity that reaches each pixel at time t, and $\eta \sim N(0, \sigma_\eta)$ is Gaussian white noise. Figure 3.1 illustrates the camera translation and the positions where some images y_t are being captured.

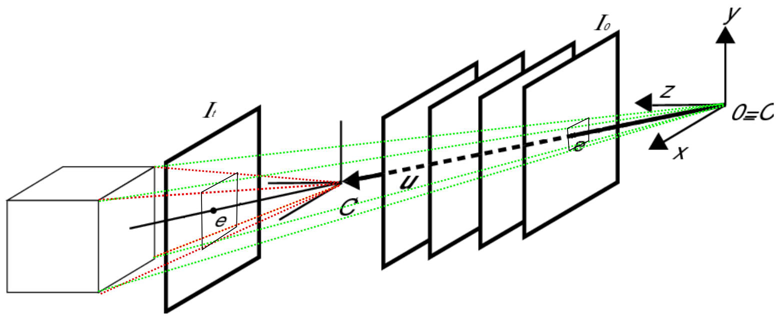


Figure 3.1.: Camera Displacement during the exposure. When the shutter opens, the origin O of the $3D$ axis is in the camera viewpoint with the z axis orthogonal to the image plane. When the shutter closes, the camera viewpoint reaches C.

Any two images I_{t_1} and I_{t_2} form a stereo pair and therefore the correspondences between these two images are related by the essential matrix [36]. Consider I_0 and y_T, the two images acquired at the initial and final camera position, respectively: we define the epipole e as the image of F in y_0 and, likewise, the epipole e' as the image in y_T of the line through O and F.

When the camera translates, e and e' have the same coordinates in both y_0 and y_T, therefore they overlap in the blurred image z as illustrated in Figure 3.1. Thus we define the epipole e in z. Moreover, all correspondences between y_0 and y_T are pointing to e [36], thus the resulting image z is blurred with rectilinear smears. More specifically, the PSF at x_i, is a straight line segment having direction θ_i and extent l_i where

$$\tan(\theta_i) = (x_{i,2} - e_2)/(x_{i,1} - e_1) \quad \text{being } e = (e_1, e_2)\,. \tag{3.2}$$

The blur extents l_i are influenced by the position of the scene point $\mathbf{x}_i$ that is imaged on x_i. Except from some particular cases, e.g those considered in Section 3.2.3 or in [10, 88], it is not possible to provide similar description for blur extents.

3.2.2. Image Blur

The blurring process can be also described as the action of a blur operator K on the original (i.e. still) image, say y_0, in the following way

$$z(x) = \mathcal{K}\big(y_0\big)(x) + \eta(x)\,, \tag{3.3}$$

$\mathcal{K}$ describes the blur at each image pixel and it is written as [4]

$$\mathcal{K}\big(y_0\big)(x) = \int_X k(x,s) y_0(s) ds \tag{3.4}$$

where the weight function $k(x_i, \bullet)$ corresponds to the PSF at pixel x_i. For purely translating cameras, $k(x_i, \bullet)$ represents the smear at x_i

$$k(x_i, \bullet) = R_{\theta_i}\big(\frac{\chi_{[-l_i/2, l_i/2]}}{l_i}\big)(\bullet - x_i)\,, \tag{3.5}$$

where R_{θ_i} is the rotation of θ_i degrees around the image axis x_1 and $\chi_{[-l_i/2,l_i/2]}$ is the characteristic function of the segment $\{-l_i/2 < x_1 < l_i/2, x_2 = 0\}$. In what follows we refer to PSFs of this kind as rectilinear PSFs.

Other blurring effects, such as the out of focus blur, lenses aberrations and camera shake, are not considered in this chapter. Therefore we assume that the same scene, captured from the same camera when it is static, is depicted in a sharp image.

3.2.3. Examples

When e goes to infinity and the captured scene is planar and parallel to the image plane, all the PSFs have the same direction and extent: thus the blur becomes spatially invariant, as shown in Figure 3.2**a**.

When the camera translates toward a planar scene, preserving the image plane parallel to the scene, the PSF extent at any image pixel is determined by its distance, on the image plane from the epipole [10, 88]. Denoting by $u = |\overline{OF}|/d$ the ratio between the scene depth d and the length of displacement $|\overline{OF}|$, and by $\overline{x_i e}$ the distance between the pixel x_i and e, it follows that the PSF extent l_i in (6.3) is

$$l_i = \overline{x_i e}\frac{u}{1+u}\,. \tag{3.6}$$

An image synthetically blurred, with e in the image center and extents given by Equation (3.6), is shown in Figure 3.4**a**. Figure 3.4**b** and Figure 3.4**c** shows the PSF direction and extent at each pixel. Figure 3.2**b** shows a camera image captured with the described settings.

Typically, only the PSFs directions are determined by e, while the extents depend on the 3D position of the scene points, like in images of Figures 3.3**c** and 3.3**d**. In particular, in Figure 3.3**c** e goes to infinity and the scene is non planar: in this case the blur direction is constant and the blur extents are proportional to the scene depth.

3.3. Proposed Solution

The coordinates of e are determined by estimating the blur directions within automatically selected image regions: for this purpose we exploit two different blur direction estimation methods. The first method is meant for regions

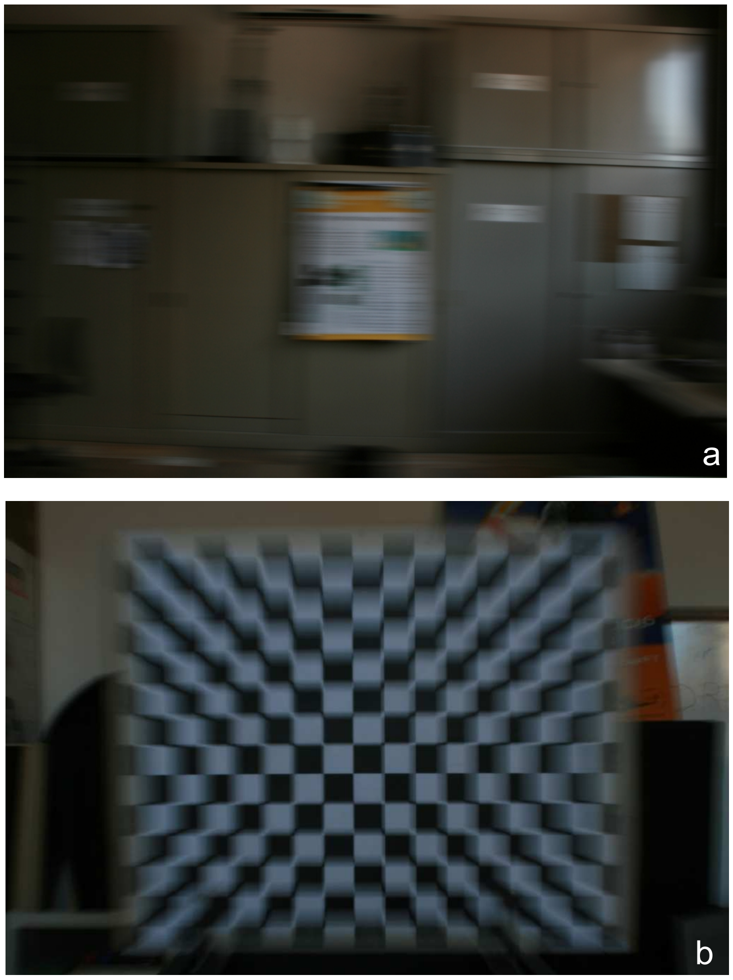


Figure 3.2.: Example of blur produced by a translating camera: **a** the blur is spatially invariant as the scene is planar, parallel to image plane and $e \to \infty$, **b** the blur is spatially variant even if the scene is planar as e lies on the image plane.

containing an image corner, while the second method applies to regions containing other blurred details. We also present a procedure for determining the method to be used in each region.

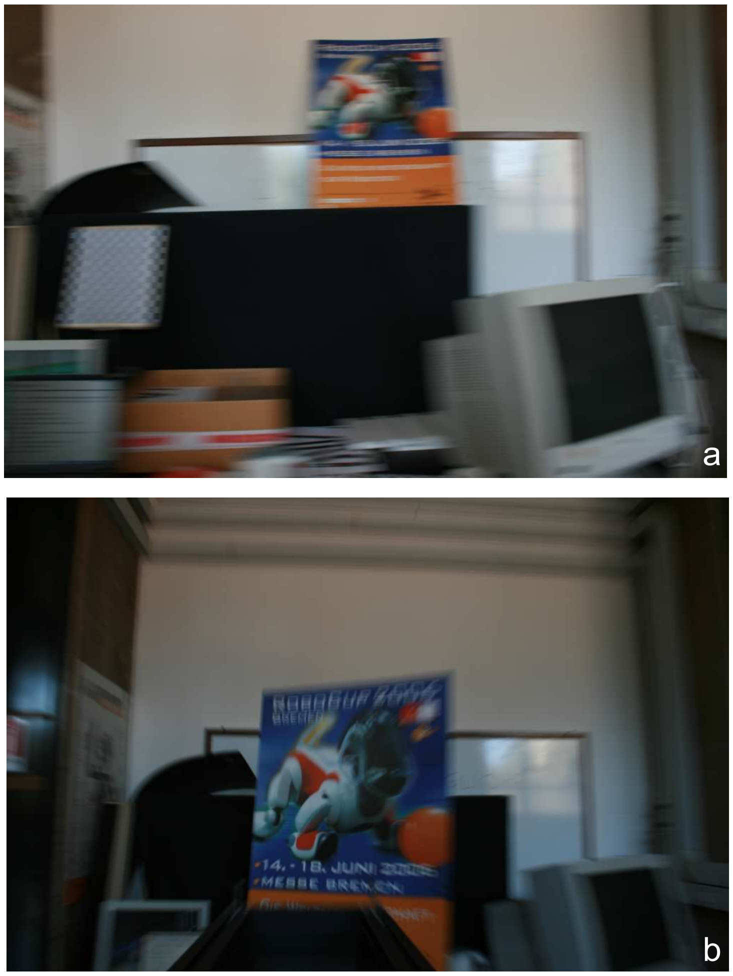


Figure 3.3.: Example of blur produced by a translating camera: **a** the blur is spatially invariant even if $e \to \infty$ as the scene is not planar (but blur directions are constant), **b** the blur is spatially invariant and the scene is not planar

3.3.1. Local Blur Parameters Estimation

In order to estimate θ_i, the blur direction in an image region U_i, we treat the blur operator K as locally spatially invariant. We assume that $\forall x_i \in \mathcal{X}, \exists\, U_i \subset \mathcal{X}$, a neighbor of x_i, and a PSF v_i, such that

$$K\big(y_0\big)(x) \approx \int_X v_i(x-z) y_0(z) dt \quad \forall x \in U_i\,. \tag{3.7}$$

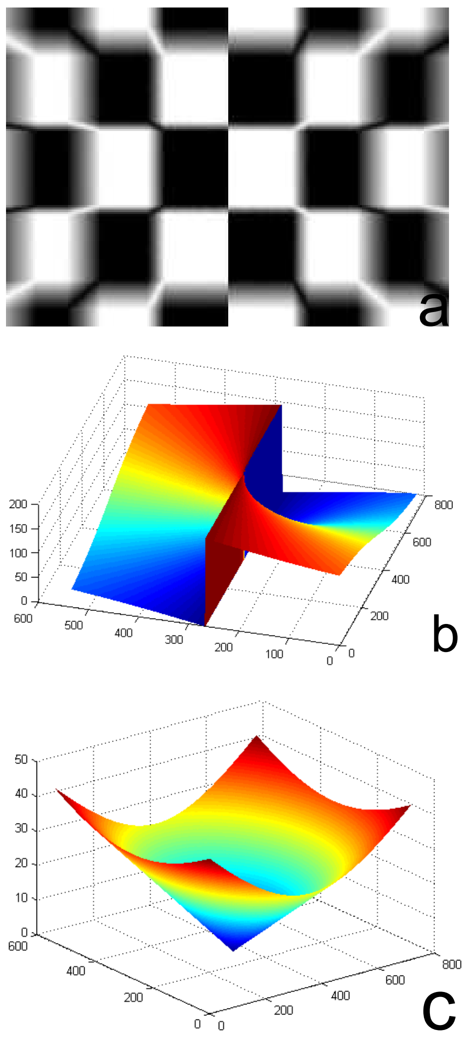
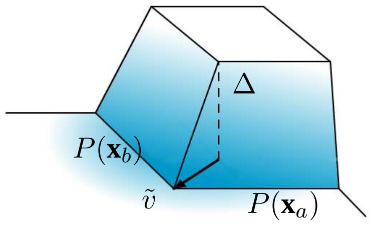


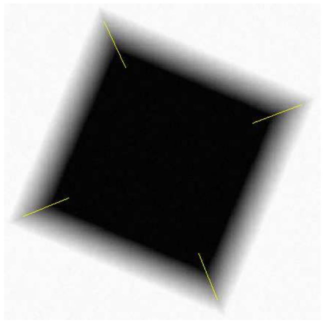
Figure 3.4.: **a** Synthetic image blurred according to Equation (3.6); **b** blur directions at any pixel; **c** blur extents at any pixel

Furthermore we assume v_i is a rectilinear PSF, having direction θ_i and extent l_i. Such approximation allows us to estimate blur parameters within selected regions with methods meant for spatially invariant blur.

Fourier domain methods, which are widely used for blur parameters estimation, do not perform properly on small image regions, as they assume periodic signals. Thus we adopt space domain methods. Within regions containing an image corner, we estimate the PSF direction by analyzing the corner edges [5].



(a) The image model within a blurred corner region. The displacement vector $\tilde{v}$ represents the blur PSF while Δ represents the difference between the corner and the background.



(b) A Synthetically blurred image (*Synth1*) and direction estimates

Figure 3.5.: Blur direction estimates at corners using the algorithm presented in [5].

Within regions where there are no blurred edges and the image is not flat, the blur direction is estimated analyzing the ℓ^1 norm of directional derivatives [93].

Blurred corners allow a clear interpretation of rectilinear PSF parameters. The method proposed in [5] estimates, in each corner region, the corner displacement vector $\tilde{v}$, which represents the blur direction and extent. This method analyzes the image gradient into the blurred edges areas and estimates Δ, the intensity gap between the corner and the background. Figure 3.5(a) illustrates the image model at blurred corners.

Within other regions we use the method proposed in [94]: the PSF direction θ_i, is estimated as the direction of the directional derivative filter d_θ having minimum ℓ^1 norm

$$\theta_i = \arg \min_{\theta \in [0,2\pi]} \left(||(d_\theta \circledast I_{|U_i})||_{\ell^1} \right), \tag{3.8}$$

where $I_{|U_i}$ is an image containing only pixels within region U_i and $||z||_{\ell^1} = \sum_{x \in \mathcal{X}} |(z(x))|$. In our experiments we use 7-tap directional derivative filters devised in [21], convolved with a 3-tap derivative filter on the orthogonal direction. This latter filter acts as a whitening on the image content as in [94]
.

In Section 2.2 we introduce an adaptive region selection procedure for corner regions that also determines the blur estimation method to be used in each region.

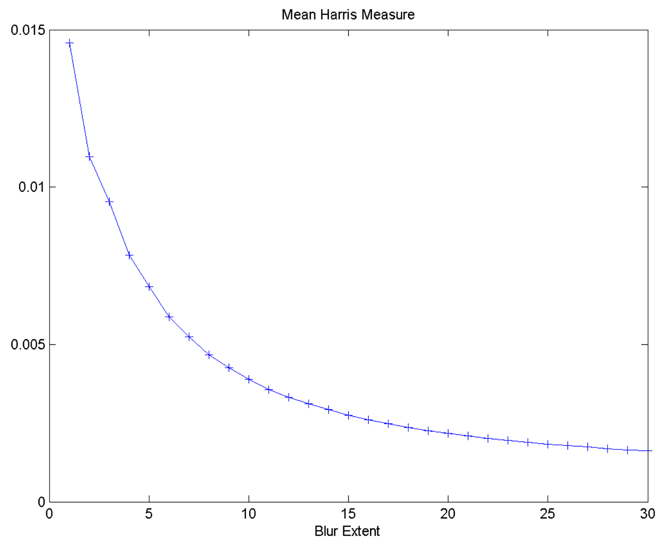


Figure 3.6.: Mean Harris measure as a function of blur extent

3.3.2. Salient Point Detection

Blurred corners are extracted from the local maxima of the Harris measure [35, 59, 65], according to the procedure described in Chapter 2. We already presented the corner detection procedure based on the Harris measure for blurred images. Let assume then that $\{x_i\}$ $i = 1, .., M$ are the pixels detected as blurred corners.

We are going now to discuss the use of local maxima of Harris measure (salient point) for detecting regions where the blur direction can be correctly estimated using Equation (6.5). Note that Equation (6.5) gives a reliable estimate within regions where the original image y_0 has the same (non zero) ℓ^1 norm response to any directional derivative filter. This for example happens when y_0 is white noise.
As mentioned before, the local maxima of Harris measure are pixels where the image is presenting a significant variation along any direction. We assume than that since these regions are blurred by a rectilinear PSF, the resolution along the blur direction decreases and the direction given by Equation (6.5) corresponds to the blur direction. However, since y_0 is unknown, we have to extract the salient points from the blurred image z.
We run the following experiments to show that the salient point in the blurred image z belong to areas of the original image y_0, where the Harris measure is

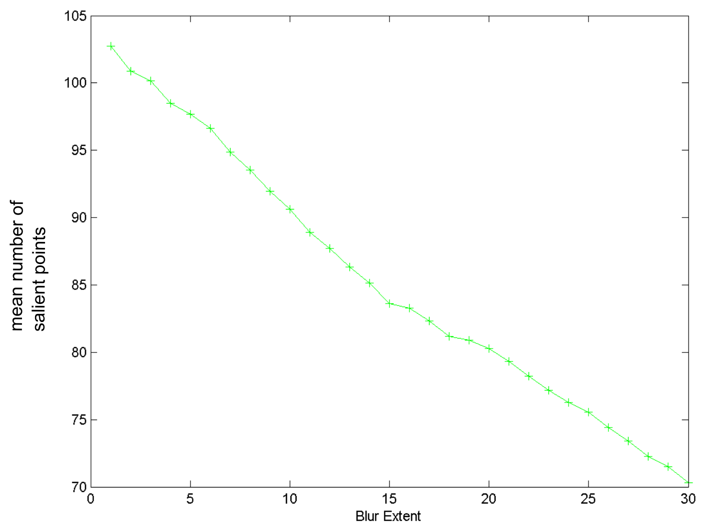


Figure 3.7.: Mean number of salient point as a function of blur extent

large. We consider a dataset of 12 common test images (rescaled in the range $[0, 1]$) and two set of parameters $\Theta = \{0, 10, .., 170, 180\}$ for the PSF directions, and $L = \{1, 2, .., 29, 30\}$ for the PSF extents. We synthetically blur each test image with a convolution against each PSF generated from all possible parameter pairs ($\Theta \times L$), obtaining thus $\{f_j\}\ j = 1, .., 12\,\#\Theta \times L$. We compute the Harris measure on each blurred image and we extract the local maxima. The Harris measure is thresholded against a fixed value $T = 0.0005$, as this is a standard procedure for reducing low-relevance features. In what follows we always consider only pixels where the thresholded Harris measure is nonzero. Figure 3.6 shows the mean Harris measure as a function of the blur extent: as expected the blur reduces the details in the image and thus the average value of the Harris measure. Also the number of local maxima decreases, as illustrated in Figure 3.7 .

We already discussed in Section 2.1 about the localization error of local maxima of the Harris measure in blurred corners. Here we are not aiming to accurately locate the salient point of y_0 by taking those of z. For our purpose it we have to show that in a neighbor of each salient point in the blurred image, the original image y_0 shows significant variations along all directions. This is enough to show that it make sense estimating the blur direction near pixels using Equation (6.5). We thus consider a squared neighbor of 10 pixels side around each salient point in each blurred image f_j, and we compute m_j as the

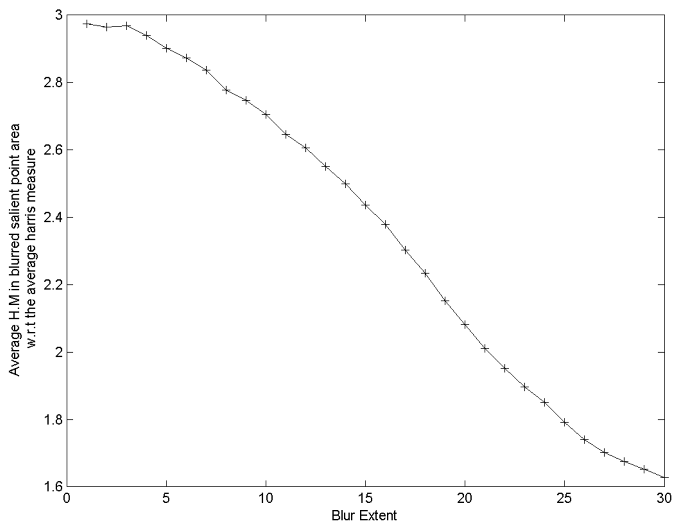


Figure 3.8.: The mean Harris measure on the original image computed over the 10 pixel square size neighborhood of the salient point in the blurred image, as a function of blur extent

average on these neighborhoods, of the Harris measure on the corresponding pixels in the original image. In order to compare values of m_j computed on different images we divide m_j with the average Harris measure on the whole corresponding original image. Figure 3.8 shows the averaged values of m_j as a function of the blur extent. The Harris measure on the selected neighborhood in the blurred image decreases as the blur extent increases. However, even when heavy blur is considered, the salient points are still in areas where the original image shows some significant variations (it is still 1.6 times the average Harris measure).

Thus we select $\{x_i\}i = M, .., N$ as the local maxima of the Harris measure of the blurred image, avoiding those pixels which have been selected as blurred corner according to the procedure of Chapter 2. Around these pixels we consider circular neighbors for estimating blur using Equation (6.5).

Note that in the presented experiment we generate blurred images with a convolution against a PSF and thus the blur is spatially invariant. However, since the Harris measure is computed locally, and the blur here is treated as locally spatially invariant, this does not reduce the validity of our experiment. Moreover results have been averaged combining results from blurred images having the same blur extent but different directions. This follows from the fact that the Harris measure in its original formulation is rotational invariant [35].

3.3.3. Drawbacks

Blur directions estimated using Equation (6.5) may be seriously influenced by the image content. For example, in edge regions the estimated direction typically corresponds to the edge direction, regardless of blur and the same for regions where the original image y_0 presents features highly self-correlated (e.g. striped textures). Other regions are non informative, for example where y_0 is flat, the blur does not produce any change. Therefore we restrict to regions containing a local maxima of Harris corner measure [35] which points out that the image significantly varies along two different directions.

3.4. Camera Displacement Estimation

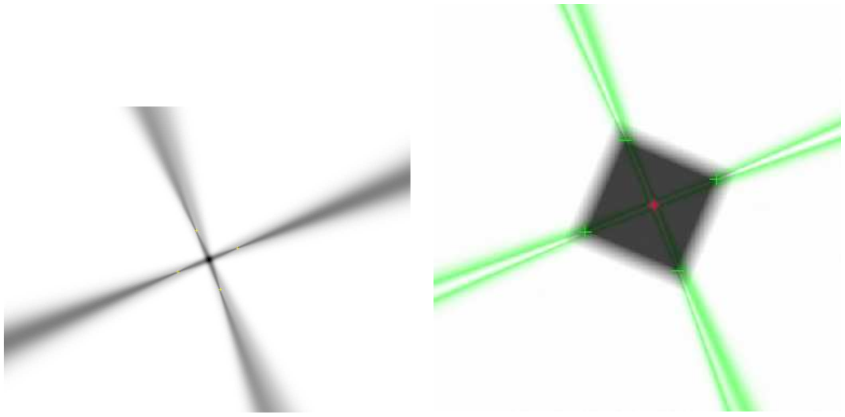
The estimated blur directions are then used to determine the epipole e. From every estimates pair (θ_i, x_i) $i = 1,..,N$ it follows that e should lie on ℓ_{i,θ_i}, the line passing through x_i and having blur direction θ_i. We use the Hough approach for estimating e as the pixel which has been crossed by the largest number of lines ℓ_{i,θ_i}. The parameter space represents the set of all the possible locations for e and it is a discrete grid larger than $\mathcal{X}$ (possibly at a different resolution) with a set of angles in $[0, \pi)$. These angles describe the case $e \to \infty$ (as in Figure 3.2**a**.).

For each estimate pair (θ_i, x_i) $i = 1,..,N$, the votes are assigned to the parameters that agree with these data, i.e. the line ℓ_{i,θ_i}, and then summed to the votes coming from all the other estimate pairs. The voting algorithm allows to take into account the uncertainty of each estimated direction and, in case of corners, also the uncertainty of corner location. Therefore the line is replaced by a weight function that assign a full vote to the exact solution and a lower vote to every parameters close to the exact solution. The weight function $\ell_{i,\theta}$ is obtained rotating of θ_i degrees and centering in x_i the following function

$$\ell(x_1, x_2) = exp\Big[-\Big(\frac{x_2}{(1 + h|x_1|)k}\Big)^2\Big]. \tag{3.9}$$

Here k expresses the localization error and h the error in the direction estimated.

In our experiments we used $k = \sigma_\eta$ for estimates coming from minimum derivative energy and $\sigma_i = k_0 + \sigma_\eta$ for estimates at corners. k_0 is a tuning parameter that compensate the errors in estimating the corner location. h is related to direction estimation error and determines the vote spread from the



(a) (b)

Figure 3.9.: Voting procedure using the estimated directions on Figure 3.5(a), (a) Parameter space and corresponding votes distributed according to (3.9), (b) Votes are represented on the green channel, e is marked with red a cross.

set of exact solutions (Figure 6.5). Another vote (also with a Gaussian spread) is then assigned to every line direction.

After having considered all the estimates, the point on the parameter space that obtained the maximum amount of votes is selected as the epipole e.

The voting algorithm adds robustness to the procedure as directions estimated are inaccurate and there are typically outliers. In case of corners, even direction correctly estimated may be far from pointing to e because of occlusion or shading: in fact, image regions representing scene parts belonging to different objects (occlusions) are not blurred according to the epipolar constraints while shadows may be also erroneously considered as blurred corner edges (see Figure 3.12.**1a**). Also the directions estimated using minimum of directional derivatives could be outliers if in the considered region there are edges or the original still image is highly correlated (see Figure 3.12.**2a**).

3.5. Experiments

Synthetic test images have been generated according to model (3.1), using the ray tracer software Pov-Ray [41]. We generated seven 3D scenarios containing parallelepipeds placed at different depths and orientations. For each scenario, we rendered a sequence of 60 frames translating the camera along the camera axis, so that e was in the image center. Each frame was rendered at a resolution

σ_η	Synth1	Synth2	Synth3	Synth4
1	2.17	1.25	9.83	4.33
2	2.50	1.61	11.23	3.33
3	2.36	2.42	13.65	2.48
4	2.78	3.90	15.36	6.03
n	4.0	15.7	14.8	16.5

σ_η	Synth5	Synth6	Synth7	House8	House9
1	9.83	15.40	3.06	8.74	8.21
2	11.23	14.81	4.28	8.30	9.40
3	13.65	17.16	6.75	9.62	10.56
4	15.36	22.38	8.39	11.59	11.29
n	31.1	22.1	10.7	74.5	85.7

Table 3.1.: Table shows the distance in pixels, averaged over 10 realization of η, of estimated epipole e from the image center. The last row contains the average number of blur directional estimates per image

of 1024 x 768 pixels in grayscale (0-255) and the blurred image z has been then obtained averaging all the frames. These test images are *Synth1* in Figure 3.5(b) and *Synth2*-*Synth7* in Figure 3.10.

Table 3.1 shows the distance in pixels between e and the image center, averaged over 10 different realization of noise for each value of σ_η. The last row of Table 3.1 shows the average number of blur direction estimates used. In these test images, the blur directions have been estimated only exploiting corners as there are no details that can be used for estimating the blur using minimum energy derivative method. The maximum length of wedge mask for corner region selection is 25 pixels.

We test our algorithm on two test images *House8* and *House9* of Figure 3.10 rendered in the same way using two more complex and textured scenarios [40] so that the resulting blurred images are suited for estimating blur with both formula (6.5) and with method for corners. The last two columns of Table 3.1 shows the algorithm accuracy, averaged over ten different realization of η. The overall number of blur direction estimates is increased w.r.t to the previous case as in several points blur has been estimated using (6.5). Figure 3.11.8.a and 3.11.9.a show in yellow the directions estimated using corner method while in red the directions estimated using Equation (6.5). Blur extents have not been estimated and segment lengths are fixed in the figures. Figure 3.11.9b and 3.11.9b show the selected corner regions and 3.11.9c and 3.11.9c the parameter space with all the votes.

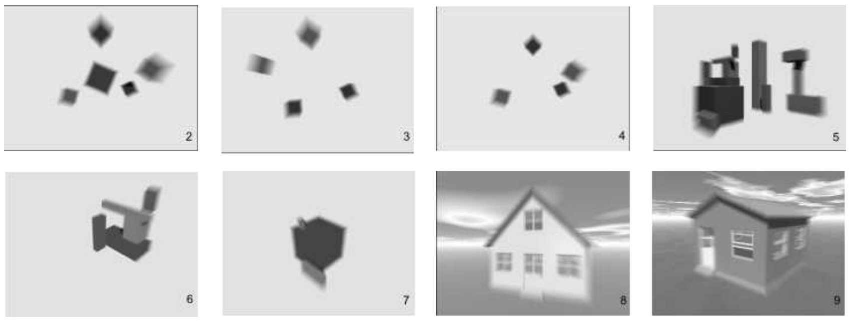


Figure 3.10.: Synthetic Images *Synth2*-*Synth7* and *House8*,*House*

The proposed algorithm has been tested also on camera images taken with a Canon EOS 400D 10Mpixel, see Figures 3.12 and 3.13. Images in Figure 3.12 have been taken with the camera mounted on a wheeled device to produce uniform motion, while images in Figure 3.13 are jpeg taken from the same camera mounted on a serving cart (Figure 3.13.3.a), moved by hands with a short exposure (Figure 3.13.4.a) or by bringing down the camera tripod. Algorithm results are presented by cyan lines joining the estimated epipole e with image borders. Blur direction estimates, votes in parameter space and corner regions are illustrated like in Figure 3.11. Even if there is no ground truth, the cyan lines show that image blur has been correctly interpreted.

3.5.1. Discussion

Tests run on synthetic and camera images show that e is estimated accurately, even in noisy images. Both blur direction estimation methods have been designed to cope with Additive Gaussian White Noise (AWGN). The result of Equation (6.5) is not influenced by AWGN, while in [5] the method for corner regions provided satisfying performance in presence of AWGN. On the other hand, the image content influences more seriously blur these methods. We already mentioned occlusions and shadings for method [5], examples of occlusions and shadings appears in images *Synth4*-*Synth7* and reduce the algorithm accuracy. Occlusions are also present in camera images, in particular in Figure 3.12.1.a where the scene presents several depth levels. The blur estimates from Equation (6.5) are easily affected by edges and line, as one can clearly see from Figure 3.12.2.a the direction estimated at pixel near the cabinet lines. In flat regions the directions estimated by Equation (6.5) is aslo not reliable. In the considered cases however, the voting procedure gives a reliable estimate of e even in presence of such outliers as far as there are enough inliers.

The overall computation time depends on the number of regions where blur

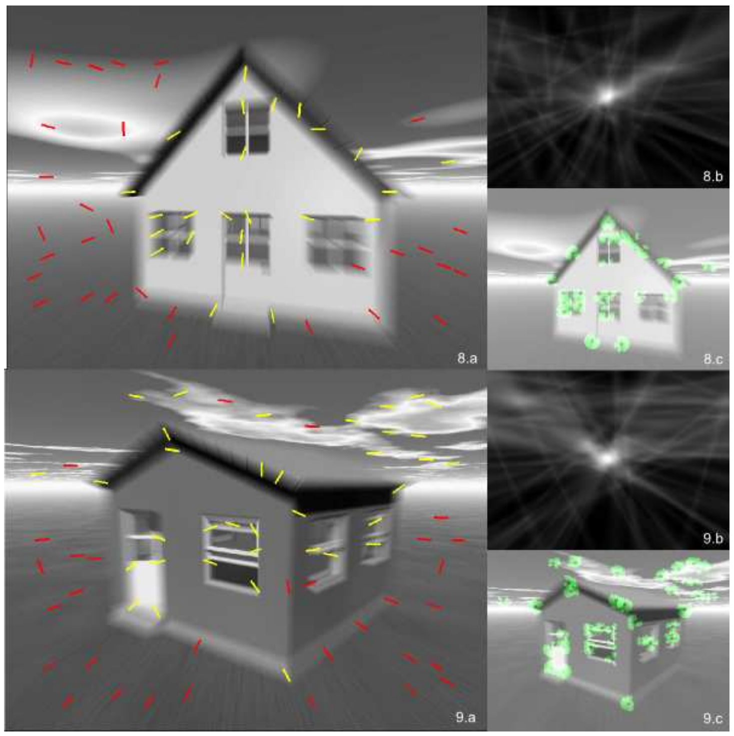


Figure 3.11.: Result on Images *House8* and *House9*

is estimated. This is the computationally heaviest part. The corner region selection procedure is based on local averages and comparisons, therefore its computation cost is linear in the region size. Corner blur estimation is also linear in the selected region sizes, [5]. Finally directional derivatives of Equation (6.5) are computed using separable filters [21], and thus each one is a linear combination of the response to four filters. The minimization of Equation (6.5) can be sped up in a multiscale approach.

3.6. Conclusions

In a blurred image produced by a translating camera, the blur direction and extent are varying through the image pixels according to the scene depth and the camera motion. In this Chapter we have described the blur produced by a translating camera, and we have derived an algorithm for estimating the vanishing point of camera displacement. In such a way the camera ego-motion direction can be estimated by analyzing the single blurred image.

The algorithm relies on the estimation of blur direction at image regions

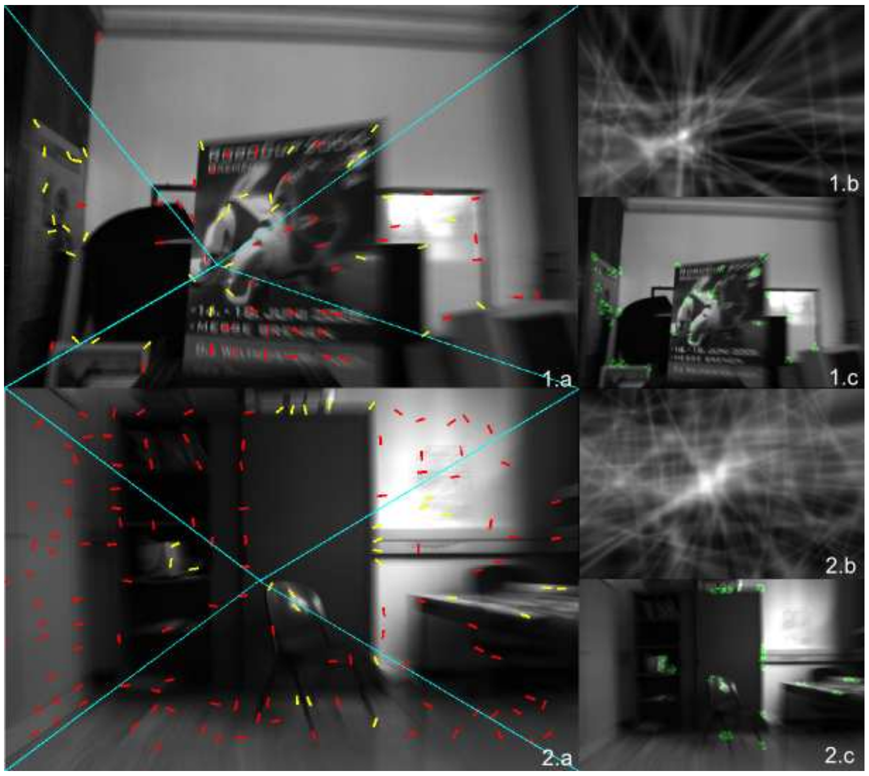


Figure 3.12.: Tests on Raw Data

which have been automatically detected and selected for improving the blur estimators.

This algorithm can be used in order to improve robot vision system based on frame analysis. In fact these systems, e.g. [19], often have to cope with blurred images because of reduced lightning conditions in indoor environments. Then, instead of discarding the blurred frames where it is not possible to match map features, the system can exploit the image blur for the estimation of the global displacement direction and the local blur estimates to replace of feature matches.

Moreover we believe that this algorithm can be used for estimating the blur from a single image and therefore can improve deblurring methods that consider also spatially varying blur, such as that one presented in [89].

Finally, we only considered translational camera motion as this particular case do not require estimates of the blur extent. Whenever there are enough blur direction and extent estimates a more general rigid camera motion can be considered, enforcing other rigidity constraints.

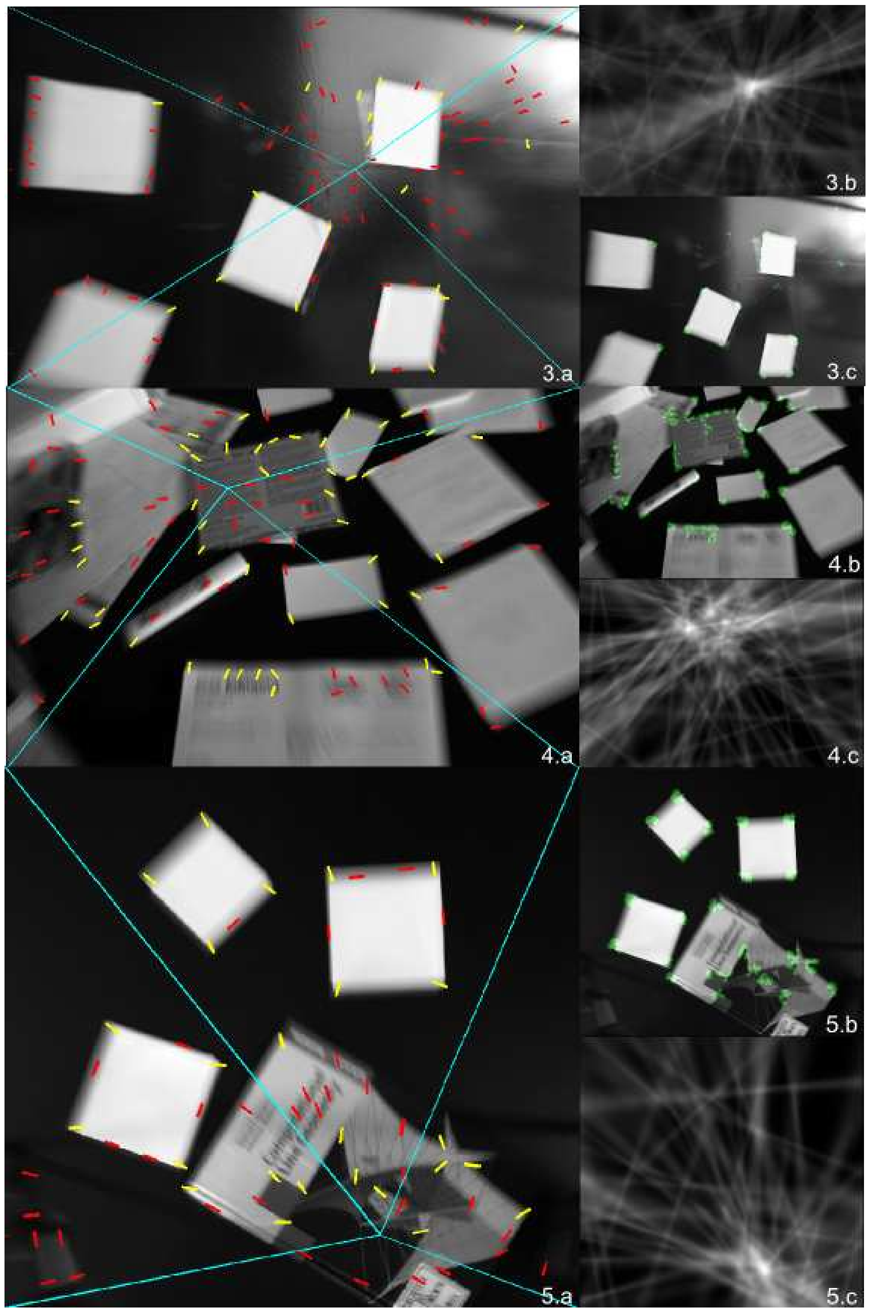


Figure 3.13.: Tests on Jpeg Camera Images

4. Deblurring Noisy Radial-Blurred Images

In the previous chapter we devised an algorithm for estimating the vanishing point of camera translation direction, e. In this chapter, we address the issue of restoring images corrupted by radial blur, i.e. blur due to camera translation not parallel to image plane. More precisely, we assume that the capture scene is planar and parallel to the image plane, as in this case the blur PSF admit an easy parametric description. In Appendix 7 we show how the considered blur model derives from the uniform motion of a pin-hole camera.

The deblurring of images corrupted by radial blur is studied. This type of blur appears in images acquired during any camera translation having a substantial component orthogonal to the image plane. The point spread functions (PSF) describing this blur are spatially varying. However, this blurring process does not mix together pixels lying on different radial lines, i.e. lines stemming from a unique point in the image, the so called “blur center”. Thus, in suitable polar coordinates, the blurring process is essentially a 1-D linear operator, described by the multiplication with the blurring matrix.

We consider images corrupted simultaneously by radial blur and noise. The proposed deblurring algorithm is based on two separate forms of regularization of the blur inverse. First, in the polar domain, we invert the blurring matrix using the Tikhonov regularization. We then derive a particular modeling of the noise spectrum after both the regularized inversion and the forward and backward coordinate transformations. Thanks to this model, we successfully use a denoising algorithm in the Cartesian domain. We use a non-linear spatially adaptive filter, the Pointwise Shape-Adaptive DCT, in order to exploit the image structures and attenuate noise and artifacts.

Experimental results demonstrate that the proposed algorithm can effectively restore radial blurred images corrupted by additive white Gaussian noise.

The materials presented in this chapter have been published in a conference paper [10].

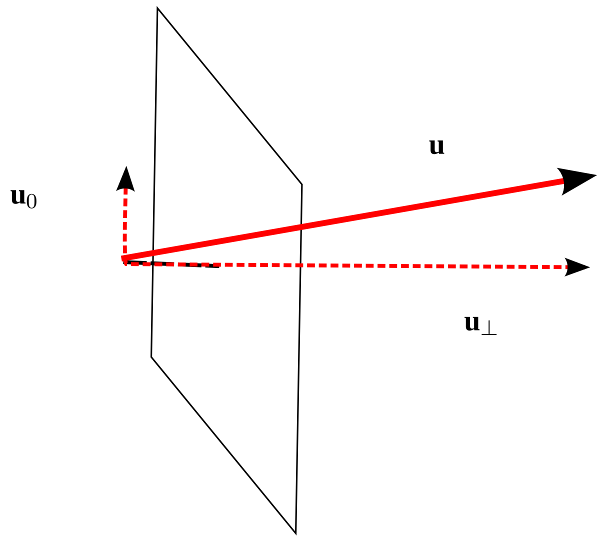


Figure 4.1.: Camera translation vector $\mathbf{u}$ and its components $\mathbf{u}_\perp$ and $\mathbf{u}_0$.

4.1. Introduction

In this chapter we consider the restoration of images corrupted by blur produced by a camera that translates in the 3D space with constant velocity during the acquisition. This situation can be formalized as follows. Let $\mathbf{u}$ be the 3D vector that identifies the camera translation during the exposure time T. This translation can be decomposed in two components: $\mathbf{u}_\perp$ and $\mathbf{u}_0$, which are orthogonal and parallel to the image plane respectively, as shown in Figure 4.1. Typically, image restoration algorithms assume $\mathbf{u}_\perp = 0$, considering the camera translating parallel to the image plane. This assumption leads to the spatially invariant blur degradation model (see [24, 76, 95] and references therein).

In this chapter we focus on the generic case when $\mathbf{u}_\perp \neq 0$. The presence of a significant component $\mathbf{u}_\perp$ makes the blur spatially variant and the image restoration becomes a much more challenging problem. For simplicity, the captured scene is assumed planar and parallel to the image plane, thus neglecting complications due to the scene depth.

Let us consider the blurred image z as integration of *sub-images* y_t,

$$z(x) = \int_0^T y_t(x)dt\,, \quad x = (x_1, x_2) \in \mathcal{X}, \tag{4.1}$$

where x represents a coordinate in the image domain $\mathcal{X}$, T is the exposure

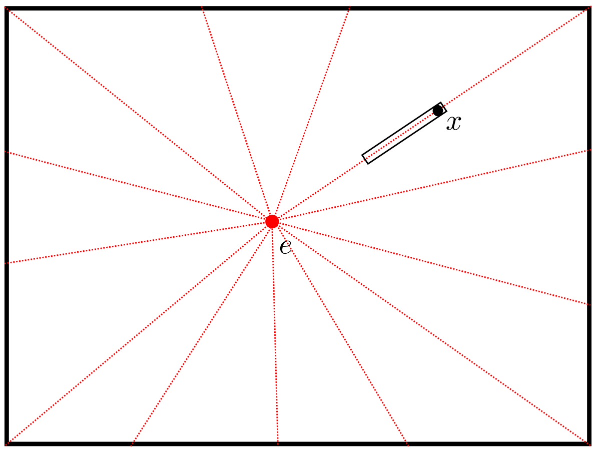


Figure 4.2.: The blur center e and the radial lines. The segment at x includes samples of the original image y that contribute to the blurred observation $z(x)$ at x.

time and y_t is a sub-image, i.e. a sharp image produced by the light intensity that reaches the sensor at the instant $t \in [0, T]$. Each sub-image y_t is acquired with a different viewpoint along the camera trajectory.

In the trivial case when $\mathbf{u}_\perp = 0$, all sub-images are shifted w.r.t. each other, i.e. $y_t(x) = y_0(x - \lambda t)$, $\lambda \in \mathbb{R}^2$. Because the shift λt is the same for each point x, the blurred image can be modeled as the convolution of the original image with a kernel, the point spread function (PSF), which has a 1D support and which is typically parametrized by its direction and its extent.

When the camera translation is not parallel to the image plane, i.e. $\mathbf{u}_\perp \neq 0$, the integration support corresponding to each point varies. The relations between any of two sub-images are described by the essential matrix [36], which acts differently on each image point. Nevertheless, in this case, the blur can be described by the blur center e and the blur extent parameter l. Such images are termed *radial blurred images* because the blur smears are directed along radial lines, i.e. lines stemming from the blur center, as shown in Figure 4.2. As we move away from the blur center, which itself is not blurred, the length of the smears grows with rate equal to the extent l. In particular, the smear at a point x such that $|x - e| = 1$ has length l, where $|x - e|$ is the distance between pixel x and the blur center e.

There are only few publications about the restoration of images corrupted by the radial blur. Webster and Reeves [88] addressed this problem and proposed a fast restoration algorithm based on resampling the blurred image on a lattice where the blur becomes spatially invariant, so that any deconvolution technique (e.g., in Fourier domain) can be used. However, in their work, the image noise was not considered explicitly. In principle, any method for restoring images corrupted by spatially variant blur [69] could be used for radial blur. However, these methods result in heavy computational costs, while the restoration of radial blurred images becomes simpler after performing adequate image transformations. In fact, when the image is transformed in polar coordinates w.r.t. to the blur center e, the blurring process having extent l can be modeled by the multiplication against an upper triangular matrix B_l.

The proposed restoration algorithm exploits Cartesian-to-polar and polar-to-Cartesian coordinate transformations which enable a computationally affordable blur inversion. Because a naive pseudo-inverse can amplify the noise and artifacts, we exploit two separate forms of regularization in both polar and Cartesian domains. In the polar domain, we invert the matrix B_l using a Tikhonov regularized inverse. Then, we perform denoising in the Cartesian domain using a non-linear spatially adaptive filter, the Pointwise Shape-Adaptive DCT [30, 28, 29], in order to exploit the image structures and attenuate both noise and artifacts. For this filtering we derive approximate models of the noise spectrum for the forward and backward coordinate transformations, in order to drive the denoising in Cartesian image domain.

The rest of the chapter is organized as follows. Section 4.2 presents a model for images corrupted by radial blur, Section 4.3 describes the blur inversion stage in the ideal, noiseless case. The noise is introduced and handled in Section 4.4, where the noise characteristics after the blur inversion are estimated in order to be applied in the denoising algorithm. Finally, in Section 4.5 we validate the proposed algorithm by experimental results.

4.2. Image Formation

Typically, a spatially variant blurred image z is modeled by the following integral

$$z(x) = \int_{\mathcal{X}} k(x,s)y(s)ds\,, \quad x = (x_1, x_2) \in \mathcal{X} \tag{4.2}$$

where y is the original image and the PSF $k(x,\cdot)$ determines how the intensity values of the original image y contribute in $z(x)$.

In the case of the radial blur, the PSF in each image pixel x is determined by its position w.r.t. the blur center $e = (e_1, e_2)$ and by the extent parameter l. The blur center e is the pixel that corresponds to the image of the vanishing point of camera translation direction, and thus it is the only pixel which is not blurred in the radial blurred image. The blur extent parameter $l > 0$ determines how the blur extent increases along each radial line, i.e. the blur extent at a pixel x is

$l\,|x-e|$. The parameter l depends on the length of the camera displacement and on the distance between the camera viewpoint and the planar scene (measured along the camera translation direction).

Figure 4.4(a) shows an example of a radial blurred image: the blur is visibly characterized by smears along the radial lines. The radial blurring process does not mix together pixels lying on different radial lines and thus the radial blur can be compactly formulated in the polar coordinates.

Let $\mathfrak{P}$ and $\mathfrak{C}$ be two continuous domain transforms that perform respectively Cartesian-to-polar and polar-to-Cartesian image mapping w.r.t. the blur center e:

$$Z(\rho,\theta) = \mathfrak{P}(z)(\rho,\theta) = z(e_1 + \rho\cos\theta, e_2 + \rho\sin\theta)\,, \tag{4.3}$$

and

$$z(x_1,x_2) = \mathfrak{C}(Z)(x_1,x_2) = Z\left(\sqrt{(x_1-e_1)^2 + (x_2-e_2)^2}, \arctan\left(\frac{x_2-e_2}{x_1-e_1}\right)\right)\,. \tag{4.4}$$

Here and in what follows, capital letters indicate images expressed in the polar coordinates, while small case letters indicate images in Cartesian coordinates.

Then, the radial blur can be written as

$$Z(\rho,\theta) = \int_{\mathbb{R}} \frac{1}{\rho\, l}\chi_{[\rho,\rho+l\rho]}(r) Y(r,\theta) dr\,, \tag{4.5}$$

where $\chi_{[a,b]}$ is the characteristic function of the interval $[a,b]$,

$$\chi_{[a,b]}(r) = \begin{cases} 1 & a \leq r \leq b \\ 0 & \text{else}, \end{cases} \tag{4.6}$$

In practice we have to deal with discrete data, therefore we use also two discrete domain coordinate transformations, $\mathcal{P}$ and $\mathcal{C}$, which perform the Cartesian-to-polar and the polar-to-Cartesian mapping, respectively. These transformations can be obtained by discretization of the continuous operators of Equations (4.3) and (4.4). The transform $\mathcal{P}$ maps the input $w \times h$ image into an $r \times \tau$ output matrix. To obtain such an output with a rectangular domain in polar coordinates, the transform implicitly pads the input image (e.g., by zero-padding). Without loss of generality and for practical reasons, we assume that the blur center e is at the image center, as this situation can be reproduced by padding and shifting the image adequately.

The radial blurred image in the polar coordinates Z can be expressed as a matrix multiplication

$$Z = B_l Y\,, \tag{4.7}$$

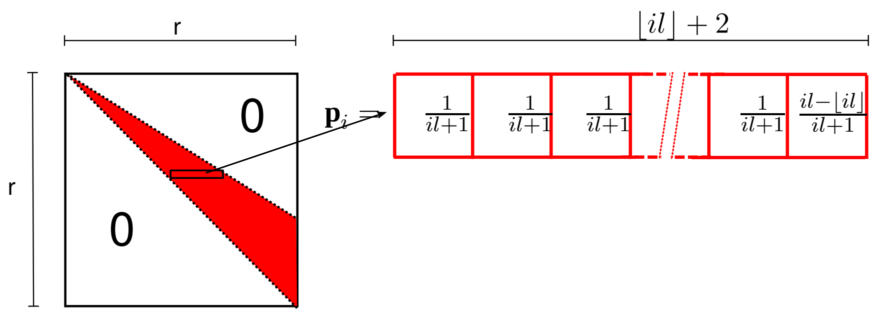


Figure 4.3.: The blurring matrix B_l is an upper triangular $r \times r$ matrix. The upper part presents only a narrow band above the diagonal where it is not null. The rows are defined in formula (4.11) and the non-zero are shaded.

where $Y = \mathcal{P}(y)$ and B_l is an upper triangular matrix, that depends on the blur extent parameter l. The noise-free observations are given by

$$z = \mathcal{C}\left(B_l Y\right) . \tag{4.8}$$

We emphasize that the above equations are discrete approximations of the continuous domain equations (4.3) - (4.5). Note also that both $\mathcal{P}$ and $\mathcal{C}$ exploit data interpolation in order to compute image values corresponding to non-integer coordinates, hence it may happen that

$$\mathcal{C}\left(\mathcal{P}(z)\right) \neq z \quad \text{and} \quad \mathcal{P}\left(\mathcal{C}(Z)\right) \neq Z. \tag{4.9}$$

4.2.1. The Blurring Matrix

The matrix multiplication between the image in polar coordinates Y and the blurring matrix B_l corresponds to a discretization of the operator of Equation (4.5), which models the radial blurring process of extent l. Note that in polar coordinates each column of Y contains r pixels lying on the same radial line, thus B_l is a $r \times r$ matrix

$$B_l = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_{r-1} \\ \mathbf{p}_r \end{bmatrix} , \tag{4.10}$$

where the row vectors $\mathbf{p}_i = \{p_{i,j}\}_{j=1,\ldots,r}$, $i = 1, \ldots, r$, are

$$\mathbf{p}_i = \begin{cases} 1/(il+1) & i \leq j \leq \lfloor i + il \rfloor \\ (il - \lfloor il \rfloor)/(il+1) & j = \lfloor i + il + 1 \rfloor \\ 0 & \text{else}, \end{cases} , \tag{4.11}$$

and $\lfloor \cdot \rfloor$ denotes the rounding to the nearest smaller or equal integer.

As shown in Figure 4.3, B_l is an upper triangular matrix and each row $\mathbf{p}_i$ corresponds to an horizontal blur PSF, which, provided that the PSF support is totally included within the matrix, has extent $il + 1$. These PSFs are exactly the sub-pixel discretization of the function χ of integral (4.5). Note that the last rows of B_l have ℓ^1-norm smaller than 1, because the supports of the corresponding PSFs lie partially outside of the image domain. In practice, this is equivalent to zero-padding the image and producing the blur with unrestricted PSFs (hence having their norm equal to 1). It results in synthetically blurred images having borders fading to zero, as shown in Figure 4.4(a).

4.3. Blur Inversion

The blur inversion consist of estimating the original image y from z, assuming that the parameters e and l are known. The observation model presented in Section 4.2 relies on two image transformations, these transformations are used also for blur inversion. In polar coordinates the blur is a multiplication against the blurring matrix B_l (4.7). Therefore, a straightforward solution to blur inversion consists of transforming the observation z in polar coordinates w.r.t. e, i.e. $Z = \mathcal{P}(z)$ and by multiplying it against the inverse of blurring matrix, B_l^{-1}.

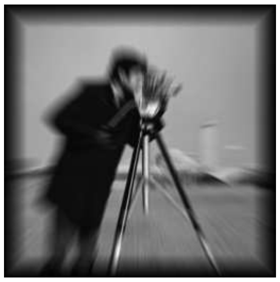
Even if matrix B_l is definite positive, and thus invertible, the inversion of the blur inevitably amplifies errors due imprecision in the modeling. Therefore we replace the naive inverse B_l^{-1} with the regularized Tikhonov inverse matrix $\widetilde{B_l}$,

$$\widetilde{B_l} = (B_l^T B_l + \alpha^2 I)^{-1} B_l^T \,, \tag{4.12}$$

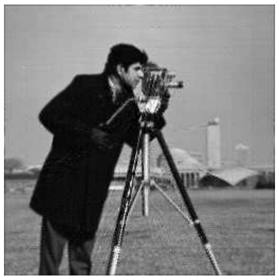
where $\alpha > 0$ is a regularization parameter. The regularized inverse of the blurred observation is then

$$z^{RI} = \mathcal{C}\left(\widetilde{B_l}\mathcal{P}\left(z\right)\right) \,. \tag{4.13}$$

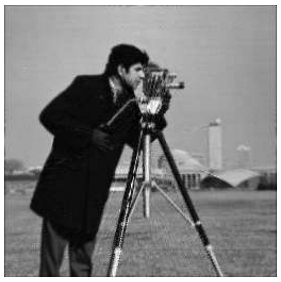
Figure 4.4 shows the blur inversion for the blurred image depicted in Figure 4.4(a). In Figure 4.4(b) and Figure 4.4(c) we can see the inverse by multiplication with the matrix B_l^{-1} and with the matrix $\widetilde{B_l}$, respectively. Both images show artifacts along some rows and columns, but these artifacts are stronger



(a) Radial blurred image: the blur center e is at the image center and $l = 0.25$.



(b) Blur inversion with naive inverse B_l^{-1}, RMSE $= 11.84$.



(c) Regularized inverse z^{RI} via Tikhonov regularization of $\widetilde{B}_l$ (4.12), RMSE $= 11.41$.

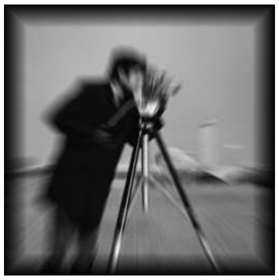
Figure 4.4.: Blur inversion with naive inverse and Tikhonov regularized inverse.

in Figure 4.4(b) than in Figure 4.4(c). These artifacts are due to the rounding errors in the coordinate transformations $\mathcal{P}$ and $\mathcal{C}$. The root mean square error (RMSE) of the image restored with the naive inversion is 11.84, while in the case of Tikhonov regularization ($\alpha = 0.005$) the RMSE is 11.41. The regularized inverse is able to reduce these artifacts and it is crucial for dealing with image noise.

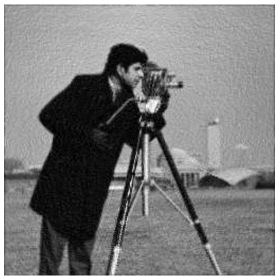
4.4. Noise and noise attenuation

For simplicity, we model the observation errors as an additive white Gaussian noise (AWGN) term in the Cartesian domain. The observation equation is as follows

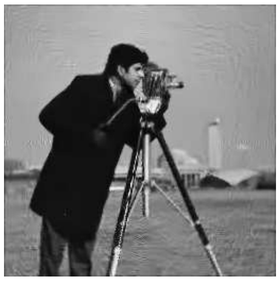
$$z = \mathcal{C}\left(B_l Y\right) + \eta\,, \tag{4.14}$$



(a) Radial blurred noisy image: the blur center e corresponds to the image center, $l = 0.25$, and $\sigma = 0.004 \times 255$. Noise is hardly perceptible in z.



(b) The regularized inverse z^{RI} presents heavy noise, RMSE= 19.94.



(c) Final restored image, obtained after Pointwise SA-DCT filtering of the regularized inverse z^{RI} shown in Figure 4.5(b): RMSE= 12.81.

Figure 4.5.: Algorithm performance on a radial blurred image z with noise.

where $\eta(\cdot) \sim \mathcal{N}(0, \sigma^2)$.

Figure 4.5 illustrates the blur inversion on a noisy image. The Gaussian noise has been added to the blurred image according to Equation (4.14), with a standard deviation $\sigma = 0.004 \times 255$, which is a very low noise and almost not perceptible in Figure 4.5(a). However, as one can clearly see in Figure 4.5(b), in the regularized inverse z^{RI} the noise has been significantly amplified and the restoration performance decreased (RMSE= 19.94).

Note that the noise characteristics change significantly after the coordinate transformations and the multiplication against the inverse of the blurring matrix. Even when the noise in z^{RI} is Gaussian distributed, which happens if

$\mathcal{P}$ and $\mathcal{C}$ exploit linear interpolation, the noise may not be independent nor identically distributed.

4.4.1. Image Restoration Algorithm

Our image restoration algorithm is based on a two step approach like many state-of-the-art deconvolution algorithms [29, 33, 70, 46]. The first step is the blur inversion. In the cited works, this consists of regularized deconvolution in Fourier domain. The noise in the deconvolved image is Gaussian colored noise and it is characterized by its power spectral density (PSD), which (up to a σ^2 scaling factor) is determined by the PSF. The second step is the removal of the colored noise. This is performed by conventional filtering techniques, such as transform-domain shrinkage [29, 33, 70]. In the cited works the noise PSD is used for computing threshold for the shrinkage.

In our case, the blur inversion consists of multiplying the image in polar domain against the matrix $\widetilde{B_l}$. Because of the coordinate transformations and because the blur is not convolutional, the noise in z^{RI} cannot simply described by a PSD. However, we treat the noise in z^{RI} as colored Gaussian and we use an approximate model for the noise PSD. The denoising is then performed by the Pointwise Shape-Adaptive DCT (SA-DCT) filtering algorithm [28, 29, 30]. This algorithm had been used earlier for Gaussian colored noise removal in deblurring [29] and inverse halftoning applications [17].

4.4.2. Noise PSD Modeling

Calculating the noise statistics in z^{RI} is a demanding task as $\mathcal{P}$ and $\mathcal{C}$ transform differently the image pixels, according to their location on the image plane. Moreover, any analytical result would depend on the interpolation methods used in $\mathcal{P}$ and $\mathcal{C}$. In what follows, we consider the transforms sequence $\mathcal{C}\left(\widetilde{B_l}\mathcal{P}\left(\cdot\right)\right)$ (i.e. Cartesian-to-polar, blur inversion, and polar-to Cartesian) as an input-output system and we study the noise statistics by a Monte Carlo approach. That is, we generate n independent realizations of standard Gaussian white noise η_i, $\eta_i(\cdot) \sim \mathcal{N}(0,1),\ i = 1,\ldots,n$, and, assuming e at the image center, we process each of them with the input-output system $\mathcal{C}\left(\widetilde{B_l}\mathcal{P}\left(\cdot\right)\right)$. More precisely, let η_i be the system input,

$$\eta'_{i,l} = \mathcal{C}\left(\widetilde{B_l}\mathcal{P}\left(\eta_i\right)\right), \quad i = 1,\ldots,n, \tag{4.15}$$

be the system output. In what follows, we restrict ourselves to operators $\mathcal{P}$ and $\mathcal{C}$ which are linear.

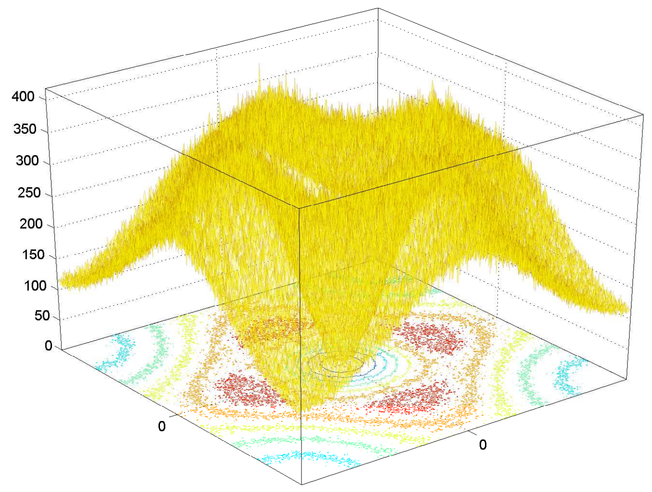


Figure 4.6.: The noise PSD $N_{0.25}$. Its values have been computed using Equations (4.15) and (4.16).

The noise PSD N_l is computed as the sample variance of the Fourier transforms of the n realizations of $\eta'_{i,l}$, $i = 1, \dots, n$, along each frequency:

$$N_l(\omega) = \text{var}\left\{\mathcal{F}(\eta'_{i,l})(\omega)\right\}, \; \omega \in \Omega, \tag{4.16}$$

where $\mathcal{F}$ denotes the 2D Fourier transform and ω represents the frequencies in the Fourier domain Ω. We remark that N_l is computed from n realizations of noise with unitary variance. Whenever the actual noise has variance $\sigma^2 \neq 1$, the noise PSD needs to be scaled accordingly and becomes $\sigma^2 N_l$. Note also that the computed PSD depends on the blur extent l.

By taking into account the PSD for the denoising, we implicitly treat the noise in z^{RI} as Gaussian colored noise. This may not necessarily hold, however this approximate noise description facilitates the application of the image denoising algorithm. In practice, this pragmatic approach yields satisfactory results from experimental evidence.

Figure 4.6 presents the noise PSD $N_{0.25}$, obtained by inverting $n = 3000$ noise realizations with radial blur of extent $l = 0.25$. One can see that despite obvious symmetries within the PSD, the middle horizontal/vertical cross-sections are considerably different from the two diagonal ones. These cross-sections, which we denote v_l (horizonal/vertical) and d_l (diagonals) are shown

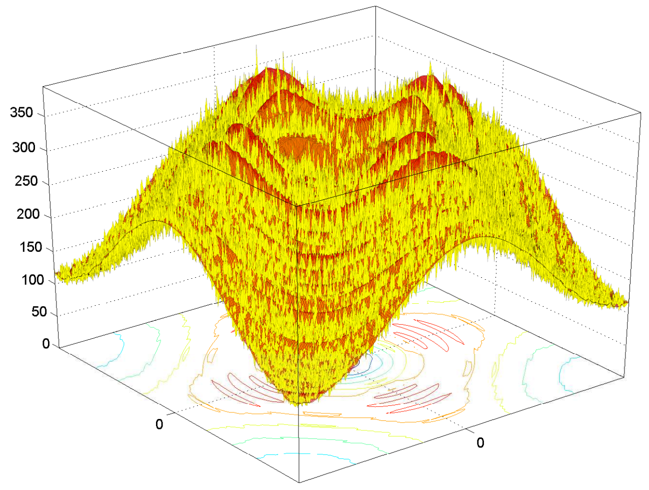


Figure 4.7.: The function $\mathcal{S}_{0.25}$ (in red), approximating $N_{0.25}$ of Figure 4.6 (in yellow).

in Figure 4.9.

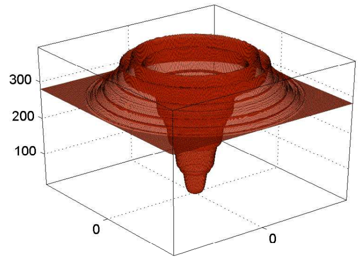
Since the noise PSD varies depending on l, we repeat the Monte Carlo procedure considering several different values of the blur extent parameter. Furthermore, for practical purposes, we replace the sampled N_l with an approximated function $\mathcal{S}_l$ generated by convex interpolation of the horizontal/vertical and diagonal cross-sections v_l and d_l of N_l. In our simulations the blur extents are $l = 1/120, \ldots, 110/120$. Figure 4.10 illustrates the cross-sections v_l and d_l when varying the blur extent l.

Finally $\mathcal{S}_l$, the function approximating N_l, is defined as

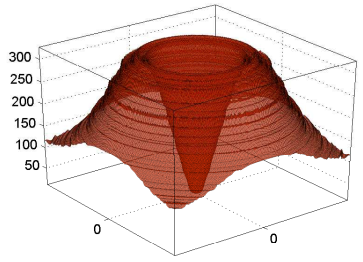
$$\mathcal{S}_l(\omega) = \cos^2(2\beta)\mathcal{S}_l^v(\omega) + \sin^2(2\beta)\mathcal{S}_l^d(\omega), \quad \omega \in \Omega\,, \tag{4.17}$$

where $\beta = \arctan(\frac{\omega_2}{\omega_1})$ is the angular component of the frequency ω and $\mathcal{S}_l^v$ and $\mathcal{S}_l^d$ are surfaces of revolution generated rotating around the origin v_l and d_l, respectively. Figures 4.8(a) and 4.8(b) show the surfaces $\mathcal{S}_{0.25}^v$ and $\mathcal{S}_{0.25}^d$, respectively, while Figure 4.8(c) shows $\mathcal{S}_{0.25}$ computed according to Equation (4.17). The quality of the approximation of $\mathcal{S}_{0.25}$ to $N_{0.25}$ is illustrated in Figure 4.7.

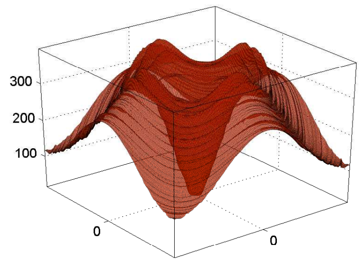
Figure 4.5(c) illustrates the image restoration performance of proposed approach when $\mathcal{S}_{0.25}$ is used as the noise PSD for the Pointwise SA-DCT denois-



(a) Surface $\mathcal{S}^v_{0.25}$.



(b) Surface $\mathcal{S}^d_{0.25}$.



(c) Surface $\mathcal{S}_{0.25}$.

Figure 4.8.: Top: examples of surfaces of revolution $\mathcal{S}^v_l$ and $\mathcal{S}^d_l$ generated from the cross-sections v_l and d_l. Bottom: $\mathcal{S}_l$, approximation of the noise PSD N_l, obtained as the convex combination of $\mathcal{S}^v_l$ and $\mathcal{S}^d_l$ defined by Equation (4.17).

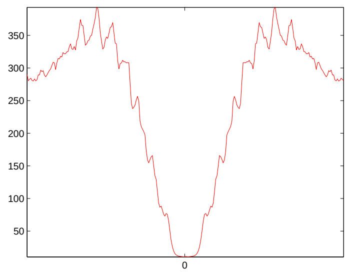
ing. The RMSE of the restored image is 12.81.

4.5. Experiments

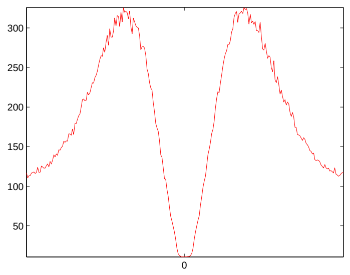
We present simulation results obtained for a set of four common grayscale test images of size 256×256. As in Equation (4.14), the blurred noisy observation are generated from the original image y as

$$z = \mathcal{C}\left(B_l \mathcal{P}(y)\right) + \eta \,. \tag{4.18}$$

We use discrete transforms $\mathcal{P}$ and $\mathcal{C}$ based on bilinear interpolation. Let us remark that both these transforms introduce errors in the observation z, seriously impairing the restoration quality even in the noise free case. The size of



(a) The values $v_{0.25}$ corresponding to the PSD values on the vertical axis of Ω.



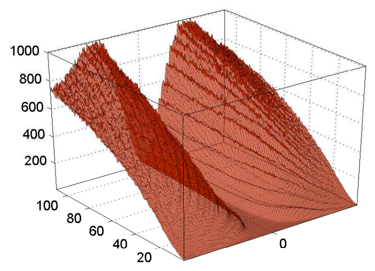
(b) The values $d_{0.25}$ corresponding to the PSD values on the diagonal of Ω.

Figure 4.9.: Examples of v_l and d_l generating the surfaces $\mathcal{S}_l^v$ and $\mathcal{S}_l^d$.

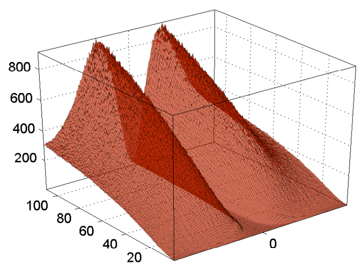
the polar domain is determined as in the work by Ribaric *et al.* [77] on images blurred because of camera rotation. In particular, we use $r = \max_{x \in \mathcal{X}}(|x - e|)$ and $\tau = \left\lceil 2\pi / \arcsin\left(\sqrt{2}/r\right)\right\rceil$, where $\lceil \cdot \rceil$ is the rounding to the nearest larger or equal integer. We limited our tests to blur having the blur center e at the image center: whenever e is in a different location on the image plane, in order to apply the restoration algorithm, the image has to be accordingly padded and shifted.

According to a common practice when testing deblurring algorithms, we add noise with a small variance to images where the blur extent is large and noise with a higher variance to images where the blur extent is small. Thus, we mimic the situation where images are acquired with different exposure times during the same camera motion. When the camera undergoes a fixed translation, images acquired with a long exposure are typically heavily blurred, while the noise affecting these images is small. On the contrary, images acquired with a short exposure, show weaker blur and stronger noise. Table 4.1 shows the pairs blur extent/noise standard deviation used for generating the blurred noisy observations. For all these cases, the Tikhonov regularization parameter, α (4.12), is fixed to $\alpha = 0.005$.

Table 4.2 shows the RMSE of the restored images w.r.t. the original image. The performance of the noise attenuation are illustrated in Figure 9, while Figures 4.12 and 4.13 shows some of the images, before and after restoration. As one may expect, the algorithm performance decreases with the increase either of the blur extent l or of the noise standard deviation. The latter appears to have a more substantial impact on the quality of the restored image. In particular,



(a) The noise PSD values at frequencies on the vertical axis of Ω when varying the parameter $l = 1/120, \ldots, 110/120$.



(b) The noise PSD values at frequencies on the diagonals of Ω when varying the parameter $l = 1/120, \ldots, 110/120$.

Figure 4.10.: PSD values used for computing $\mathcal{S}_l$, with $l = 1/120, \ldots, 110/120$.

the RMSE corresponding to Exp.6, which is the noise-free experiment, is always significantly lower than the RMSE in Exp.5, where the observations are generated with the same blur but with $\sigma = 0.002 \times 255$. Finally, let us observe that the restored images of Figures 4.12 and 4.13 show some artifacts along the radial lines; these are not due to noise but rather to interpolation errors introduced by the coordinate transformations $\mathcal{P}$ and $\mathcal{C}$: indeed such artifacts appear also in the noise-free experiments.

Table 4.3 gives the execution times of each separate stage of the algorithm, applied on a 256×256 image (Cameraman). The times correspond to our Matlab implementation running on a computer with AMD 64 1.81-GHz processor. As one can see from the Table 4.3, the impact of coordinate transformations on the overall execution time is of negligible, as nearly all the time is actually taken by the Pointwise SA-DCT denoising step.

4.6. Conclusions

In this chapter we presented a novel restoration algorithm for noisy radial blurred images. The restoration algorithm includes two main steps: the blur inversion in the polar domain, and the noise removal in the Cartesian domain. The denoising part consists of an adaptation of a spatially adaptive transform-based denoising method, namely the Pointwise SA-DCT filter [30, 29].

Experimental results with synthetically generated observations show that

Experiment	$60\,l$	$\sigma/255$
Exp.1	5	0.006
Exp.2	10	0.004
Exp.3	**15**	**0.004**
Exp.4	20	0.002
Exp.5	25	0.002
Exp.6	25	0

Table 4.1.: Experimental settings: blur extent and noise standard deviation values used for testing the proposed algorithm. The third row (in bold) shows the parameters used in the examples of Figures 4.4 and 4.5.

	Exp.1	Exp.2	Exp.3	Exp.4	Exp.5	Exp.6
Cameraman	11.27	11.98	12.82	12.24	12.52	12.33
House	6.93	6.93	7.33	6.50	6.63	6.94
Aerial	11.13	11.43	12.34	10.69	11.09	9.43
Peppers	8.79	8.81	9.35	8.46	8.63	8.42

Table 4.2.: Root mean squared error (RMSE) of each restored image in the experimental settings of Table 4.1. Examples of the restoration quality are shown in Figures 4.12 and 4.13.

the denoising step improves significantly the restoration performance. At the same time, it emerges the need of an accurate model of noise statistics after both blur inversion and coordinate transformations. In the future we will investigate algorithms for estimating e and l from a given blurred image, in order to combine the estimation and the restoration procedures together into a blind deblurring algorithm.

Algorithm part	time (sec)
Cartesian-to-polar	0.28
Blur inversion	0.07
Polar-to-Cartesian	0.12
SA-DCT filtering	4.91

Table 4.3.: Execution times for a 256 $\times$ 256 test image on a AMD 64 1.81-GHz.

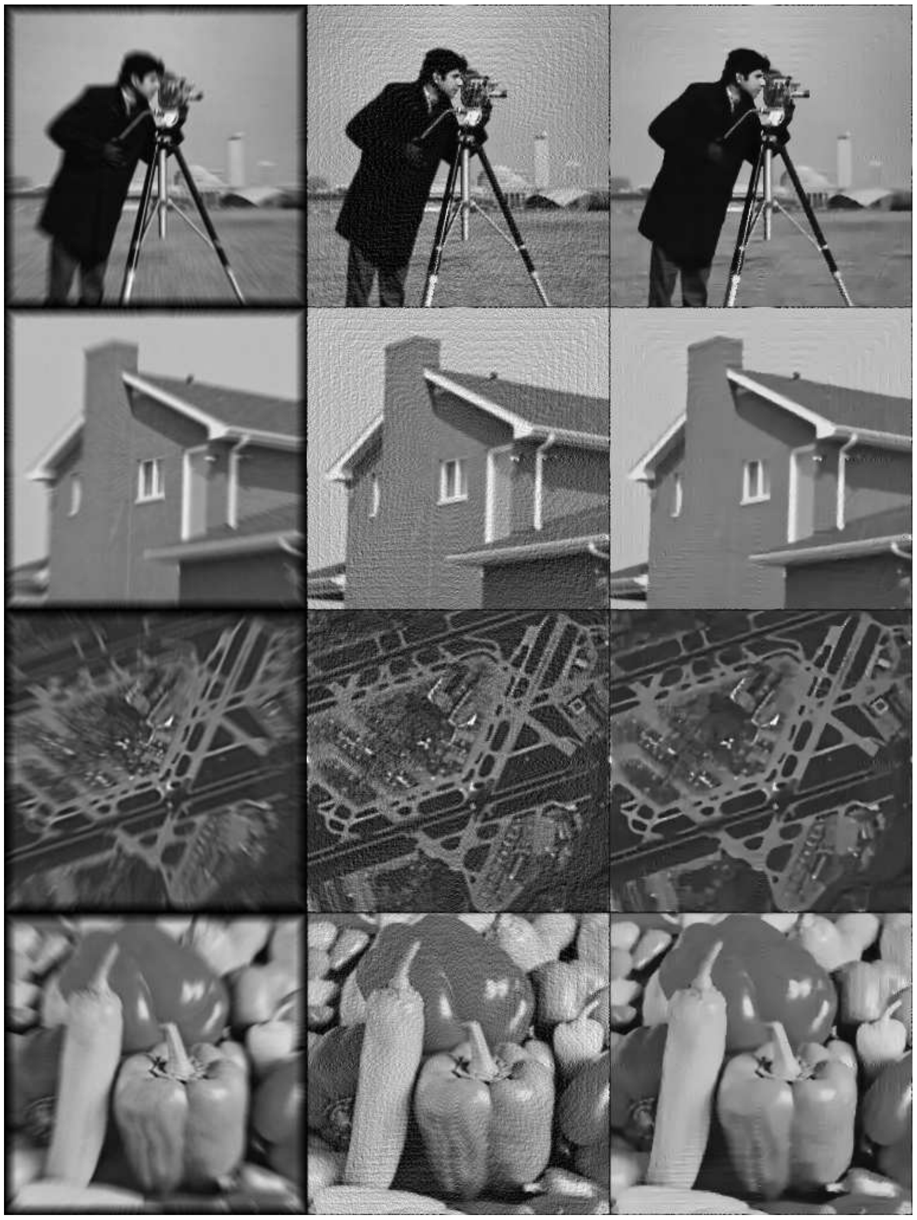


Figure 4.11.: Experiment 1: noise attenuation. The first column shows the observations, the second column the corresponding regularized inverses z^{RI}, and the third column shows the restored images. Restoration performance are listed in Table 4.2.

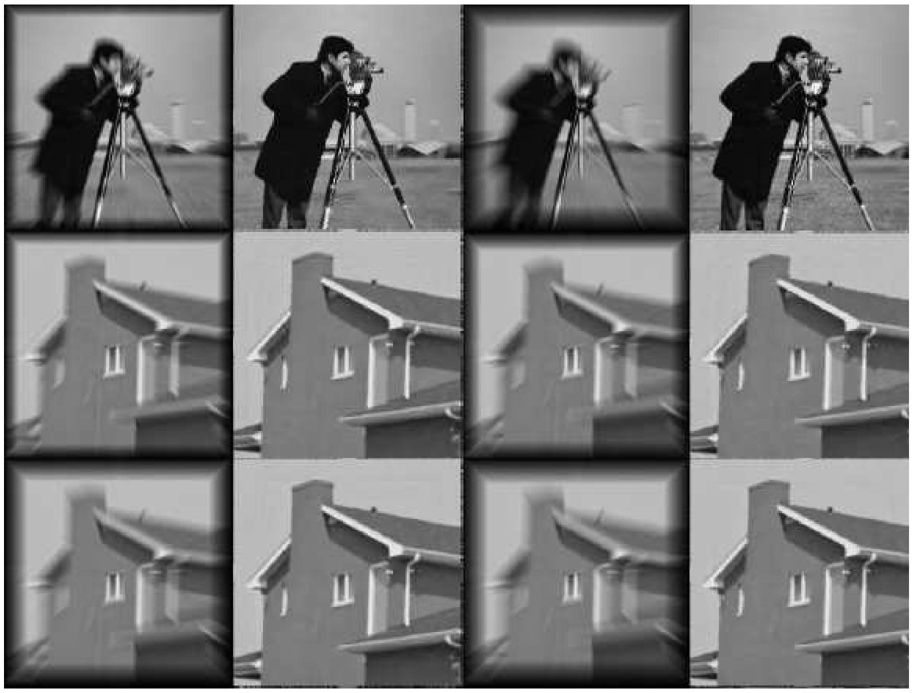


Figure 4.12.: Blurred and noisy images (first and third column) and restored images (second and forth column) obtained with the proposed method. See Table 4.1 for parameters used in the experiments and Table 4.2 for restoration performance in terms of RSME. Row 1: “Cameraman” experiments 2 and 6. Row 2: “ House ” experiments 2 and 3. Row 3: “House” experiments 4 and 5.

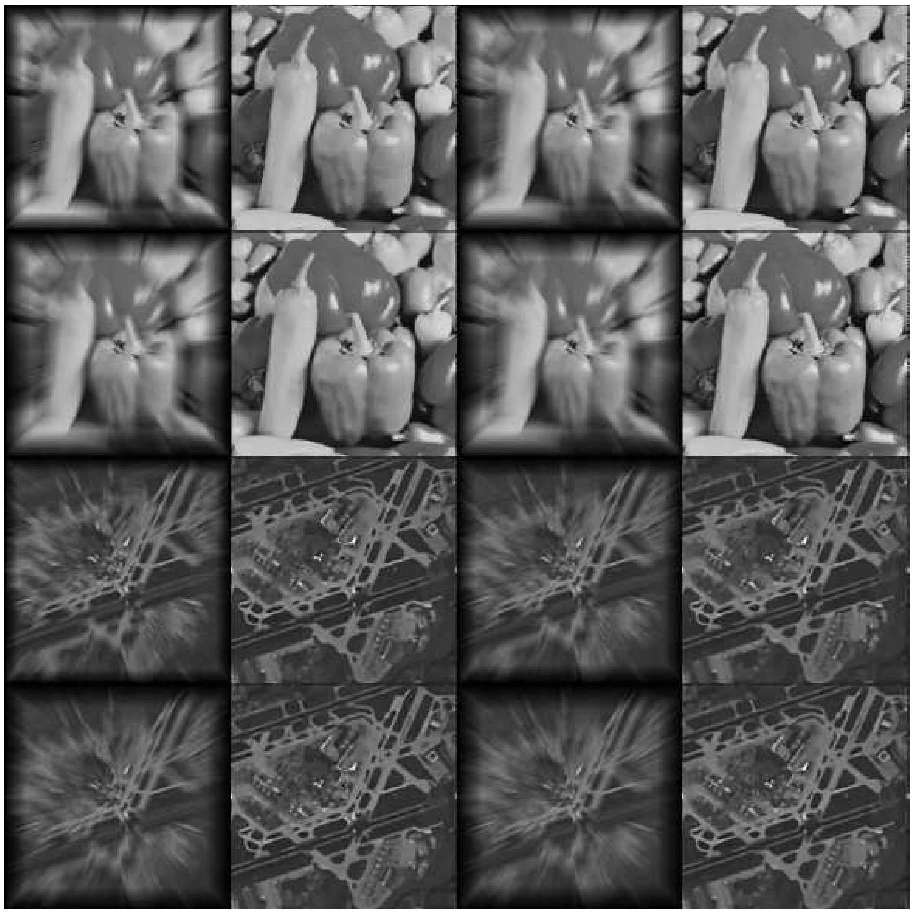


Figure 4.13.: Blurred and noisy images (first and third column) and restored images (second and forth column) obtained with the proposed method. See Table 4.1 for parameters used in the experiments and Table 4.2 for restoration performance in terms of RSME. Row 4: “Peppers” experiments 3 and 4. Row 5: “Peppers” experiments 5 and 6. Row 6: “Aerial” experiments 2 and 3. Row 7: “Aerial” experiments 4 and 5.

5. Structure From Motion Blur

This chapter discusses the capabilities of estimating the 3D scene structure from a single motion blurred image. We present an algorithm for estimating the scene depth from a single radial blurred image, that do not require any user interaction. While we are not able to cope with more general camera motions, we focus on radial blur and we discuss the shortcomings and the advantages of estimating scene depth from motion blur.

We also briefly describe some methods that combines priors on the image content with motion blur analysis, in order to cope with more complicated 3D motion.

5.1. Structure From Motion Blur

Several techniques in literature have been proposed for estimating the 3D scene structure by exploiting blur. In some cases, the image blur can be directly related to the scene depth and therefore, when properly analyzed, the blur allows to infer the 3D scene.

There are several methods in literature concerning the depth estimation from out of focus blur: the image focus depends on the scene depth and thus local estimates of focus blur can be used to infer the scene depth. These methods are known as *shape from defocus* and *shape from focus*. Shape from defocus methods infer the scene depth given a set of defocused images of the same scene. In shape from focus methods the camera focus is actively changed in order to estimate the scene depth. In turn, shape from defocus methods are further classified in active and passive methods according to the use of structured light or not. See [23] for an exhaustive review on depth estimation methods from image blur. Recently, a shape from defocus method from a single image acquired with a coded aperture camera has been proposed [53]. This method performs also the restoration of out of focus images, where blur varies according to the scene depth. Typically depth from defocus methods exploit multiple images.

Motion blur is determined both by the 3D camera motion and the scene structure, therefore can be also used for 3D reconstruction. A motion blurred image is obtained by the integration of several *sub-images* y_t, each one captured by the camera having the viewpoint in a different position, see Equation (3.1),

$$z(x) = \int_0^T y_t(x)dt + \eta(x), \quad x = (x_1, x_2)\,, \tag{5.1}$$

where the sub-images y_0 and y_T form a pair of views, corresponding to the time instants where the camera shutter opens and closes, respectively.

Let us assume that the camera intrinsic parameters are known, while the extrinsic parameters are not, and discuss the issue of reconstructing the scene from a single motion blurred image. If we are able to estimate the trajectories followed by some pixel on the image plane during the exposure, and if we can associate to these trajectories matching between y_0 and y_T, we can reproduce a stereo vision system from a single motion blurred image.

Let γ_i be the trajectory associated with the PSF at x_i. Examples of these trajectories are the corner displacement vectors presented in Chapter 1. The basic idea is to estimate the pairs $(\gamma(0), \gamma(T))$, formed by the extremities of γ_i, and to associate them the correspondences one would obtain by feature matching between the sub-images y_0 and y_T. These allow us to reconstruct the 3D camera motion and the scene depth via standard epipolar geometry techniques [36].

The following problems are encountered when trying to extract stereo vision from a single blurred image:

Trajectories are not straight lines. While the correspondences determined from two focused views are oriented segments (i.e. vectors, optical flow descriptors), the blur trajectories may not be straight segments. The estimation of trajectories which are far from being straight lines could not be trivial. Even assuming a short exposure time and fast camera motion may not guarantee rectilinear trajectories. In fact, blur produced by camera rotation shows smears which are arcs of conic sections, as described in Chapter 6, which makes the estimation of correspondence pairs more difficult.

Correspondence Pairs Orientation. Once a blur trajectory γ has been correctly estimated, it is not possible to determine which one of the end points $\gamma(0)$ and $\gamma(T)$, belongs to y_0 and which one belongs to y_T. In fact, there is no way to estimate from the resulting blurred image which is the orientation of the underlying motion, as already discussed in Section 1.8. Unluckily, 3D reconstruction techniques based on feature matching in image pairs require the orientations of the determined correspondences, which are always known (up to a swap of all the orientations) as features of each pair are taken from a different image.

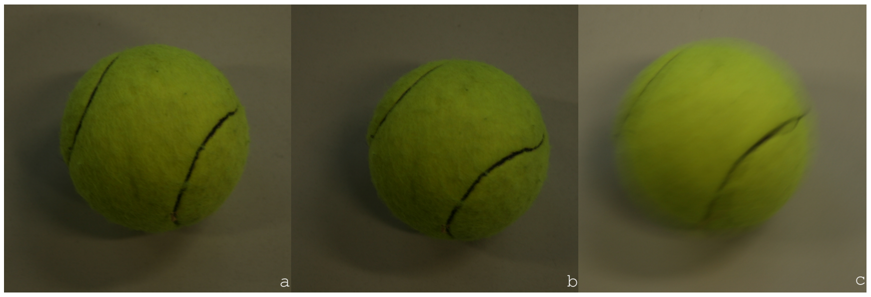


Figure 5.1.: Texture in figures **a** and **b** show high self-similarity thus matching feature between **a** and **b** may be complicated. The correspondence problem does not hold in when analyzing blurred smears **c**.

A solution could be taking into account the two orientations for each correspondence and, enforcing additional constraints, such as those deriving from rigidity of camera motion, use the Ransac algorithm [27] to determine the correct orientations.

Although these shortcomings, estimating the correspondence pairs from blur presents some advantages w.r.t. to feature matching based algorithm. Let us consider for example two frames of a moving object containing repetitive textures, as presented in Figure 5.1. In this case, because of high self-similarity of the image content, it is difficult to correctly estimate matches between the two frames, when no further assumptions can be made on the matches. From a picture, captured with a longer camera exposure of the same moving scene, it could be easier to estimate the correspondence pairs by analyzing blur in image patches, although the orientation ambiguity still holds. Roughly speaking, in some cases the estimation of correspondence pairs from a blurred image does not present the matching problem at the cost of losing information about pairs orientation.

Typically, there are few reliable correspondence pairs estimates that can be obtained from a blurred image. Their number typically decreases when the blur smears are not straight line segments. The 3D reconstruction from a single image can be then addressed assuming that there are several blurred images (possibly in a sequence), and enforcing some prior on the scene or on the original image, or possibly assuming that a noisy image of the same scene is also available. These approaches however have not been investigated in this thesis work.

We focus on depth reconstruction from a single blurred image and we enforce some constraints on the camera motion. We restrict to radially blurred images, i.e. images acquired from a purely translating camera, as discussed in

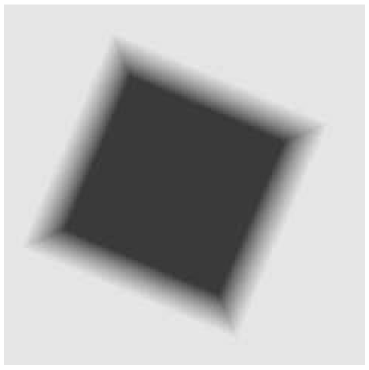
Chapter 3. Depth estimation from radially blurred images is simpler than motion blurred images, as the correspondence pairs orientation can be neglected; moreover the blur smears are straight lines.

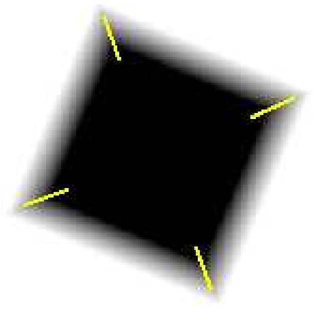
A motion blurred image acquired during a pure rotation with the camera viewpoint lying on the rotation axis, does not allow to infer the scene depth as the camera viewpoint does not move, and thus multiple view reconstruction can no be done (see Chapter 6).

5.2. Depth Estimation From A Radial Blurred Image

Let assume the camera is purely translating, as illustrated in Figure 3.1. In this case the local blur estimates can be associated to the correspondences between images y_0 and y_T. In fact, from the constraints on the camera motion ([36] and Chapter 3), it follows that the matchings between y_0 and y_T are directed and oriented toward the epipole. Thus, also the local blur estimates orientation can be inferred and the blur estimates are equivalent to the feature matches. Moreover, the blur estimates obtained using one of the methods presented in Chapter 3 are reliable as the smears are straight line segments.

Figure 5.2(a) shows a synthetic image produced with a ray-tracer software Pov-Ray [41] by averaging frames rendered from a translating camera. These images have been rendered according to the procedure described in Section 3.5.


(a) A Synthetic Radially blurred image, used for estimating depth..


(b) Epipole estimation using blur analysis at corners

The algorithm outline can be represented in three steps

Epipole Estimation. This is performed exploiting the algorithm presented in Chapter 3 or the epipole location on the image plane can possibly be assumed known in specific applications.

Correspondence Pairs Estimation. Once the epipole has been estimated (for example by exploiting corner regions), the blur direction is determined at any image pixel. Then, also blurred edge areas can be used for determining correspondence pairs $(\gamma(0), \gamma(T))$. We randomly select pixels belonging to the Γ masks described in Section 2.1, and estimate the blur extents within a region elongated along the blur direction of these pixels estimated using Equation (1.8), as only one parameter is unknown. Other techniques can be used such as autocorrelation-based like in [93] or techniques based on analysis of profiles along blur directions [8]. Note that the estimated correspondence pairs are indeed vectors, as their orientation, up to a global swap, is determined by the epipole. Figure 5.2 shows these vectors. Each green point represents the position of a pixel in the sub-image y_0, whereas the corresponding blue point identifies its position in y_T.

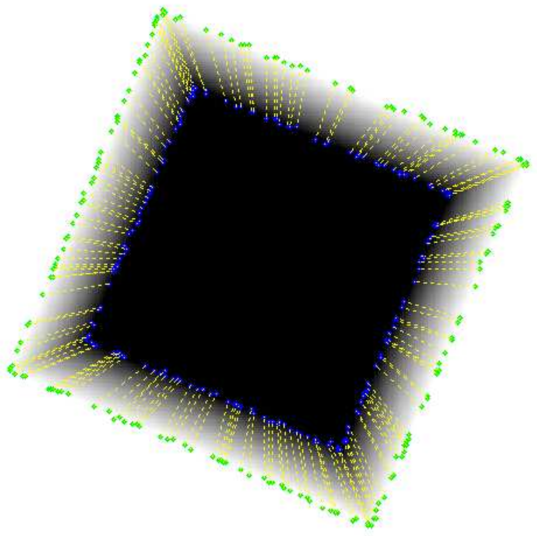


Figure 5.2.: Estimated correspondence pairs from Figure 5.2(a). Green and Blue points indicate the correspondence.

Depth Reconstruction By assuming a still camera and a translating object, the 3D scene can be reconstructed with a triangulation [36]. Figure 5.3

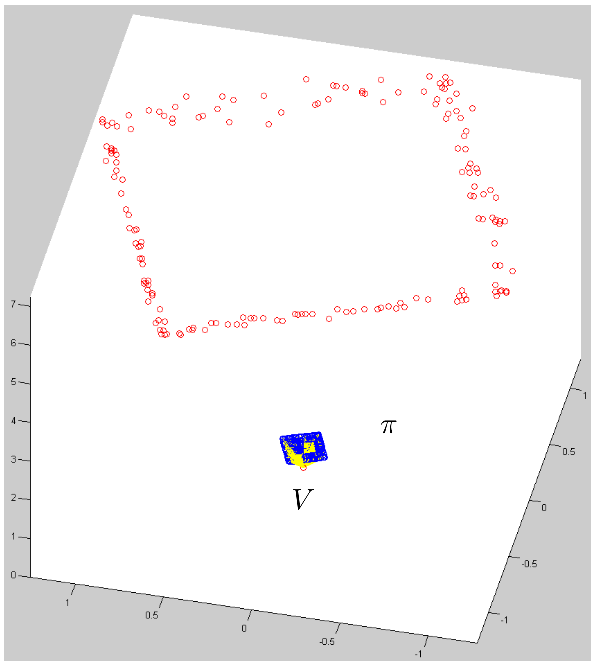


Figure 5.3.: Estimated Depth from Figure 5.2(a). V is the camera viewpoint and π the image plane. Error was 5% the ground truth.

represents the depth estimates. Two planes are fitted between two clouds of points, each one represents the initial and final position of the square plane in the scene. Assuming a static camera, the initial and final positions of the scene plane can be reconstructed. Figure 5.4 shows the depth map estimated from Figure 5.2.

Depth estimates are far from being accurate as the mean absolute error of the point clouds is 3% of the true depth for the initial position and 5% of the true depth for the final position. This is mostly due to the fact that there is no a significant change in subimages y_0 and y_T as the perspective is the same. Note that the same procedure works also when $e \rightarrow \infty$, as the depth influences at the same way the blur extents, while leaving blur direction constant.

5.3. Sphere Full 3D Motion Reconstruction From a Single Blurred Image

Whenever the scene content is known (even partially), it is possible to design custom algorithms in order to estimate the scene depth or motion. In this

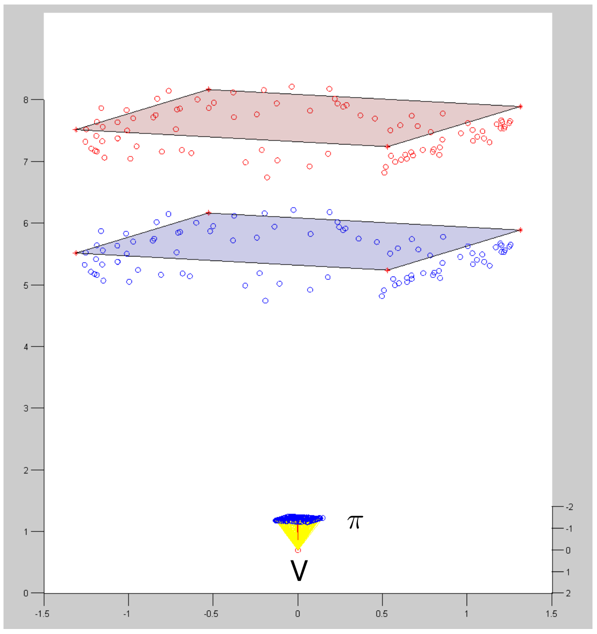


Figure 5.4.: Estimated Depth from Figure 5.2(a). Assuming a static camera the initial and final position of the scene plane can be reconstructed. V is the camera viewpoint and π the image plane. Error was 3% and 5% of the ground truth value for the closest and futher position of the scene plane, respectively.

section we outline algorithms targeted for images depicting moving sphere [8] and [9]. These algorithms have been devised for high resolution images of sports events depicting a moving ball. These algorithms are also based on the image formation model of Equation 5.1.

A sphere contour is projected into an ellipse which allows, up to a scale factor, to reconstruct the 3D sphere position. Typically the scale factor is fixed by means of the known ball radius. If blur is correctly handled the 3D sphere position and translation can be estimated directly from a single blurred image.

We assume that the ball is monochromatic and that it is moving on a monochromatic background. This situation corresponds to the transparency (or the alpha matte) of the moving ball and gives, in each image pixel, the percentage of exposure time it has been covered by the ball. Typically the transparency map is assumed known [44, 72], as this can be estimated thanks to user interaction [55, 79, 83, 90] or thanks to the knowledge of the background [31, 66, 82].

The ball 3D displacement is estimated from the transparency map by fitting two ellipses c_1 and c_2, representing ball position at the opening and at the closure of the shutter, see Figure 5.5.

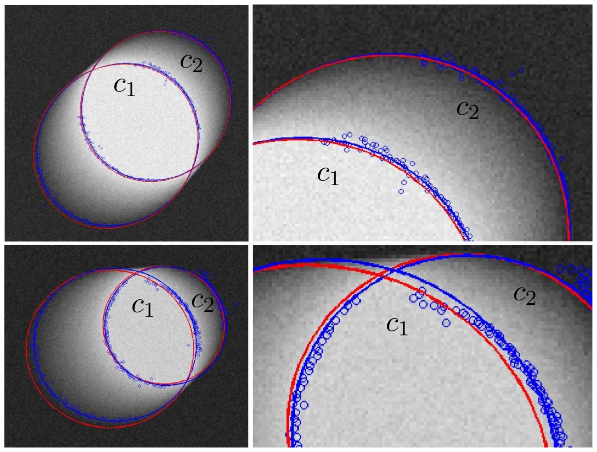


Figure 5.5.: The ellipses c_1 and c_2 represent the ball at the opening and closure of the shutter. From these ellipses, it is possible thus to reconstruct the initial and final position of the sphere in the 3D scene.

The analytical expressions of c_1 and c_2 are determined by fitting two ellipses to points of Figure 5.5, which are determined by analyzing 1D image profiles of the blurred ball with the techniques presented in [8], see Figure 5.6. The profile analysis procedure corresponds to the estimation of correspondence pairs $(\gamma(0), \gamma(T))$ of Section 5.1. Figure 5.7 shows algorithm performances on camera images.

Once the 3D ball displacement is known it is possible to estimate the ball rotation axis and the spin velocity [9]. The blur on image regions depicting the ball surface is analyzed and, exploiting geometrical constraints deriving from sphere motion, it is possible to estimate the 3D position of the rotation axis. For a complete description of the algorithm, please refer to [9]. Figure 5.8 shows rotation axis estimated from camera images.

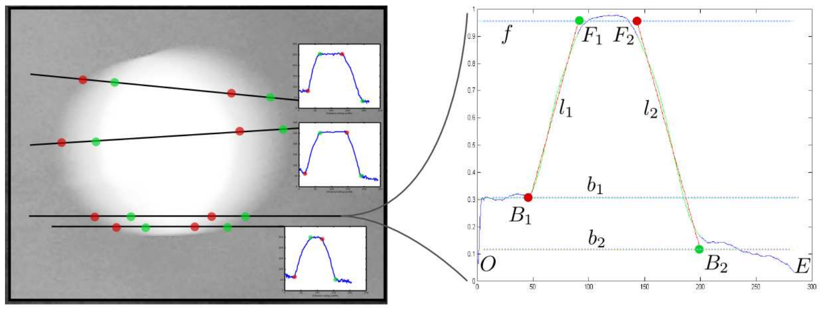


Figure 5.6.: Procedure used for estimating points of ellipses c_1 and c_2, for a detailed description see [8].

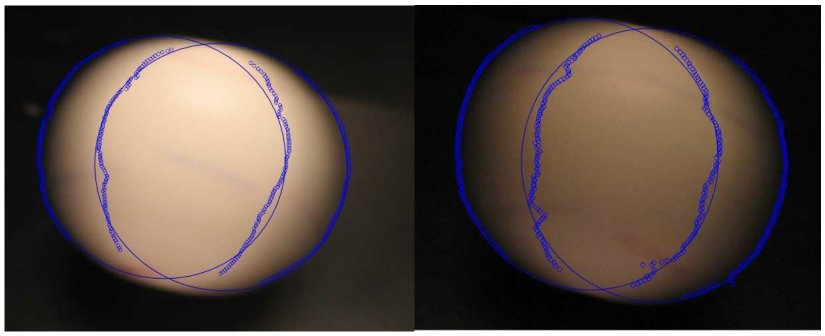


Figure 5.7.: Example of Algorithm [8] performances on camera images.

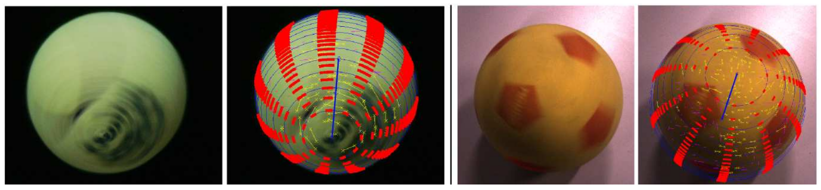


Figure 5.8.: Example all rotation axis estimation: [9] performances on camera images.

6. Estimating Camera Rotation Parameters from a Blurred Image

A fast rotation of the camera during the image acquisition results in a blurred image, which typically shows curved smears. We propose a novel algorithm for estimating both the camera rotation axis and the camera angular speed from a single blurred image. The algorithm is based on local analysis of the blur smears. Contrary to the existing methods, we treat the more general case where the rotation axis is not necessarily orthogonal to the image plane, taking into account the perspective effects that affect the smears.

The algorithm is validated in experiments with synthetic and camera blurred images, providing accurate estimates.

6.1. Rotational Blur Estimation

This paper concerns images corrupted by blur due to camera rotation or to a rotating object in the scene. When the camera or the captured object are purely rotating, the image blur is determined by only two factors: the camera rotation axis a and its angular speed ω. We present a novel algorithm for estimating both a and ω, by analyzing the blur in a single image.

When the camera rotation axis and the angular speed are known, the rotationally blurred image can be restored by image coordinates transformation and blur inversion. In broad terms, the image is transformed from Cartesian to polar coordinates so that the blur becomes space invariant and can be inverted using a deconvolution based algorithm. Estimating correctly the camera rotation axis and its angular speed is therefore crucial for restoring these images as small errors in the polar transformation are amplified by the blur inversion. Moreover, estimating a and ω from a single image can be also of interest for robotic application as these describe the camera ego-motion.

Figure 6.1 shows an image acquired during camera rotation. The shapes of the blur smears show that the blur is space variant. Typically, these are

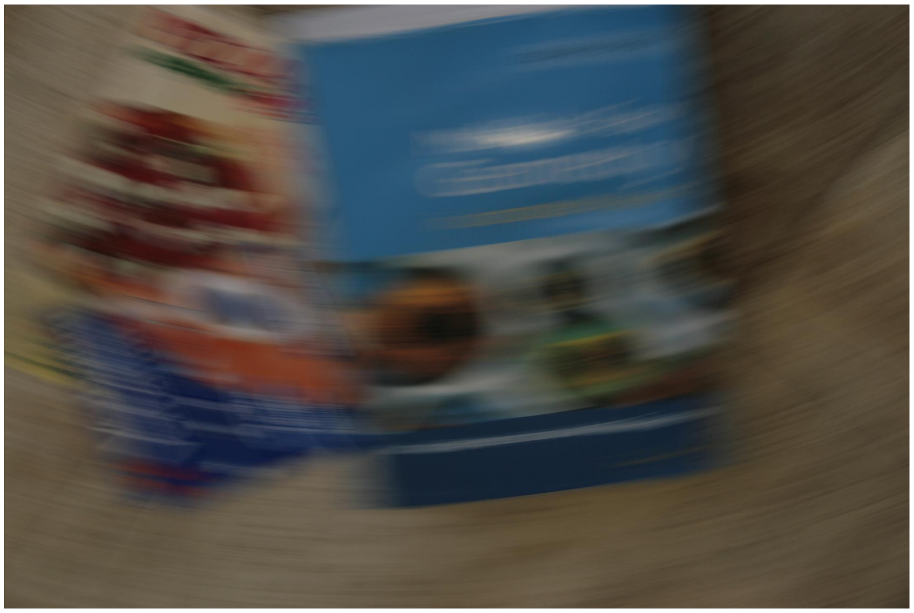


Figure 6.1.: A rotationally blurred image.

assumed arcs of circumferences, all having the same center. However, this approach neglects the perspective effects that occur when the rotation axis is not orthogonal to the image plane. The proposed algorithm estimates the camera rotation axis in the most general case when it is not necessarily orthogonal to the image plane. To the best of our knowledge this issue has never been correctly addressed before.

The early works concerning rotational blur were focused on blur modeling and image restoration. Sawchuk [80], addressing the issue of image restoration in case of spatially variant blur, described a model for the rotational blur. A restoration algorithm specific for rotationally blurred images has been proposed in [84]. Recently, Uçar *et al.* [87] have proposed a fast and parallel implementation for restoration of space variant blurred images, which is tested also on rotational blurred images. All these methods assume that the camera rotation axis is perpendicular to the image plane and that both the angular speed and the intersection between the rotation axis and the image plane are known. A different issue has been addressed in [77], which presents a study on the transformation from the image plane to a polar lattice. Rekleitis [75] provides an algorithm to compute the optical flow from a blurred image, using image tessellation and analysing the Fourier spectrum in small regions where the blur is treated as space invariant. This algorithm has been tested also on rotationally blurred images.

Besides the early works concerning rotational blur modeling and restoration, Hong and Zhang [39] addressed the issue of both rotational blur estimation and removal. Their method is based on an image segmentation along circumferences to estimate the blur and restore the image separately in these subsets. Recently, an algorithm for estimating the camera rotation from a single blurred image has been proposed [51]. The algorithm is meant as a visual gyroscope and it is targeted to an efficient implementation. In particular, this algorithm requires edges in the scene.

Jia [44] proposed an algorithm for estimating and removing the blur due to an object rotation in a single image, taking into account also a translation component. However, this method require the user to mark the end point of blur smears at some pixels and to separate background from foreground in order to estimate the *transparency map* of the blurred object [55].

All the existing methods, concerning both image restoration and blur estimation, assume that the blur smears are arcs of circumferences having the same center. Therefore these methods are accurate only on images where the rotation axis is orthogonal to the image plane.

We present an algorithm for estimating the camera rotation axis and angular speed in the most general case, where the rotation axis is not orthogonal to the image plane. The proposed algorithm is mostly targeting to high accuracy rather than efficiency and does not require the presence of edges in the scene.

6.2. Problem Formulation

We propose an algorithm for estimating the camera rotation axis a and its angular speed ω by analyzing a single blurred image acquired during camera rotation. We assume that the camera is calibrated, the rotation axis a passes through its viewpoint V, i.e. $V \in a$, and w is constant. Figure 6.2.a illustrates the situation typically considered in literature, where the rotation axis is perpendicular to image plane π. The principal point P and the intersection between the image plane and the rotation axis $C = \pi \cap a$ then coincide. Analogous blur is obtained when $a \perp \pi$ and $V \notin a$, but the capture scene is planar and parallel to π [77].

In this work we consider the most general situation, illustrated in Figure 6.2.b, where a is not orthogonal to π and the camera viewpoint $V \in a$.

6.2.1. Image Blur

A *blurring path* is defined as the set of image pixels that a viewing ray intersects during a camera rotation of 2π around axis a. Figure 6.2 illustrates ex-

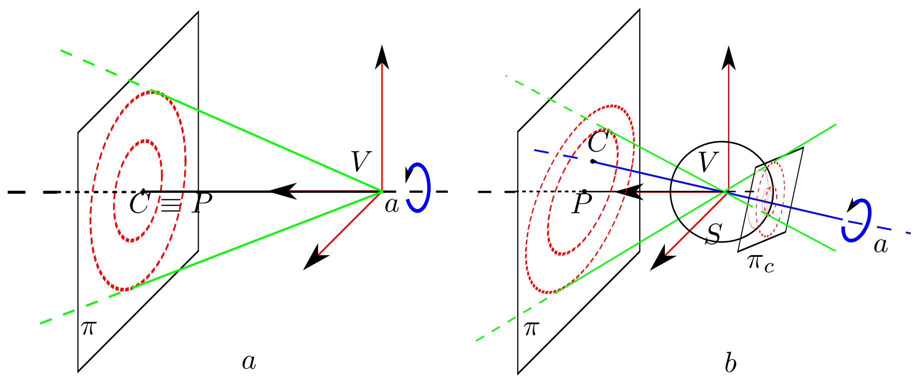


Figure 6.2.: (a) Blurred image formation, $a \perp \pi$. Blurring paths are circumferences. (b) Blurred image formation, a is not orthogonal to image plane π and $V \in a$. Blurring paths are conic sections on the image plane, while they are circular when projected on an ideal spherical sensor and on a plane perpendicular to the rotation axis.

amples of blurring paths. In rotationally blurred images every pixel is merged with neighboring pixels from the same blurring path, see Figure 6.1. The blur is therefore space variant and cannot be represented as a linear shift invariant system. We therefore model the rotational blur by an operator K on the original image y so that the observed (blurred and noisy) image z is [4]

$$z(x) = K\big(y\big)(x) + \eta(x) \quad x = (x_1, x_2) \in X\,, \tag{6.1}$$

where x are the coordinates in the discrete image domain X and $\eta \sim N(0, \sigma_\eta^2)$ is white Gaussian noise. The blur operator K can be written as

$$K\big(y\big)(x) = \int_X k(x,s)y(s)ds\,. \tag{6.2}$$

where $k(x, \bullet)$ is a kernel

$$k(x,\bullet) = A_{\theta,e}(\bullet)\,, \tag{6.3}$$

and $A_{\theta,e}$ corresponds to the point spread function (PSF) at x. $A_{\theta,e}$ is an arc of the blurring path at x, i.e. it is an arc of conic section having tangent line with direction θ and arc length e. The parameters θ, e varies between image pixels according to the rotation axis a. Other blurring effects, such as the out of focus blur, lenses aberrations and camera shake, are not considered.

6.3. The Algorithm

The proposed algorithm consist of three steps: in the first step the lines tangent to blurring paths at some image pixels are estimated (Section 6.3.1). In the second step, these lines are used in a voting procedure for estimating the rotation axis a (Sections 6.3.2 and 6.3.3). The third step consists of the estimation of the angular speed ω (Section 6.3.4).

6.3.1. Blur Tangent Direction Estimation

Image blur is analyzed within N image regions taken around selected pixels $\{x_i\}_{i=1,..,N}$. There are no particular requirements in selecting $\{x_i\}_{i=1,..,N}$, however smooth areas should be avoided, while covering uniformly the image. Therefore we take the local maxima of the Harris corner measure [35], or whenever these do not cover uniformly the image, we take $\{x_i\}_{i=1,..,N}$ on a regular grid.

Blur is analyzed using the approach proposed by Yitzhaky *et al* [94] for estimating the direction of blur "smears" by means of directional derivative filters. This method, proposed for space invariant blurs with PSF having "rectilinear" support, assumes the image isotropic. The blur direction $\widehat{\theta}$ is estimated as the direction of the derivative filter d_θ having minimum ℓ^1 norm response

$$\widehat{\theta} = \arg\min_{\theta\in[0,\pi]} \left(||(d_\theta \circledast z)||_1\right), \tag{6.4}$$

where $||(d_\theta \circledast z)||_1 = \sum_{x\in X} |(d_\theta \circledast z)(x)|$, the ℓ^1 norm.

Equation (6.4) is motivated by the fact that the blur removes all the details and attenuates edges of y along blur direction. Therefore the blur direction can be determined by the directional derivative filter having minimum energy. This method cannot be directly applied to rotationally blurred images, as the blur is not space invariant because in every pixel the circumference approximating the blurring path (i.e the PSF) changes.

At x_i, the center of each region U_i, we estimate the direction θ_i of the line l_i tangent to the blurring path in x_i, as

$$\theta_i = \arg\min_{\theta\in[0,\pi]} \sum_{x_j\in U_i} w_j \big((d_\theta \circledast z)(x_j)\big)^2 . \tag{6.5}$$

where w is a window function rotationally symmetric with respect to the center. By using Gaussian distributed weights, it is possible to reduce the influence of pixels in Equation (6.5) with the distance from x_i. We adopted the 3 tap derivative filters presented in [21] for blur analysis in Equation (6.5).

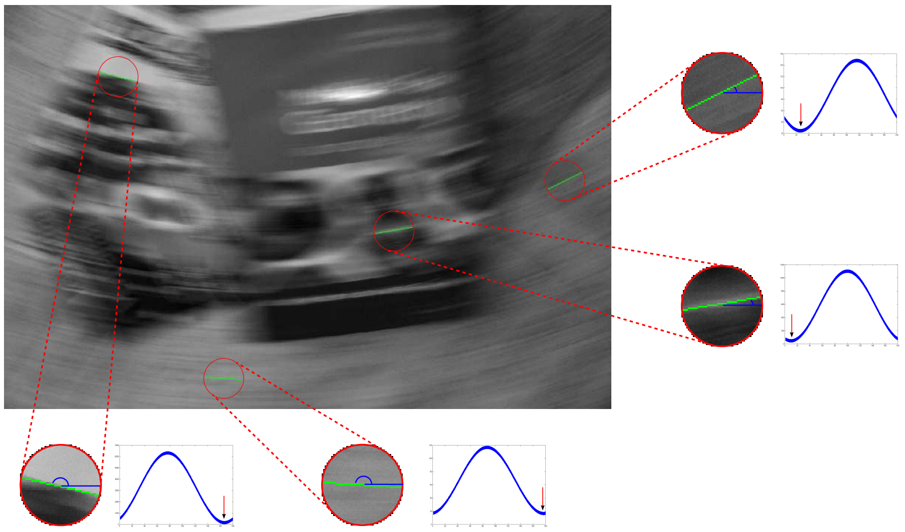


Figure 6.3.: Rotationally blurred image and plots of directional derivatives energy in four regions.

These filters have been selected as they provide good accuracy and as they are separable. Experimentally the ℓ^2 norm gave better results than the ℓ^1 norm.

Figures 6.3 shows $\sum_{x_j \in U_i} w_j \big((d_\theta \circledast z)(x_j)\big)^2$ as a function of $\theta \in [0, \pi]$ within regions of the blurred image containing isotropic textures or edges. Regions containing edges, as pointed out in [51], can be exploited for estimating the camera rotation: in z only edges tangent to the blurring paths are preserved. Formula (6.5) gives accurate results also when U_i contains a blurred edge, as the direction minimizing the derivatives energy is the edge direction, i.e the blur tangent direction, see Figure 6.4

The directions tangent to the blurring paths, estimated with formula (6.5), are therefore reliable also in regions containing blurred edges.

6.3.2. Voting Procedure for Circular Blurring Paths

When the camera optical axis and the rotation axis a coincide, the blurring paths are circumferences centered in $C = \pi \cap a$, see Figure 6.2.a. Circular blurring paths are obtained also when a is parallel to the optical axis and the scene is planar and parallel to the image plane [77, 39]. In this case C can be determined by a Generalized Hough Transform [1].

The Generalized Hough Transform is a procedure for computing robust solution to a problem, given some input data. The procedure is developed by means of a parameters space P, which is the set of all the possible solutions.

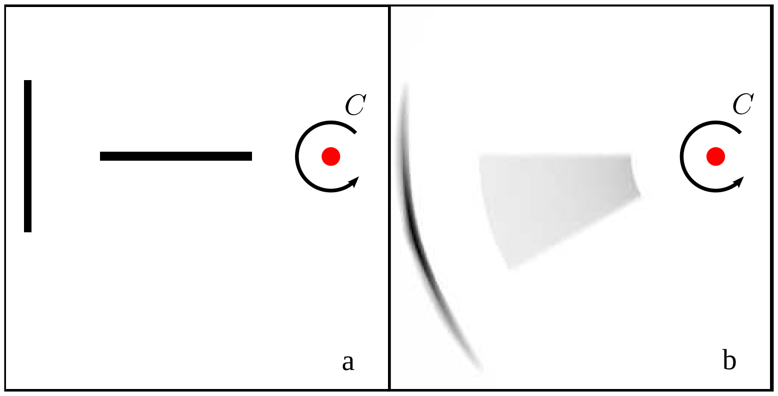


Figure 6.4.: (a) Two straight segments are rotated around C (b) In the blurred image, the part of the segments which are tangent to the blurring paths are preserved, while the others are lost (in those directed along radial lines).

A vote is assigned to every parameter that satisfy a datum and then summed to the votes coming from the other data. After having considered all the data, the parameter which received the highest vote is taken as a solution.

In our case P is a discrete grid of all the possible location for $C \in \pi$ and data are the pairs $(x_i, \theta_i) \quad i = 1, .., N$. Note that C could be outside of the image grid X. Each data (x_i, θ_i) identifies a line l_i, the line tangent to the blurring path at x_i. The set of all the possible rotation centers C, given the line l_i, is the line perpendicular to l_i and passing through x_i.

Consider the root mean square error of each θ_i,

$$\sigma_i = \sqrt{E[(\theta_i - \theta_i^*)^2]} \tag{6.6}$$

where θ_i^* represents the true tangent blur direction at x_i and $E[\bullet]$ the mathematical expectation. Since we cannot directly compute σ_i, we approximate it with an indirect measurement: for example considering the amplitude of the area near θ_i in the energy function minimized in (6.5) or considering σ_i proportional to σ_η (6.1). Noise standard deviation is estimated using [20]. Given a datum (x_i, θ_i), we assign a full vote to all the exact solutions and we spread smaller votes to the neighboring parameters, according to the errors in θ_i.

Let now $p = (p_1, p_2)$ represent a coordinate system in the parameters space and assume $\theta_i = 0$ and $x_i = p_i = (0, 0)$. Let now model the vote spread assuming that along the line $p_1 = 1$ the errors are distributed as $\sigma_i\sqrt{2\pi} \cdot N(0, \sigma_i)$. We model the vote spread so that along line $p_1 = k$, the votes are still Gaussian distributed with a full vote at the exact solution $(k, 0)$ and for

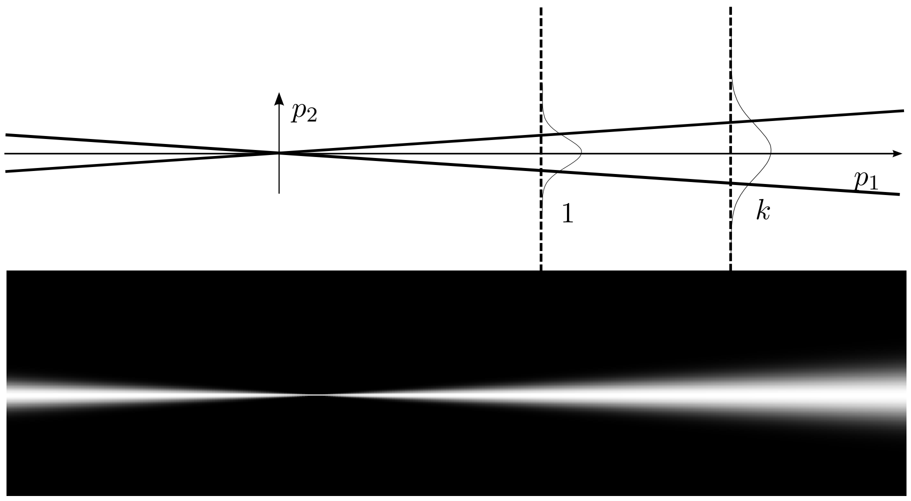


Figure 6.5.: Weight function used for the votes spread.

neighboring parameters the votes depend only on the angular distance from θ_i, see Figure 6.5. Therefore the following weight function is used for distributing the votes in the parameter space (when $x_i = p_i = (0,0)$ and $\theta_i = 0$),

$$v_i(p_1, p_2) = exp\left[-\frac{p_2^2}{1+p_1^2\sigma_i^2}\right] , \tag{6.7}$$

The votes weight function v_i, associated to other data (x_i, θ_i), correspond to Equation (6.7) opportunely rotated and translated. When all pairs (x_i, θ_i) $i = 1, .., N$ have been considered, the parameter that received the highest vote is taken as the solution, i.e.

$$\hat{p} = \arg\max_{p\in P} \mathcal{V}(p) , \quad \text{being} \quad \mathcal{V}(p) = \sum_{i=1}^{N} v_i(p) . \tag{6.8}$$

The coordinates of $C = \pi \cap a$ are determined from $\hat{p}$.

Figure 6.6 illustrates the voting procedure on a synthetically blurred test image.

6.3.3. Conic Section Blurring Paths

Assuming circular blurring paths reduces the complexity load but gives inaccurate solutions whenever a is not perpendicular to π. We present an algorithm for estimating a and ω when $V \in a$ and a is in a general position w.r.t. π. In

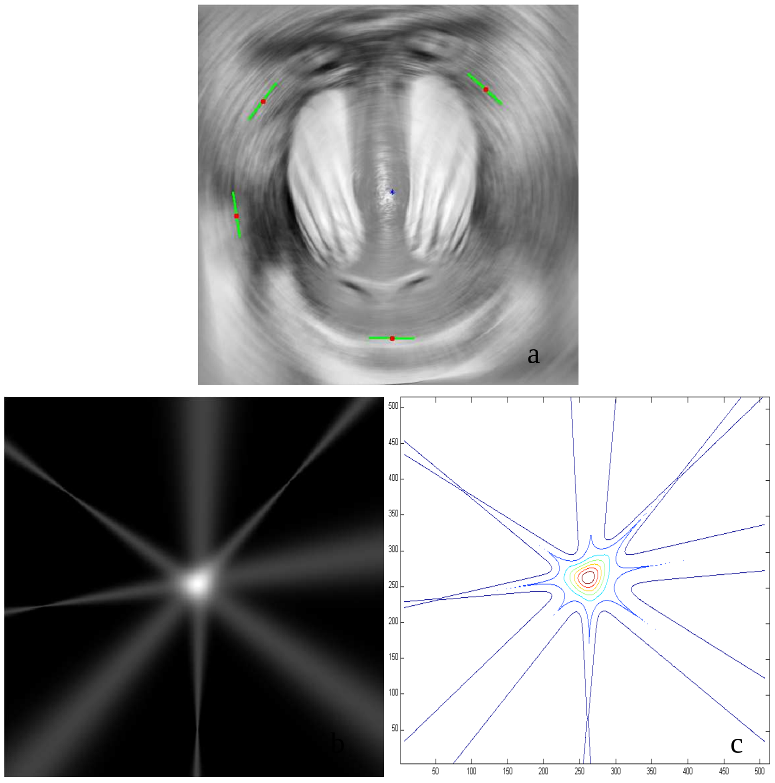


Figure 6.6.: (a) Rotationally blurred image with some blur tangent direction estimates. (b) Votes in the parameter space, (c) votes contours

particular, if we call π_C a plane perpendicular to a, π_C is obtained by two rotations of α and β from π. We do not consider $V \notin a$ as in this case the blur would depend on the scene depth.

Votes in the parameters space illustrates what happens if circular blurring paths are assumed when a is not orthogonal to π. Figure 6.7.a shows a blurred image produced when the plane orthogonal to a forms angles $\alpha^* = 45°$ and $\beta^* = 0°$ with π. If we treat the blurring paths as circumferences, the votes in the parameters space do not point out a clear solution, as shown in Figure 6.7.b and 6.7.c.

Directions θ_i obtained from (6.5) represent the blurring paths tangent direction, even when the blurring paths are conic sections. But the blurring paths themselves are not circumferences, thus lines perpendicular to these tangent lines do not cross at the same point.

From basic 3D geometry considerations, and as pointed out in [51], it follows that the blurring paths are circumferences on an ideal spherical sensor

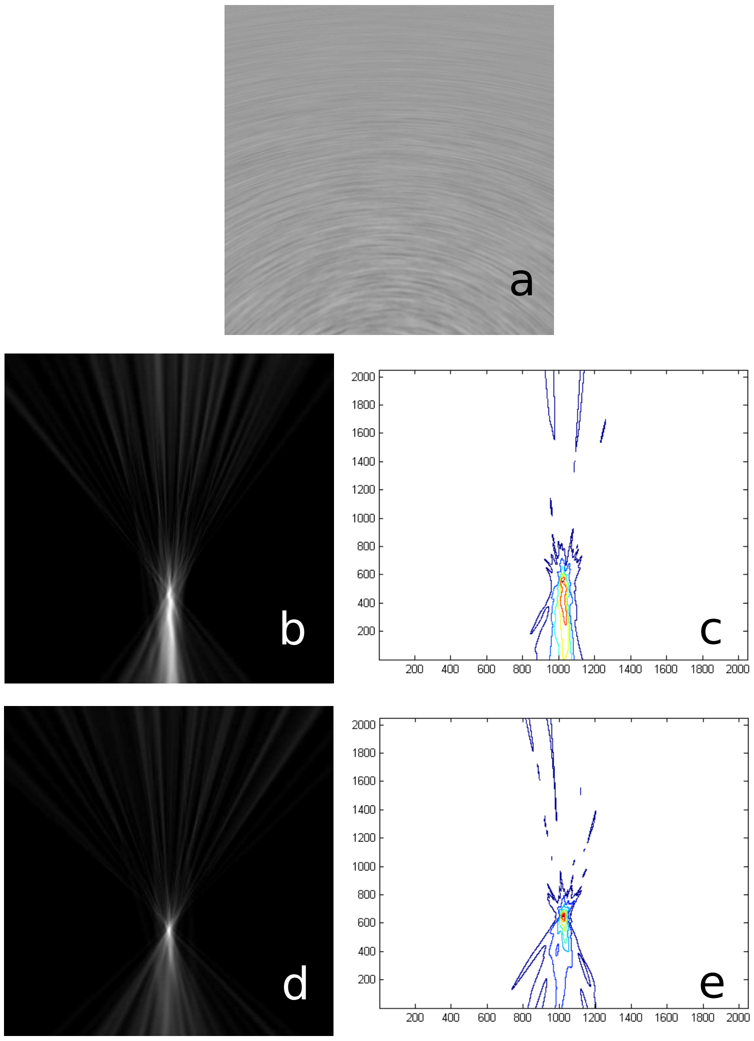


Figure 6.7.: (a) Rotationally blurred image with rotation axis $\alpha^* = 45°, \beta^* = 0°$. (b) Votes assuming circular blurring paths, (c) votes contours. (d) Votes obtained transforming the data with $\hat{\alpha} = 45°, \hat{\beta} = 0°$, (e) votes contours. The maximum vote in (d) is 33% higher than the maximum vote in (b). This is due to the fact that transforming the data with $M_{45,0}$ the blurring paths become circumferences having the same center.

S, Figure 6.2.b. Then, if we project the image from π on S surface, the blurring paths become circumferences. Each of these circumferences belongs to a plane and all these planes have the same normal: the rotation axis a. Let now consider one of these planes, π_C, tangent to the sphere. The projections of the blurring paths on π_C are circumferences, Figure 6.2.b.

The plane π and the plane π_C are related by a projective transformation determined by two parameters, namely (α, β), the angles between the two planes. Define the map $M_{\alpha,\beta} : \pi \mapsto \pi_{\alpha,\beta}$ as the projection from V between π and $\pi_{\alpha,\beta}$, which is the plane tangent to S, forming angles (α, β) with π [78]. We search for (α, β) that project the blurring paths into circumferences, by modifying the voting procedure of Section 6.3.2.

There is no need to transform the whole image with $M_{\alpha,\beta}$ as each l_i, the line tangent to the blurring path at x_i, can be directly mapped via $M_{\alpha,\beta}$. Let $v_i^{\alpha,\beta}$ be the weight function (6.7) associated to data $(x_i, \theta_i) \quad i = 1, .., N$ mapped via $M_{\alpha,\beta}$. The parameters pair identifying the plane π_C is estimated as

$$(\hat{\alpha}, \hat{\beta}) = \arg \max_{\alpha,\beta} \mathcal{V}^{\alpha,\beta}(\hat{p}_{\alpha,\beta}), \tag{6.9}$$

where

$$\hat{p}_{\alpha,\beta} = \arg \max_{p \in P} \mathcal{V}^{\alpha,\beta}(p) \, , \quad \mathcal{V}^{\alpha,\beta}(p) = \sum_{i=1}^{N} v_i^{\alpha,\beta}(p) \, . \tag{6.10}$$

Figure 6.7.d and 6.7.e represent the votes in case the data have been transformed according to the correctly estimated parameters $\hat{\alpha} = 45°, \hat{\beta} = 0°$. These votes are much more concentrated than votes in Figure 6.7.b and 6.7.c.

Once $\hat{\alpha}$ and $\hat{\beta}$ have been estimated, the camera rotation axis a is determined and it is possible to map the image z to $M_{\hat{\alpha},\hat{\beta}}(z)$. As said before, in $M_{\hat{\alpha},\hat{\beta}}(z)$ the blurring paths are circumferences centered at $M_{\hat{\alpha},\hat{\beta}}(C) \equiv \pi_C \cap a$ and it is therefore possible to transform $M_{\hat{\alpha},\hat{\beta}}(z)$ in polar coordinates for estimating the angular speed.

6.3.4. Angular Speed Estimation

Once C has been determined, it is possible to transform $M_{\hat{\alpha},\hat{\beta}}(z)$ (the image projected on π_C) on a polar lattice (ρ, θ) w.r.t to $M_{\hat{\alpha},\hat{\beta}}(C)$ [77]. On the polar lattice, the blur is space invariant with the PSF directed along lines $\rho = const$.

We estimate the PSF extent using the method proposed by Yitzhaky [94] as this can be applied to a restricted image area, avoiding lines which contain several pixels of padding introduced by the polar transformation. The PSF extent, opportunely scaled by the factor due to the polar lattice resolution, divided by the exposure time gives the camera angular speed.

6.4. Implementation Details

We adopted the 3 tap derivative filters presented in [21] for blur analysis in equation (6.5). These filters have been selected as they provide good accuracy and as they are separable. The separability allows fast implementation as only the responses to two basic filters need to be computed. We experienced also that the use of cross derivatives increases the accuracy of the estimated blurring path tangent directions. This is due to the fact that when one derivative component is directed as the blur, the orthogonal component performs whitening on the image, reducing its intrinsic spatial correlation [94]. Moreover using cross derivative filters reduces the range of the considered angles in (6.5) to $[0, \pi/2]$. Then for determining the correct blur tangent direction among θ_i and $\theta_i + \pi/2$ (with $\theta_i \in [0, \pi/2]$ solution of (6.5)), we compute the responses w.r.t. derivative filters along θ_i and $\theta_i + \pi/2$ and we take the one having minimum energy. Moreover, as the plots in Figure 6.3 show that the term $\sum_{x_j \in U_i} w_j \big((d_\theta \circledast z)(x_j)\big)^2$ varies smoothly w.r.t. θ, the minimization could be done in a multi-scale manner, considering first a coarse set of angles and then increasing the resolution in a neighborhood of the minimum. Finally, the window w in (6.5) has Gaussian weights with the maximum in the window center.

In the voting procedure for estimating $(\hat{\alpha}, \hat{\beta})$ we considered two set of angles A and B for α and β respectively. For each pair $(\alpha, \beta) \in A \times B$, we run the voting procedure (6.10) sampling the function $v_i^{\alpha,\beta}$. This makes the algorithm computationally demanding. The voting procedure can be sped up in a multi scale implementation which can be applied to A, B and also to the parameter space P.

However, since the analytic expression of vote spread is known (see Equation (6.7)), it is possible to use any numerical minimization techniques.

6.5. Experiments

The algorithm has been validated both on synthetic and camera images. Synthetic images of Figure 6.8 have been generated with a raytracer software [41] rotating the camera in front of planar tiles of test images. Blurred images are obtained averaging all the rendered frames, according to Equation (6.1). Ten frames (512x512 pixels, grayscale 0-255) are rendered per each rotation degree. The blurring paths tangent directions are estimated in 121 equally spaced regions having a 10 pixel radius, using formula (6.5).

Table 6.1 shows $\mathcal{V}^{\alpha,\beta}(\hat{p}_{\alpha,\beta})$ (the value of the maximum vote obtained with (α, β)) as a percentage w.r.t $\mathcal{V}^{\hat{\alpha},\hat{\beta}}(\hat{p}_{\hat{\alpha},\hat{\beta}})$ (the maximum vote obtained with

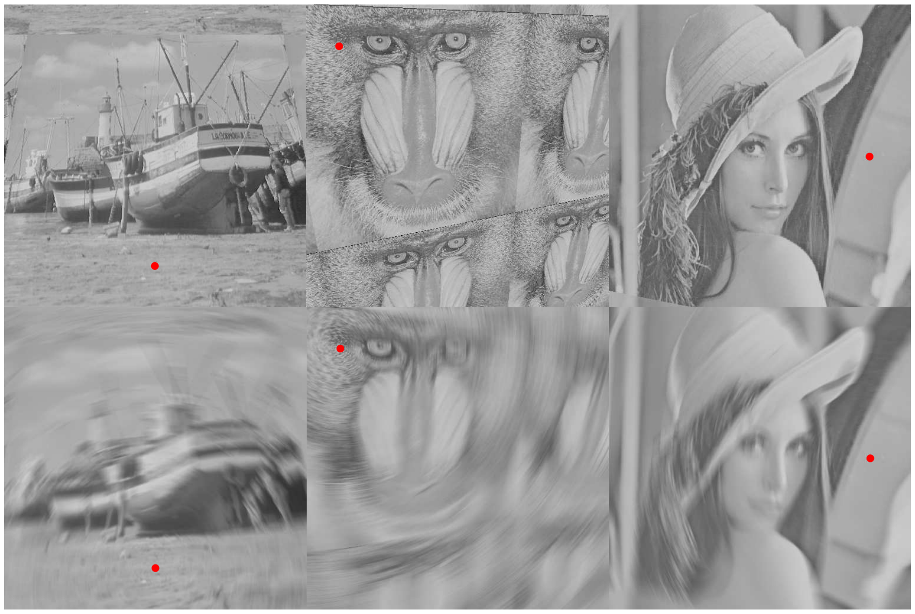


Figure 6.8.: Still and rotationally blurred synthetic images. First row, left to right: Boat ($\alpha^* = 20°, \beta^* = 0°$), Mandrill ($\alpha^* = -20°, \beta^* = 20°$) and Lena ($\alpha^* = 0°, \beta^* = -20°$). Second row: Boat, Mandrill and Lena, rotationally blurred with an angular speed of 6, 8 and 6 deg/s, respectively, assuming 1 second of exposure time. Intersection between image plane and rotation axis is marked with a red circle.

$(\hat{\alpha}, \hat{\beta})$). Here $(\hat{\alpha}, \hat{\beta})$ coincides with (α^*, β^*), the ground truth. Table 6.2 shows the results at a second iteration considering a refinement around $(\hat{\alpha}, \hat{\beta})$).

Table 6.3 shows results obtained on synthetic images of Figure 6.8. Each of them has been tested adding white Gaussian noise with standard deviation 0, 0.5 and 1 and considering α and β in $A = B = \{-40°, -20°, 0°, 20°, 40°\}$. Algorithm performance are evaluated with $\Delta(\alpha)$=$|\hat{\alpha} - \alpha^*|$ and $\Delta(\beta)$=$|\hat{\beta} - \beta^*|$. $\Delta(\hat{C})$ and $\Delta(\hat{\omega})$ represent the absolute error between the ground truth and the estimated values of $C = \pi \cap a$ and ω, respectively.

The effectiveness of our algorithm is evaluated as

$$adv = \frac{\mathcal{V}^{\hat{\alpha},\hat{\beta}}(\hat{p}_{\hat{\alpha},\hat{\beta}}) - \mathcal{V}^{\alpha_2,\beta_2}(\hat{p}_{\alpha_2,\beta_2})}{\mathcal{V}^{\alpha_2,\beta_2}(\hat{p}_{\alpha_2,\beta_2})}\,, \tag{6.11}$$

where $\mathcal{V}^{\alpha_2,\beta_2}(\hat{p}_{\alpha_2,\beta_2})$ represents the maximum vote obtained among other parameters (α, β). The higher this ratio, the better. Finally, $\Delta(C^{0,0})$ and $\Delta(\omega^{0,0})$

α β	-40	-20	0	20	40
-40	33	52	72	53	48
-20	39	52	83	63	44
0	34	57	83	63	43
20	40	55	100	55	44
40	35	42	62	39	34

Table 6.1.: Boat. Highest votes corresponding to (α, β) in the parameters space, expressed as a percentage with respect to the maximum vote.

α β	-10	0	10
10	71	79	80
20	63	100	74
30	57	82	61

Table 6.2.: Refinement around $(\hat{\alpha}, \hat{\beta})$ from Table 6.1.

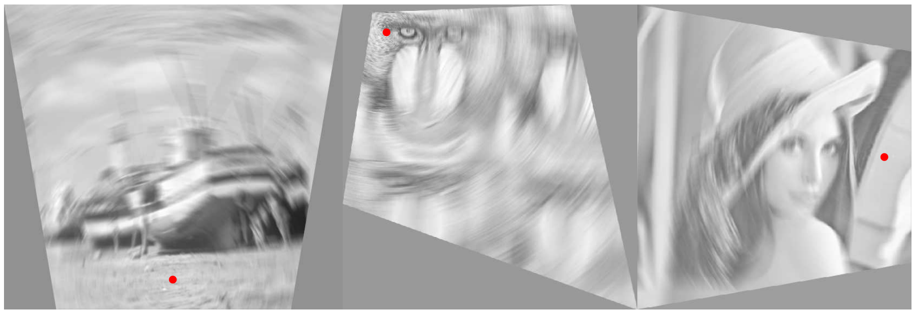


Figure 6.9.: Boat, Mandrill and Lena rectified with the corresponding estimated $(\hat{\alpha}, \hat{\beta})$. Intersection between the image plane and the rotation axis is marked with a red circle.

are the corresponding errors obtained assuming circular blurring paths. Results for noisy images represent the average over ten different noise realizations.

Results reported in Table 6.3 show that our algorithm can cope with a reasonable amount of noise, obtaining regularly better results than the circular blur assumption. This is more evident in the estimation of the angular speed, which lacks physical meaning when the rotation axis is not correctly identified. Figure 6.9 shows blurred images of Figure 6.8 transformed with the corresponding $M_{\hat{\alpha},\hat{\beta}}$.

Camera images have been captured rotating a Canon EOS 400D camera on a tripod, assuring that a is orthogonal to the floor. The ground truth α^* and β^*, can be then computed rectifying still images of a checkerboard on the floor.

Image	σ_η	$\Delta(\alpha)$	$\Delta(\beta)$	$\Delta(\hat{C})$	$\Delta(\hat{\omega})$	*adv*(%)	$\Delta(C^{0,0})$	$\Delta(\omega^{0,0})$
Boat	0	0	0	2.20	0.23	20.44	33.06	4.83
Boat	0.5	0	0	5.46	0.24	20.23	21.27	114.55
Boat	1	0	0	8.84	0.19	8.84	19.25	71.98
Mandrill	0	0	0	1.00	0.09	5.66	7.07	0.96
Mandrill	0.5	2	2	1.48	0.11	6.13	4.81	2.85
Mandrill	1	4	4	1.17	0.26	5.25	4.41	2.29
Lena	0	0	0	3.00	0.08	11.01	12.08	0.60
Lena	0.5	0	0	3.88	0.20	14.06	33.64	64.94
Lena	1	0	4	5.23	0.48	6.00	29.43	62.58

Table 6.3.: Algorithm performance on synthetic images. When $\sigma_\eta > 0$, averages over 10 noise realizations.

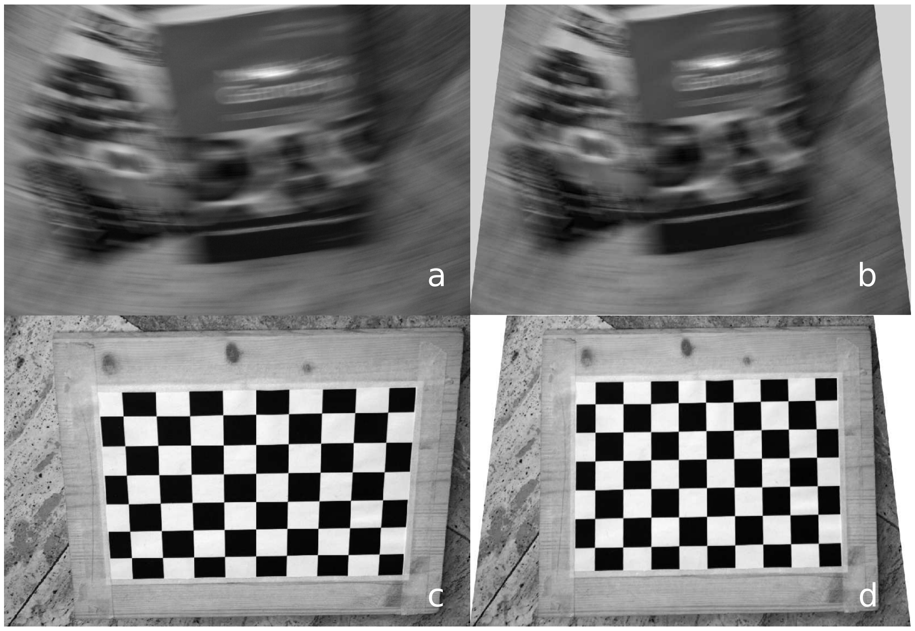


Figure 6.10.: Blurred camera image. (a) Blurred image ($\alpha^* = -27°$, $\beta^* = 0°$), (b) rectified image with estimated $\hat{\alpha} = -30°$, $\hat{\beta} = 0°$, (c) checkerboard with the same camera inclination, (d) checkerboard rectified with $\hat{\alpha} = -30°$, $\hat{\beta} = 0°$.

Figures 6.10.a and 6.12.a show the downsampled RAW converted in grayscale used to test our algorithm. The blurring path tangent directions are estimated on 187 uniformly spaced regions, having 10 pixel radius.

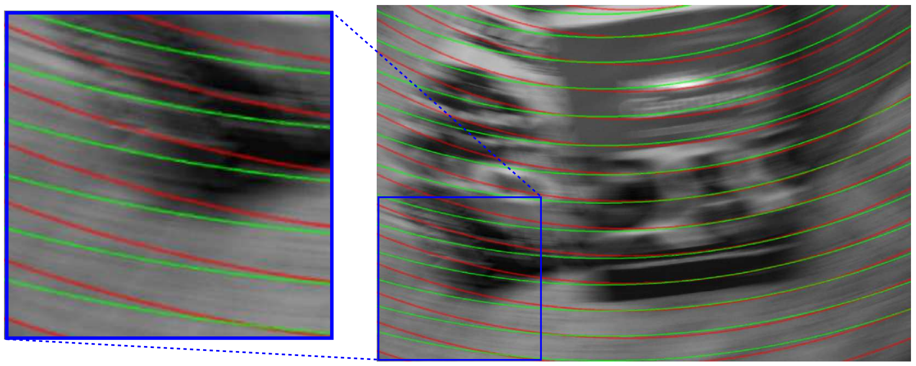


Figure 6.11.: Comparison between circular (red) and conic section (green) blurring paths on 6.10.a. Green blurring paths describe more accurately the image blur.

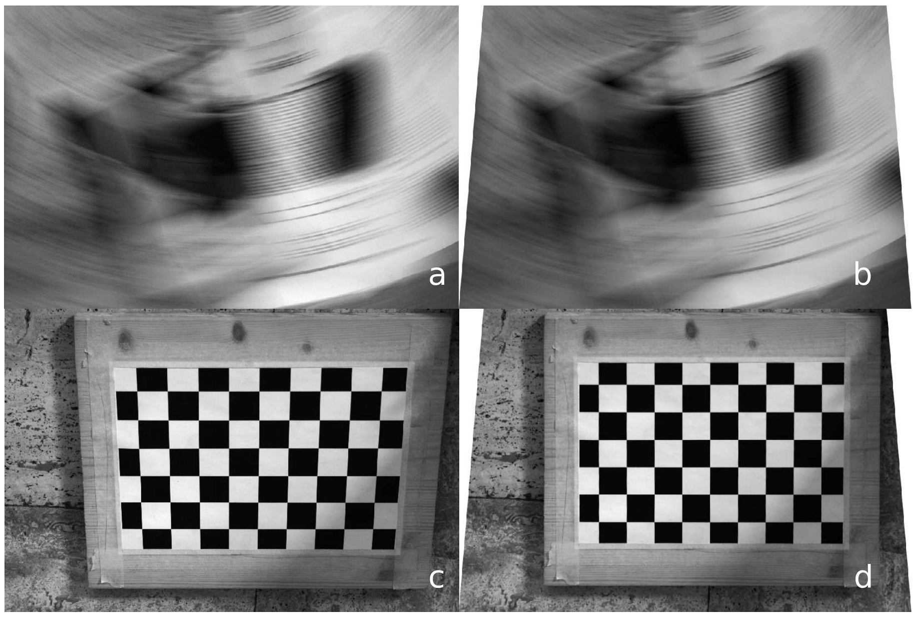


Figure 6.12.: Blurred camera image. (a) Blurred image ($\alpha^* = -20°$, $\beta^* = 0°$), (b) rectified image with estimated $\hat{\alpha} = -20°$, $\hat{\beta} = 0°$, (c) checkerboard with the same camera inclination, (d) checkerboard rectified with $\hat{\alpha} = -20°$, $\hat{\beta} = 0°$.

Tables 6.4 and 6.5 show the results of the execution of two iterations of the algorithm on Figure 6.10.a. The solution obtained is $\hat{\alpha} = -30°$ and $\hat{\beta} = 0°$, which is acceptable as the ground truth, obtained from the checkerboard, is

α β	-40	-20	0	20	40
-40	63	88	100	88	52
-20	62	77	78	70	54
0	67	74	80	70	62
20	62	81	85	65	63
40	71	73	82	78	77

Table 6.4.: Camera Image. Highest votes corresponding to (α, β) in the parameters space, expressed as a percentage with respect to the maximum vote.

α β	-10	0	10
-50	82	76	80
-40	81	96	72
-30	85	100	83

Table 6.5.: Camera Image. Refinement around $(\hat{\alpha}, \hat{\beta})$ from Table 6.4.

$(-27°, 0°)$. Figure 6.11 points out the differences between the blurring paths estimated with the circular approximation (in red) and the conic section paths estimated by our method (in green). As clearly seen from the detail, the blur is correctly interpeted by the green blurring paths. Figure 6.12 shows results on another camera image, having $\alpha^* = -20°$ and $\beta^* = 0°$. After two iterations, the algorithm converges exactly to the correct solution. Figures 6.10 and 6.12 show the blurred images and the checkerboard images rectified with the estimated $(\hat{\alpha}, \hat{\beta})$.

6.6. Conclusions

We described a novel algorithm for estimating the camera rotation axis and the angular speed from a single blurred image. The algorithm provides accurate estimates also in the most challenging cases, when the rotation axis is not orthogonal to the image plane. To the best of the authors' knowledge, none of the existing methods handles these cases correctly since known methods assume circular blurring paths. We have shown how this assumption produces inaccurate estimates when the rotation axis is not orthogonal to the image plane, while our algorithm is more accurate.

The algorithm is aiming to high accuracy rather than efficiency. Accuracy in the estimation of these parameters is a primary issue in restoring such images as the deblurring is typically based on a coordinate transformation and a deconvolution, which are highly sensitive to errors.

Ongoing works concern the design of a more noise-robust method for blur analysis on image regions and the implementation of a faster voting procedure.

7. Future Work

In this thesis we addressed the issues of analyzing and restoring motion blurred images, i.e. images where the blur is due to any camera motion. In particular the focus was on the blur due to camera translation and the camera rotation. We devised two innovative algorithms form estimating both the blur parameters and the camera motion when a single image, blurred because of translation or rotation is available. Moreover a restoration algorithm for radial blurred images that takes into account also the image noise has been proposed. We also devised algorithms that exploit the image content in order to estimate the PSF parameters within corner regions.

The blurred corner detection and the adaptive region selection procedures presented in Chapter 2 have to be further investigated. Both an improved formulation and more exhaustive experimental validation will be done in the next furure.

Ongoing works concern blind deblurring algorithms for rotational and translational blur. We are investigating procedures for estimating the blur extent parameter in radial blurred images, so that the restoration algorithm presented in Chapter 4 can be used in cascade with the epipolw estimation algorithm of Chapter 3. We also are planning to model the statistics after the rotational blur inversion, following the procedure illustrated in Section 4.4.2, in order to restore images corrupted by both rotational blur and noise. Also in this case the crucial issue is the accuracy in the coordinate transformation that maps the blurred image on a plane orthogonal to the rotation axis, as this is required when inverting the blur.

On the contrary, we are not planning to work on the depth estimation from a single motion blurred image, presented in Chapter 5. In case of a more generic camera motions depth estimation from a single blurred image may be not feasible because of the orientation problem. Finally, the accuracy provided by the blur estimates, seems not satisfying for depth reconstruction. It will be rather investigated how to exploit blurred frames in video sequences and how to perform blurred target detection and localization in images and videos.

Techniques For Estimating Standard Deviation of Additive White Gaussian Noise from a Single Image

Additive White Gaussian Noise (AWGN) is commonly used in image formation models to consider electronic and thermal noise, quantization errors and most of signal independent random effects. In this thesis we often considered AWGN (Chapters 3, 4 and in experiments of Chapters 1 and 2) as this noise term is commonly used to approximate the sum of the noise from different sources and the quantization errors (at least in correctly exposed blurred images).

Noise standard deviation estimation is a preliminary step in several image restoration (e.g. denoising and deblurring) and image analysis algorithms (e.g. background subtraction, tracking). This has been used extensively also in this thesis, for example in Chapters 1 and 2 for removing pixels having small gradient norm, in Chapter 4 it is used in the Shape Adaptive DCT denoising algorithm. In Chapters 1, 3 and 6 the noise standard deviation has been used to tune the amplitude of vote spread.

In this appendix we present an overview of existing techniques for estimating AWGN standard deviation from a single image and we focus on Median of Absolute Deviation (MAD) based estimators.

State of the art in AWGN standard deviation estimation in images

We consider the following observation model

$$z(x) = y(x) + \eta(x), \quad \eta(x) \sim N(0, \sigma^2) \quad x \in \mathcal{X} \tag{7.1}$$

where y is the (unknown) true image value, η represents the Additive White Gaussian Noise (AWGN) that corrupts the observed image y and x is a vector representing pixel coordinates on image domain $\mathcal{X}$.

We give an overview of methods for estimating the standard deviation of a stochastic process η, whose values $\eta(x)$ are independent and identically distributed realizations a Gaussian random variable for any $x \in \mathcal{X}$.

Algorithms that perform this task mainly follow two approaches: the filtering approach and the block-based approach. A good survey and performance comparison between some of these methods has been presented by Olsen [71].

The filtering approach exploit the separation of noise from true image, which is generally obtained subtracting from the (noisy) observation z a smoothed observation z_s obtained by filtering z. This can be done using both linear and non linear filtering such as averaging filters or block-wise median [71]. More sophisticated algorithms following this approach have been later introduced: Rank *et al.* [73] for example propose an algorithm based on Differentiating Filters. Immerkaer [42] introduced a Laplacian mask filtering on the noisy image that allows fast noise variance estimation. The same Laplacian filtering followed by an Edge detector has been suggested in [16].

The filtering approach includes transform based algorithms [97, 20]. The well known Donoho and Johnstone algorithm [20] is based on Wavelet decomposition and exploit the MAD estimator. This is also filtering based, as Wavelets decomposition [60], [61] exploits linear filtering and down-sampling. The wavelet detail coefficients are considered noise, and this idea motivates the Wavelet Shrinkage method. Therefore the noise standard deviation can be computed using a robust estimator, the Median of Absolute Deviations (MAD), directly on the first order Wavelet detail coefficients. The performances of this algorithm varies according to the number of vanishing moments of wavelet filters. Typically Daubechies wavelets [18] with three vanishing moments are used.

In the block-based approach the noise standard deviation initially estimated locally on each component of an image tessellation. Then, a statistical procedure selects the most reliable value between possible standard deviation estimates [71], [63].

This basic idea has motivated several algorithms that combine filtering techniques in order to separate the image noise, with some statistical analysis of the standard deviation estimates on image blocks. In particular [73] performs AWGN standard deviation estimation on the whole image by computing the histogram of blocks (or local) noise standard deviation estimates and correcting it according to some prior. A mixed approach has been recently suggested by Shin *et al.* [81], the idea here is to perform a Gaussian smoothing filter in low-variance areas, which have been selected before with a block-wise analysis.

An algorithm based on a completely different approach is described in [91].

It exploits the autocorrelation function of the noisy image obtained considering 1D shift. This histogram presents a maximum at zero which corresponds to the case when image is point-wise multiplied with its non shifted version. Since noise is an i.i.d. process we have

$$E[\eta(x_a) \cdot \eta(x_b)] = 0 \quad x_a \neq x_b, \tag{7.2}$$

while

$$E[\eta(x_a) \cdot \eta(x_b)] = \sigma^2 \quad x_a = x_b. \tag{7.3}$$

Therefore the autocorrelation values are not influenced by the noise but in the origin of the autocorrelation function. A spline or a Gaussian function is fitted in a neighbor of the origin (excluding the origin itself) to the autocorrelation values, estimating thus the autocorrelation of the noise-free image (7.2). The difference between the fitted autocorrelation function and the values of the autocorrelation function in the origin represent the anomalous peck which corresponds to the noise variance (7.3).

Median of Absolute Deviation for AGWN standard deviation estimation

The Median of Absolute Deviation (MAD) [34] is a robust statistic for estimating the standard deviation of Gaussian samples. It can then be used for estimating the standard deviation of AWGN whereas the noise has been filtered from the image. It is used in the wavelet based algorithm [20], which is one of the most performing algorithms for AWGN standard deviation estimation.

Let us explain why robust estimators are used in conjunction with filtering based approaches. In these methods the separation of the image from the noise is never performed exactly, and the filtered observation z_s does not correspond to the original image y. Thus, $z - z_s$ does not represent the pure noise. Therefore the estimation of the standard deviation from $z - z_s$ has to handle values $z - z_s$ which are not Gaussian distributed like η. Differentiating filters typically produce outputs of small magnitude on slow varying signals, with large values in correspondence with rapid changes. Such large values represents outliers in noise estimate $\widetilde{\eta}_0$, and thus a robust estimator for the standard deviation is preferred to the square root of the sample variance.

Let us introduce the MAD, in the context of a trivial filtering based algorithm for the AWGN standard deviation estimation. The noise separation step consist of processing the image with differentiating filter, like

$$\widetilde{\eta} = z \circledast [1, -1] = z(x_1, x_2) - z(x_1, x_2 + 1), \tag{7.4}$$

which consists in subtracting a shifted version of the noisy observation to the observation itself. This is a basic high pass filter, therefore $\widetilde{\eta}$ contains the image high frequency, influenced mostly by the noise.

The noise standard deviation is then estimated using the MAD estimator on $\widetilde{\eta}$. It is defined for Gaussian samples η as

$$\widehat{\sigma} = mad\,(\eta) = \frac{median(|\eta - median(\eta)|)}{0.6745}\,. \tag{7.5}$$

Noise standard deviation can be estimated by using the MAD on $\tilde{\eta}$ and possibly averaging the results obtained with $\tilde{\eta}$ obtained from horizontal and vertical filters.

Wavelet based method [20] corresponds to using MAD on $\widetilde{\eta}$, whereas the filter $[1, -1]$ in equation (7.4) is replaced by Daubechies Wavelet filters. Similar noise estimators has been suggested also in [73] but used in cascade, this situation corresponds to Donoho' algorithm where filters used are those of Haar Wavelet decomposition.

Characterization of Radial Blur

In this Appendix we derive a characterization of the radial blur which have been extensively used in Chapters 3 and 4.

In particular we assume that the blur is a linear process w.r.t to the original image y, i.e. that the blurred observation z can be written as in Formula (4.2) :

$$z(x) = \int_{\mathcal{X}} k(x,s)y(s)ds\,, \quad x \in \mathcal{X}. \tag{7.6}$$

We further assume that the blurring process preserves the ℓ^1 norm of the signal i.e. $||z||_1 = ||y||_1$.

The goal of this Appendix is to determine the PSF at x, i.e $k(x,\bullet)$, in case of camera translation toward planar scene.

From the epipolar constraints and the remarks of Section 3.2.2, it follows that the support of the PSF at x_i, i.e. supp$(k(x_i,\bullet))$ is a straight line segment, having direction

$$\theta_i = \arctan \frac{x_{i,2} - e_2}{x_{i,1} - e_1} \tag{7.7}$$

where $x_{i,1}$ and $x_{i,2}$ are the horizontal and vertical coordinates on the image grid, respectively.

Now we will prove the following Proposition,

Proposition 1 (Uniform blur) *When the captured scene is planar and parallel to the image plane, and when the camera is purely translating, in continuous image domain, each PSF has constant value on its support, at any image pixel.*

Proof. From Equation (7.7), it follows that in order to derive the PSF at x, it is enough to consider the pixels on the straight line on the image plane passing through e and x. Moreover, for simplicity we prove the equivalent case where the camera is static and the planar scene is translating toward the camera: see Figure 7.1. The initial and at the final position of the scene are indicated by S^i and S^f, while the image plane is indicated by π. The scene moves along the straight line through V and e the camera viewpoint and the epipole, respectively.

It follows, that the scene points that intersect the viewing ray at x during the the scene translation lies on the segment $\overline{X'X''}$. This segment is obtained by back-projecting on S^i along the camera translation direction, the point X which represents the intersection between the viewing ray and the scene plane S^f.

However, the blur in Equation (7.6) has to be expressed as a function of the original image y, not as a function of the scene. Therefore we intersect the viewing ray associated to X'' with π and we obtain that $z(x)$ is given by the integral over $\overline{xx''}$.

Since the displacement has been covered at uniform speed, every point of $\overline{xx''}$ is taken into account with the same weight. It follows that in Equation $k(x, \bullet) = const$.

Moreover, since the blur must preserve the ℓ^1 norm of the signal, we have that

$$\int_{\mathcal{X}} k(x,s)\, ds \; = 1 \text{ and} k(x, \bullet) = const\, .$$

■

Note that Proposition 1 holds also in case $e \rightarrow \infty$, and in this case we have standard uniform motion blur case, like those assumed in [13, 67, 68, 94, 95].

Note that on the discrete image domain, Proposition 1 may not hold because near X' wider areas of the scene are imaged into a pixel than near X''. This fact has been neglected in our discrete formulation of Chapter 4. Another difference between discrete domain PSF and the continuous domain PSF here formulated concern the PSF values. While in continuous domain Proposition 1, assures $PSF(x) = const \forall x$, in discrete domain this typically does not hold because of subpixel interpolation. Therefore the PSF is still identified by its direction and extent but it is not constant.

We will now prove that the blur extent, i.e. the size of the PSF support, at each pixel a is proportional to the distance $\overline{ae}$.

Proposition 2 (Blur Extent) *Let a be an image pixel, then under the assumptions of Proposition 1, it follows that l, the blur extent at x is proportional to the distance $\overline{ae}$.*

Proof. Let a and b be two pixels, and let us start proving that, referring to Figure 7.2,

$$\overline{bb'} = \lambda \overline{eb} \Rightarrow \overline{aa'} = \lambda \overline{ea} \tag{7.8}$$

Note that thanks to Talete Theorem from

$$\overline{BB'} = \lambda \overline{EB} \Rightarrow \overline{aa'} = \lambda \overline{ea} \tag{7.9}$$

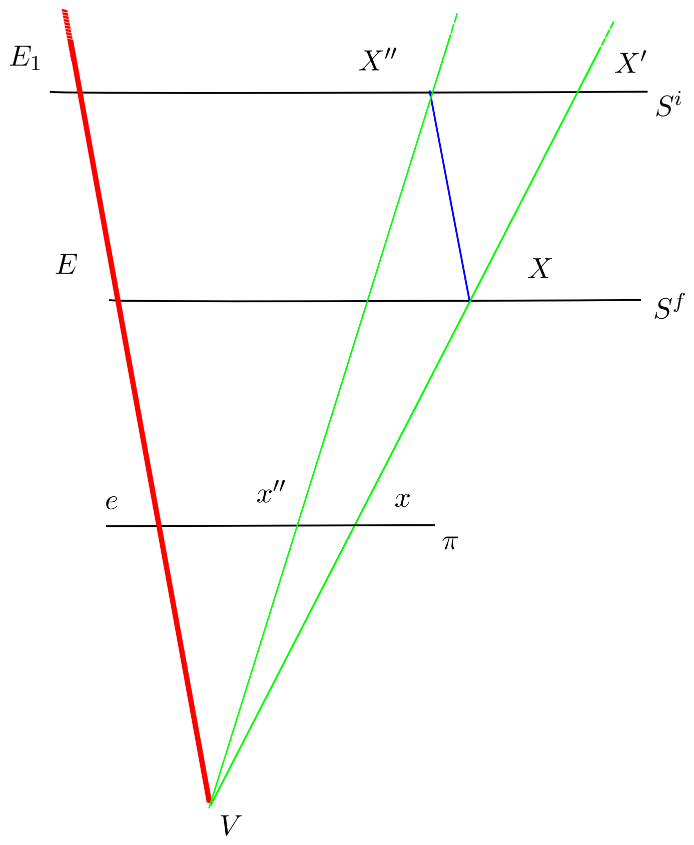


Figure 7.1.: Blurred image formation in the purely translating camera. In green the viewing rays, the red line represents the camera displacement direction. The blue segment identifies the scene displacement and while the scene points $\overline{X'X''}$ are imaged into x.

follows relation (7.8).

In order to prove relation (7.9) we define

$$\mu = \frac{\overline{B'B_1}}{\overline{E_1B_1}}.$$

Since the triangle $\widehat{AA'A_1}$ is similar to the triangle $\widehat{VE_1A_1}$, the following equalities hold

$$\nu = \frac{\overline{E_1A_1}}{\overline{A_1A'}} = \frac{\overline{E_1V'}}{\overline{AA'}} = \frac{\overline{E_1V'}}{\overline{BB'}} = \frac{\overline{E_1B_1}}{\overline{B_1B'}} = \mu\,.$$

Thus relation (7.9) has been proved and relation (7.8) follows.

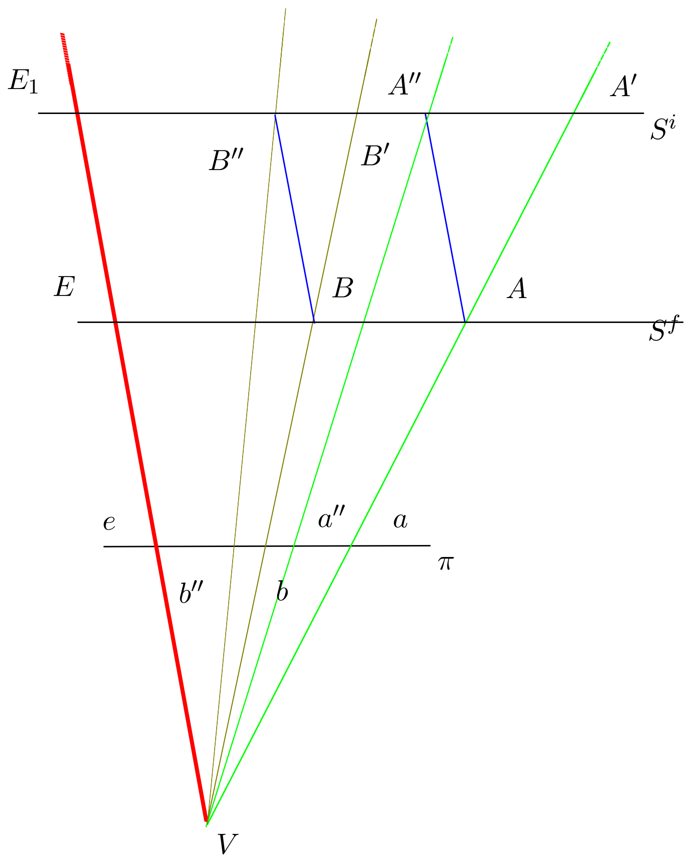


Figure 7.2.: Blurred image formation in the purely translating camera. In green the viewing rays, red represent the camera displacement direction. The blue segment identifies the scene pixels that are blurred in x_i.

We therefore proved that l, the blur extent at a pixel x is proportional to its distance from the epipole $\overline{xe}$, i.e. $l = \lambda\overline{xe}$.

However this does not conclude our proof as there is still to show how that the same λ coefficient holds for any radial line. This trivially follows from the fact that image plane and scene are parallel. $\blacksquare$

The following corollary concludes the proofs for deriving the radial blur model used in Chapter 4

List of Figures

List of Tables

Bibliography

[1] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. pages 714–725, 1987.

[2] B. Bascle, Andrew Blake, and Andrew Zisserman. Motion deblurring and super-resolution from an image sequence. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 573–582, 1996.

[3] M. Ben-Ezra and S.K. Nayar. Motion based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):689–698, Jun 2004.

[4] M. Bertero and P. Boccacci. *Introduction to Inverse Problems in Imaging*. Institute of Physics Publishing, 1998.

[5] Giacomo Boracchi and Vincenzo Caglioti. Corner displacement from motion blur. In *Inproceedings of ICIAP 2007,Modena*, 2007.

[6] Giacomo Boracchi and Vincenzo Caglioti. Motion blur estimation at corners. In *Inproceedings of VISAPP 2007 Conference, Barcelona*, pages 296–302, 2007.

[7] Giacomo Boracchi, Vincenzo Caglioti, and Alberto Danese. Estimating camera rotation parameters from a blurred image. In *Inproceedings of VISAPP 2008 Conference, Funchal Madeira*, 2008.

[8] Giacomo Boracchi, Vincenzo Caglioti, and Alessandro Giusti. Ball position and motion reconstruction from blur in a single perspective image. In *Inproceedings of ICIAP 2007,Modena*, 2007.

[9] Giacomo Boracchi, Vincenzo Caglioti, and Alessandro Giusti. Single-image 3d reconstruction of ball velocity and spin from motion blur. In *Inproceedings of VISAPP 2008 Conference, Funchal Madeira*, 2008.

[10] Giacomo Boracchi, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Deblurring noisy radial-blurred images: spatially-adaptive filtering approach. In *Electronic Imaging, Science and Technology, 2731 January 2008, San Jose, California, USA*, 2008.

[11] T.M. Cannon. Blind deconvolution of spatially invariant image blurs with phase. *ASSP*, 24(1):58–63, February 1975.

[12] T.F. Chan and C.K. Wong. Total variation blind deconvolution. *IEEE Transaction on Image Processing*, 7(3):370–375, March 1998.

[13] A.M.; Erdem A.T. Chang, M.M.; Tekalp. Blur identification using the bispectrum. *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, 39(10):2323–2325, Oct 1991.

[14] Denis Chekhlov, Mark Pupilli, Walterio Mayol-Cuevas, and Andrew Calway. Real-time and robust monocular slam using predictive multi-resolution descriptors. In *2nd International Symposium on Visual Computing*, November 2006.

[15] Ji Woong Choi, Moon Gi Kang, and Kyu Tae Park. An algorithm to extract camera-shaking degree and noise variance in the peak-trace domain. *Consumer Electronics, IEEE Transactions on*, 44(3):1159–1168, Aug. 1998.

[16] B. R Corner, R. M.Narayanan, and S.E. Reichenbach. Noise estimation in remote sensing imagery using data masking. *Int. J. Remote Sensing*, 24- 4:689 –702, 2003.

[17] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Inverse halftoning by pointwise shape-adaptive DCT regularized deconvolution. In *Proc. Int. TICSP Workshop Spectral Methods Multirate Signal Process.*, Florence, Italy, September 2006.

[18] Ingrid Daubechies. *Ten Lectures on Wavelets (C B M S - N S F Regional Conference Series in Applied Mathematics)*. Soc for Industrial & Applied Math, December 1992.

[19] A. Davison. Real-time simultaneous localisation and mapping with a single camera, 2003.

[20] David L. Donoho and Iain M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

[21] H. Farid and E.P. Simoncelli. Differentiation of discrete multi-dimensional signals. *IEEE Transactions on Image Processing*, 13(4):496–508, 2004.

[22] P. Favaro. Shape from focus and defocus: Convexity, quasiconvexity and defocus-invariant textures. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–7, 2007.

[23] Paolo Favaro and Stefano Soatto. *3-D Shape Estimation and Image Restoration: Exploiting Defocus and Motion-Blur*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[24] Quansheng Liu Feng Xue and Jacques Froment1. An a contrario approach for parameters estimation of a motion-blurred image. In Springer Berlin / Heidelberg, editor, *Lecture Notes in Computer Science, Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2007.

[25] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T. Roweis, and William T. Freeman. Removing camera shake from a single photograph. *ACM Trans. Graph.*, 25(3):787–794, 2006.

[26] Javier Portilla Filip Rooms, Wilfried Philips. Parametric psf estimation via sparseness maximization in the wavelet domain. *Wavelet Application in Industrial Processing*, II, 2004.

[27] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[28] A. Foi and K. Dabov. Pointwise shape-adaptive dct demobox. http://www.cs.tut.fi/˜foi/SA-DCT.

[29] A. Foi, K. Dabov, V. Katkovnik, and K. Egiazarian. Shape-adaptive DCT for denoising and image reconstruction. In *Proc. SPIE Electronic Imaging: Algorithms and Systems V*, volume 6064A-18, San Jose, CA, USA, January 2006.

[30] A. Foi, V. Katkovnik, and K. Egiazarian. Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images. *IEEE Trans. Image Process.*, 16(5), May 2007.

[31] Alessandro Giusti and Vincenzo Caglioti. Isolating motion and color in a motion blurred image. In *Proc. of BMVC 2007*, 2007.

[32] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[33] J. A. Guerrero-Colon, L. Mancera, and J. Portilla. Image restoration using space-variant gaussian scale mixtures in overcomplete pyramids. *IEEE Transactions on Image Processing*, 17(1):27–41, January 2008.

[34] F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, and W.A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. Wiley, 1986.

[35] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference,*, pages 147–151, 1988.

[36] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[37] Samuel W. Hasinoff and Kiriakos N. Kutulakos. Confocal stereo. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 620–634, 2006.

[38] Ellen Catherine Hildreth. *Measurement of Visual Motion*. MIT Press, Cambridge, MA, USA, 1984.

[39] H. Hong and T. Zhang. Fast restoration approach for rotational motion blurred image based on deconvolution along the blurring paths. *Optical Engineering*, 42:347–3486, December 2003.

[40] http://www.f lohmueller.de.

[41] http://www.povray.org/.

[42] John Immerkær. Fast noise variance estimation. *Computer Vision and Image Understanding*, 64(2):300–302, 1996.

[43] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[44] Jiaya Jia. Single image motion deblurring using transparency. In *Inproceedings of CVPR 2007, Minneapolis*, 2007.

[45] V. Katkovnik. A spatially adaptive nonparametric regression image deblurring. *IEEE Trans. Signal Process.*, 47(9):2567–2571, September 1999.

[46] V. Katkovnik, K. Egiazarian, and J. Astola. A spatially adaptive nonparametric regression image deblurring. *IEEE Trans. Image Process.*, 14(10):1469–1478, October 2005.

[47] Vladimir Katkovnik, Karen Egiazarian, and Jaakko Astola. *Local Approximation Techniques in Signal and Image Process.*, volume PM157. SPIE Press, 2006.

[48] K. Egiazarian Katkovnik V., D. Paliy and J. Astola. Frequency domain blind deconvolution in multiframe imaging using anisotropic spatially-adaptive denoising. *Proc. 14th European Signal Process. Conf, EUSIPCO 2006, Florence*, 2006.

[49] S. Kawamura, K. Kondo, Y. Konishi, and H. Ishigaki. Estimation of motion using motion blur for tracking vision system. In *World Automation Congress, 2002. Proceedings of the 5th Biannual*, volume 13, pages 371–376, 9-13 June 2002.

[50] I. Klapp and Y. Yitzhaky. Angular motion point spread function model considering aberrations and defocus effects. *Journal of the Optical Society of America A*, 23:1856–1864, August 2006.

[51] Georg Klein and Tom Drummond. A single-frame visual gyroscope. In *Proc. British Machine Vision Conference (BMVC'05)*, volume 2, pages 529–538, Oxford, September 2005. BMVA.

[52] D. Kundur and D. Hatzinakos. Blind image deconvolution. *SPMag*, 13(3):43–64, May 1996.

[53] A. Levin, R. Fergus, F. Durand, and W.T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics, SIGGRAPH 2007 Conference Proceedings, San Diego, CA*, 2007.

[54] Anat Levin. Blind motion deblurring using image statistics. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

[55] Anat Levin, Dani Lischinski, and Yair Weiss. A closed form solution to natural image matting. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 61–68, Washington, DC, USA, 2006. IEEE Computer Society.

[56] Huei-Yung Lin. Vehicle speed detection and identification from a single motion blurred image. *wacv-motion*, 01:461–467, 2005.

[57] Huei-Yung Lin and Chia-Hong Chang. Automatic speed measurements of spherical objects using an off-the-shelf digital camera. In *Mechatronics, 2005. ICM '05. IEEE International Conference on*, pages 66–71, 10-12 July 2005.

[58] Huei-Yung Lin and Chia-Hong Chang. Depth recovery from motion blurred images. *icpr*, 1:135–138, 2006.

[59] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[60] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(7):674–693, 1989.

[61] Stéphane Mallat. *A Wavelet Tour of Signal Processing, Second Edition (Wavelet Analysis & Its Applications)*. Academic Press, September 1999.

[62] Markku Vehvilainen Marius Tico. Estimation of motion blur point spread function from differently exposed image frames. In *Proceedings of Eusipco 2006, 4-6 September 2006 Florence, Italy*, 2006.

[63] P. Meer, J.M. Jolion, and A. Rosenfeld. A fast parallel algorithm for blind estimation of noise variance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):216–223, 1990.

[64] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[65] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1/2):43–72, 2005.

[66] Y. Mishima. Soft edge chroma-key generation based upon hexoctahedral color space. U.S. Patent 5,355,174, 1993.

[67] Mohsen Ebrahimi Moghaddam and Mansour Jamzad. Linear motion blur parameter estimation in noisy images using fuzzy sets and power spectrum. *EURASIP Journal on Advances in Signal Processing*, 2007:Article ID 68985, 8 pages, 2007. doi:10.1155/2007/68985.

[68] Mohsen Ebrahimi Moghaddam and Mansour Jamzad. Motion blur identification in noisy images using mathematical models and statistical measures. *Pattern Recogn.*, 40(7):1946–1957, 2007.

[69] James G. Nagy and Dianne P. O'Leary. Restoring images degraded by spatially variant blur. *SIAM Journal on Scientific Computing*, 19(4):1063–1082, 1998.

[70] R. Neelamani, H. Choi, and R. Baraniuk. Forward: Fourier-wavelet regularized deconvolution for ill-conditioned systems. *IEEE Trans. Image Process*, 52:418–433, 2004.

[71] S. Olsen. Noise variance estimation in images. Proc. 8th SCIA, Tromsø, Norway,, may 25-28 1993.

[72] Jiaya Jia Qi Shan, Wei Xiong. Rotational motion deblurring of a rigid object from a single image. In *International Conference on Computer Vision (ICCV), Rio de Janeiro, Brazil*, 2007.

[73] K. Rank, M. Lendl, and R. Unbehauen. Estimation of image noise variance. *Vision, Image and Signal Processing, IEE Proceedings-*, 146(2):80–84, 1999.

[74] Alex Rav-Acha and Shmuel Peleg. Two motion-blurred images are better than one. *Pattern Recogn. Lett.*, 26(3):311–317, 2005.

[75] Ioannis M. Rekleitis. Optical flow recognition from the power spectrum of a single blurred image. In *International Conference in Image Processing*, Lausanne, Switzerland, Sep. 1996. IEEE Signal Processing Society.

[76] Ioannis M. Rekleitis. Steerable filters and cepstral analysis for optical flow calculation from a single blurred image. In *Vision Interface*, pages 159–166, Toronto, May 1996.

[77] S. Ribaric, M. Milani, and Z. Kalafatic. Restoration of images blurred by circular motion. In *Image and Signal Processing and Analysis, 2000. IWISPA 2000. Proceedings of the First International Workshop on*, pages 53–60, 14-15 June 2000.

[78] C.A. Rothwell, A. Zisserman, C.I. Marinos, D.A. Forsyt h, and J.L. Mundy. Relative motion and pose from arbitrary plane curves. *IVC*, 10:250–262, 1992.

[79] M. Ruzon and C. Tomasi. Alpha estimation in natural images. In *Proceedings of CVPR 2000*, 2000.

[80] A. A. Sawchuk. Space-variant image motion degradation and restoration. *IEEE Proceedings*, 60:854–861, 1972.

[81] D.H. Shin, R.H. Park, S. Yang, and J.H. Jung. Block-based noise estimation using adaptive gaussian filtering. *Consumer Electronics, IEEE Transactions on*, 51(1):218–226, 2005.

[82] Alvy Ray Smith and James F. Blinn. Blue screen matting. In *SIGGRAPH '96: Proc. of the 23rd annual conference on Computer graphics and interactive techniques*, pages 259–268, New York, NY, USA, 1996. ACM Press.

[83] Jian Sun, Jiaya Jia, Chi-Keung Tang, and Heung-Yeung Shum. Poisson matting. In *ACM SIGGRAPH 2004 Papers*, pages 315–321, New York, NY, USA, 2004. ACM Press.

[84] R. Takiyama and N. Ono. Restoration of rotationally blurred images. In *Pattern Recognition, 1988., 9th International Conference on*, pages 1020–1022vol.2, 14-17 Nov. 1988.

[85] Marius Tico and Markku Vehvilainen. Bayesian estimation of motion blur point spread function from differently exposed image frames. In *In Proceedings of 14th European Signal Processing Conference (EUSIPCO), Florence, Italy, September 4-8, 2006*, 2006.

[86] P. H. S. Torr and D. W. Murray. Outlier detection and motion segmentation. In P. S. Schenker, editor, *Sensor Fusion VI*, pages 432–443. SPIE volume 2059, 1993. Boston.

[87] Bora Uçar, Cevdet Aykanat, Mustafa C. Pinar, and Tahir Malas. Parallel image restoration using surrogate constraint methods. *J. Parallel Distrib. Comput.*, 67(2):186–204, 2007.

[88] Christopher B. Webster and Stanley Reeves. Radial deblurring with ffts. In *Proceedings of ICIP International Conference of Image Processing, Sant Antonio, USA*, 2007.

[89] Martin Welk, David Theis, and Joachim Weickert. Variational deblurring of images with uncertain and spatially variant blurs. In *DAGM-Symposium*, pages 485–492, 2005.

[90] Y. Wexler, A. Fitzgibbon, and A. Zisserman. Bayesian estimation of layers from multiple images. In *Proc. of European Conference on Computer Vision (ECCV)*, 2002.

[91] L. Yaroslavsky. *Digital Holography and Digital Image Processing: Principles, Methods, Algorithms*. Kluwer Accademic Publishers Boston, 2004.

[92] Y. Yitzhaky, Mor I., Lantzman A., and N. S. Kopeika. Direct method for restoration of motion-blurred images. *Journal of the Optical Society of America A*, 15:1512–1519, June 1998.

[93] Y. Yitzhaky and N. S. Kopeika. Identification of the blur extent from motion-blurred images. In G. C. Holst, editor, *Proc. SPIE Vol. 2470, p. 2-11, Infrared Imaging Systems: Design, Analysis, Modeling, and Testing VI, Gerald C. Holst; Ed.*, pages 2–11, May 1995.

[94] Y. Yitzhaky and N. S. Kopeika. Identification of blur parameters from motion blurred images. *Graph. Models Image Process.*, 59(5):310–320, 1997.

[95] Y. Yitzhaky, R. Milberg, S. Yohaev, and N. S. Kopeika. Comparison of direct blind deconvolution methods for motion-blurred images. *Applied Optics*, 38:4325–4332, July 1999.

[96] Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Image deblurring with blurred/noisy image pairs. *ACM Trans. Graph.*, 26(3):1, 2007.

[97] Xiaohui Yuan and Bill P. Buckles. Subband noise estimation for adaptive wavelet shrinkage. *icpr*, 04:885–888, 2004.