Analysis of a Subpixel Point Spread Function Estimation Algorithm

Master Thesis in Electrical Engineering

Mauricio Delbracio Bentancor

Advisors: Prof. Andrés Almansa, Prof. Pablo Musé

Instituto de Ingeniería Eléctrica
Facultad de Ingeniería
Universidad de la República

Montevideo, Uruguay
November 2009

Contents

1 Introduction

The price of compact digital cameras has dropped down in the last decade, mostly due to electronic devices decreasing costs, causing their massive market penetration. Unfortunately, image quality has not increased as fast as digital cameras sales have done. Digital images present certain characteristical problems caused by the intrinsic nature of these devices. Blur produced by light diffraction, geometrical distortions caused by the use of low-cost lenses and thermal noise due to electronic circuits are examples of this kind of outcomes.

In this report we describe a mathematical digital camera image formation model that takes into account the whole process of digital image acquisition and all their associated effects. Image blur can be a consequence of camera misusing (e.g. wrongly setting the camera focal distance), but also of physical camera phenomena as light diffraction or sensor averaging. Our goal is to accurately estimate a function, called *point spread function* -PSF, that models the blur due to intrinsic camera phenomena. This function can be locally interpreted as the impulse response of a linear translation invariant system.

The most important application of PSF estimation is image super-resolution. Image super-resolution is the process of enhancing the resolution of an imaging system. There are at least two different ways of approaching this problem: by interpolating a single-frame image or by fusing together several low-resolution images. If the PSF is known at an accurate subpixel resolution, then we can use this information to apply the inverse de-blurring procedure (also known as image deconvolution) to a single low resolution image, or we can fuse all the low resolution images in a more elaborated way. In most typical digital cameras, images are acquired at a sampling rate under the Nyquist rate, causing aliasing effects. While at first sight this might be seen as a drawback, the acquired image has "hidden" information of higher frequencies components. If we do not impose a model over the original image (e.g. regular in some sense) the process of recovering the high resolution image is irreversible. By subpixel estimating the PSF, we learn how the frequencies are mixed, which is very useful information for recovering the high resolution image. The proposed subpixel PSF estimation algorithm is strictly connected with the proposed image formation model.

Precise PSF estimation is also of great interest for the Modulation Transfer Function - MTF estimation problem. The MTF is the Fourier transform of the PSF and is widely used by lens sellers, like Canon or Nikon, to describe the quality of a lens. They offer the lens MTF as a chart showing the response of the system to thicker sets of parallel periodic straight lines. The patterns are presented to the camera at some particular orientations, and the camera response is measured at certain locations of the image plane (see [1] for a description). This information given by optical designers is not exactly the Fourier transform of the PSF but is closely related to it. MTF charts usage for image quality evaluation is widely accepted by the photography community making PSF estimation a relevant topic for the area.

Although there exists a significant amount of work in blind deconvolution associated to image restoration, very little has been done in accurate PSF estimation. Existing PSF estimation methods can be classified into two categories: blind PSF estimation and non-blind PSF estimation. Blind methods estimate the PSF without knowing the target scene: that is to make the estimation from a single or a set of "blurred" images. On the other hand, non-blind methods do the estimation using a known pattern calibration image. In the present work we propose a non-parametric PSF estimation method using a non-blind approach. For that purpose, we use a specially designed grid pattern that captures local blur information allowing local subpixel PSF estimation (see Figure 1). Our work is based in [2, 3] where the authors proposed an algorithm for both blind and non-blind subpixel PSF estimation.

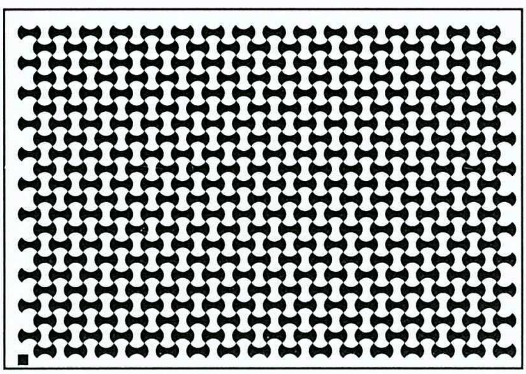


Figure 1: Gird pattern for local PSF estimation

The grid pattern presents checkerboard corners that are used to locally register it. Because of the aliasing problem present on digital images, the task of detecting the corners at subpixel precision is not trivial. We analyze different corner detection algorithms and we compare their performance depending on the aliasing and noise power levels.

Using the detected local pattern features we estimate the geometrical distortion introduced by the digital camera. We can model this distortion as a mapping between 2D planes (i.e. the printed grid pattern and the digital image). There are several sophisticated models for geometrical lens distortion. However, according to the literature [4, 5] lens distortion is generally completely dominated by the radial components. In this work we explore two ways of modeling geometrical lens distortion. First, as a composition of a homography and a radial distortion, and second as a non-parametric smooth 2D to 2D mapping, modeled by thin-plates splines. We show that the non-parametric spline model is more accurate.

Finally, as we know the geometrical transformation between the grid pattern and the captured digital image, we can search for the kernel that convolved with the grid pattern and distorted by the mapping, gives the best explanation of the observed image. We formulate this problem as a variational minimization adding a regularization constraint for stability. As we do the grid pattern registration locally we can find a local PSF. This also allows us to estimate other non local distortions (e.g. vignetting).

Most digital cameras have only one CCD photo-sensor-array, so in order to acquire color information, each photo-sensor is filtered to capture only the wavelengths of a particular band for the red, green, and blue channels. In order to avoid the preprocessing done by the camera built-in software we use raw image data (i.e. data accessed directly from the camera sensor) and compute a PSF for each of the channels. This also gives us an idea of the camera chromatic aberrations caused by the fact that different wavelengths focus at different planes.

In order to validate the proposed methodology we performed several tests with both simulated and real data. Using simulated data we were able to assess the correct performance of our PSF estimation algorithm. Good results were obtained even in the presence of noise and aliasing due to under-sampling. In the case of real data tests the estimation could not be directly validated as the ground-truth about the real camera PSF is not known. Instead, we use the super-resolution problem to indirectly evaluate our PSF estimation. Our goal is not to develop a state-of-the-art super-resolution algorithm, but to show the advantages of having an accurate PSF estimation for this problem. Within a Bayesian framework, we propose a single image deconovolution/super-resolution algorithm that uses the subpixel PSF estimation to find the a posteriori most probable super-resolved image. We based our work in novel results on natural image statistics which justify a sparse model on the image gradient.

This manuscript is organized as follows. In Section 2 we overview relevant work on PSF estimation and image deconvolution. In Section 3 we describe a mathematical digital camera model which will rule our PSF estimation.

In Section 4 we present our work in subpixel PSF estimation using a known calibration grid pattern and all the associated problems. In Section 5 we present experimental results generated with both simulated and real camera data. In Section 6 we introduce a single image superresolution algorithm that is based on our subpixel PSF estimation while in Section 7 we analyze its performance through simulations and real digital camera images. Finally, in Section 8 we close with a discussion of our work and present some ideas for future work.

2 Review of PSF Estimation Methods

There are several causes of image blur. Some of them, like light diffraction, are direct consequence of the optical system and of unavoidable physical phenomena. Others, like out of focus, are caused by the configuration of the scene or the photograph expertise. In this section we present an image formation model, which will be the cornerstone of the PSF estimation approach. First, because we need to know if it is possible to model a digital camera response with a PSF but also because our estimation will be conditioned by the adopted model. In the second part of this section, we review some of the existing methods for PSF estimation and how they are associated to the blind deconvolution problem.

The goal of PSF estimation is to recover the kernel that causes an image to have blur. Nevertheless, as we have shown in the last section, this is strictly attached to the image formation model that is considered. Most of the existing work estimates the best linear shift invariant kernel PSF, known as point spread function, which convolved with the true image generates the observation,

$$g(i,j) = f * \mathrm{psf}(i,j) + n(i,j) = \sum_{(k,l)} f(i,j)psf(k-i,j-l) + n(i,j) \qquad (1)$$

where $i,j \in \mathbb{Z}$ and n is a stochastic process modeling the random noise.

From this equation it is clear that if we want to estimate the PSF from the observation g we also need to estimate f. There are two big different approaches for PSF estimation: blind estimation and non-blind estimation. The first one considers that the original *true* image f is unkwnown, so the pair (f,PSF) is estimated simultaneously in order to give a good interpretation of g. The second approach assumes that the *true* image f is given and known so it consists in finding the best kernel PSF which convolved with the true image f results in g. Intermediate approaches, where some information on the nature of f is known, have also been proposed, but are less common.

Although there exists a lot of work in blind deconvolution associated to image restoration, there exists little work on accurate PSF estimation. We refer to [6] for a complete survey on blind deconvolution, and to [7] for a state of the art blind deconvolution algorithms performance evaluation.

Some researchers have approached the PSF estimation problem by estimating the modulated transfer function (MTF) of a system. The MTF of an optical system is an accepted way of describing its optical properties and its quality [8]. If the convolution model of Eq. 1 is accepted, then by considering its Discrete Fourier Transform we get,

$$G(u,v) = F(u,v) \cdot \mathcal{F}(\mathrm{psf})(u,v) \qquad (2)$$

where $\mathcal{F}$ is the Discrete Fourier Transform (DFT), F and G the DFT of f and g respectively. The MTF is defined as

$$\mathrm{mtf} = \mathcal{F}(\mathrm{psf})$$

Hence, the problem of PSF estimation and MTF estimation are equivalent.

The MTF is widely used in consumer lens sellers like Canon or Nikon to describe the quality of a lens. However, they offer the lens MTF as a chart showing the response of the system to thicker sets of parallel periodic straight lines. The patterns are presented to the camera at some particular orientations, and the camera response is measured at some particular locations of the image plane. Hence, the MTF is given as a set of lines (figure 2). This information given by optical designers is not exactly the Fourier transform of the PSF but is closely related to. Here we do not pretend to analyze how to read an MTF chart (see [1] for a description), but our intention is to remark the importance of its estimation as it is well accepted by the photograph community.

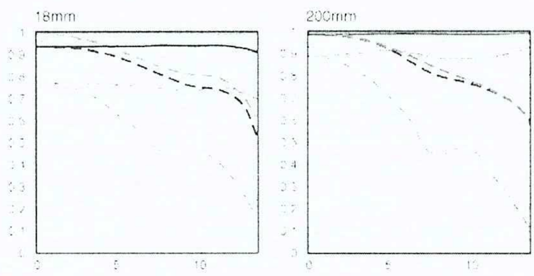


Figure 2: Sample MTF chart from a Canon EF-S 18-200mm f/3.5-5.6 IS lens.

In [8, 9] the authors proposed a novel approach for MTF estimation by using a random target. The best advantage of this method is that it does not need to align or even to know the true pattern image. It only needs to know that it is a realization of a white noise process. Then, based on the fact that the input image has a flat spectrum, the estimate of the MTF follows directly from the computation of the power spectral density of the acquired image. However, as much of the work on MTF estimation, they do not estimate a bi-dimensional MTF. Instead they suppose that the PSF is symmetric, so it is sufficient to give the MTF in a particular direction. As this could not always be a valid hypothesis, several authors and lens sellers present the MTF information as a set of curves reflecting the changes in different particular directions.

Another area which is closely related to PSF estimation is blur estimation which consist of estimating the level of blur in an image. Most of the work in this area assumes a simple PSF parametric model, such as a Gaussian function in [10]. However in [11] the author proposes a novel way of analyzing the blur formation. It is shown that the convolution model is imprecise especially when object occlusion occurs. In the best case where the convolution model is valid, an estimation of the local blur level is given. The work is based on the analysis of the topographic map by using the mean curvature motion filtering to find the locations where the blur estimation can be done.

In order to review the existing methods of PSF estimation we separate them arbitrarily in three arbitrary categories: blind PSF estimation, non-blind PSF estimation and parametric PSF estimation. These categories are not mutually exclusive, in particular the parametric methods belong to either the blind or non-blind category.

Parametric PSF Estimation In Section 3 we will describe the physical process that governs the image blur. There are some existing approaches that try to estimate the PSF in particular cases supposing a simple parametric PSF model. For example in [12] the authors propose a model for the out of focus blur. However more complex blur, like the one generated by general motion blur or simple, an accurate estimation of the PSF due to diffraction, cannot be represented by such a simple model.

Several methods within this category use specific information about the *true* image, such as being formed by perfect point sources or by step-edges, in order to estimate the parameters of the PSF model. The most popular ones use the frequency domain zeros of the acquired image to perform the PSF estimation [12]. From the simple model presented in Eq. 2 and supposing there is no noise, the observed zeros of G give some information about the zeros of the MTF. Then, in order to completely characterize the MTF, one needs to assume that the PSF is of a known parametric form and that given its frequency domain zeros, its associated parameter values can be uniquely determined.

As out of focus, or simple motion blur estimation is not the goal of our work we do not dwell further on this. Behind low order parametric PSF models, we do not know any method of parametric PSF estimation. Possibly due to the complexity of the physical process behind the acquisition. In particular, according to what we show in Section 3, the camera aperture shape need to be parametrized in a precise way.

Non-Blind estimation It is important to remark that even in the case we know exactly the *true* image, image deconvolution is an ill-posed problem due to the loss of information during the blurring process and the noise presented in the observation. Thus, the most important thing is to develop methods which impose some prior knowledge in order to disambiguate the inverse problem.

The principal methods for image deconvolution are the Lucy-Richardson's algorithm, the Wiener filter deconvolution, least-squares deconvolution and deconvolution based in image priors derived from natural image statistics [13].

The first three methods are classical ways of facing image deconvolution problem, while the last one is a novel significant improvement based in an sparse image gradient model. This sparse assumption seems to be reasonable in natural images. However, in our case in which what we want to estimate is the PSF a simpler least squares with some reasonable prior should be more appropriate.

Blind estimation Blind image deconvolution is one of the most challenging topics in image processing. It is by definition a very ill-posed problem, so some constraints on the image and also in the PSF should be imposed. For a formal definition of the problem where the most important aspects are considered see [6].

The most classical assumptions for the PSF are:

- PSF values are non negative. In other words blurring in is a purely additive process.
- PSF preserves image energy. That is $\int \mathrm{psf}(\mathbf{x})d\mathbf{x} = 1$.
- PSF is symmetric. Central symmetry along its barycenter or radially which is a stronger hypothesis.
- PSF is known in a parametric way. We have already commented this case.

Notice that while the first two assumptions are very reasonable (as they should not eliminate any valid PSF), the rest are much more restrictive, as they suppose particular shapes of PSF kernels which in practice eliminate *real* camera PSFs.

In order to give an accurate PSF estimation, it is important to also constrain the class of *true* images to be utilized. As much as we restrict the set of possible images, we can utilize more prior knowledge on the input set, and therefore we can stabilize the original ill-posed deconvolution problem. A typical assumption is that the edges presented in an image are step-edges, so in practice it is possible to recover the *true* image from the degraded observation. For example, this is done in [2]. Then a non-blind deconvolution algorithm is used to estimate the PSF. Other assumptions are that the images are formed by point sources as found in astronomical images.

Nevertheless, as the goal of this work is to give a precise subpixel PSF estimation we opted for a non-blind PSF approach. We do not discard to explore blind estimation approaches in future work.

3 Digital camera image formation model

The formation of a digital image implies several physical processes that convert the 3D world scene into a bounded digital 2D image. The goal of this Section is to formalize the mathematical model behind this procedure.

Up to our knowledge there is no suitable universal camera model. The main reason, is the difficulty in giving an accurate model of the whole image acquisition chain, both because of its variability and its complexity. Nevertheless, a lot of researchers typically assume a pin-hole camera model, that is a camera with no lens and a very small (a point) aperture. By assuming this, the process of image formation consists only in a perspective projection which maps the 3D world into a 2D plane. The pinhole model is an ideal model: in practice a camera will have a lens which can introduce some geometrical distortions. The aperture cannot be so small because it could produce high diffraction effects and a lens is needed to concentrate light in the aperture (otherwise the image would be extremely dark). This leads to a series of effects that are not taken into account by the classic pin-hole model: diffraction, averaging due to the non infinitesimal aperture size, geometric and chromatic aberration introduced by the lens, etc.

The model adopted here is a generalization of the ideal pin-hole camera model. It considers the diffraction effect due to finite camera aperture, the out of focus due to setting the focus only for a specific depth, and also the digitization process. We also incorporate a geometrical distortion transformation in order to contemplate possible lens distortions. As typically lens systems are constructed from a series of individual lenses centered on a common axis, it is difficult to precisely define a model for the lens system. In the literature the one which seems to be the most realistic model is presented in [14]. Figure 3 shows a diagram of the image acquisition process.

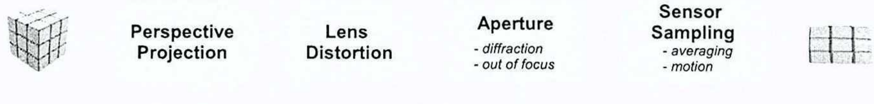


Figure 3: Image formation Model.

Perspective Projection

A 3D perspective projection is a geometric transform that maps 3D world points to a 2D plane. The geometry behind this projection involves treating the 2D projection as being viewed through a camera with a point aperture. A perspective projection maps 3D straight lines in 2D lines or 2D points.

and 3D points in 2D points. Mathematically, if we consider a plane P in 3D space its perspective projection can be described by an *homography*, which is defined by 8 parameters or by the image of 4 points in generic position [15]. In Section 4.3 we comment on how to determine the parameters of the homography by taking advantage of the grid pattern.

In practice, geometric distortions occur causing that perfect lines in the 3D world are not exactly projected as 2D lines. Consequently, the lens distortion has to be considered in the camera model.

Lens Distortion

There are several sophisticated models that try to model lens distortion, however according to the literature [4, 5] in general the lens distortion is completely dominated by the radial components. The classical model for lens distortion is to consider a radial distortion governed by a low degree polynomial. This requires to determine the center of the distortion (a point in the image plane which does not suffer from distortion) and the coefficients of the polynomial. In Section 4.3 we discuss this phenomena and we also give an approach to compute the radial distortion.

Finite Aperture

The finite aperture has two consequences: diffraction and out of focus.

The first and most important is the diffraction due to a non-infinite aperture. It can be modeled as a convolution, we refer the reader to [15] *Theorem 1* (Fraunhofer Diffraction). The diffraction kernel is determined by the aperture's shape and size, the focal length and by the wavelength of the considered monochromatic light. Therefore, in theory each of the three channels RGB will have a different diffraction. If the shape and size of the aperture is known it can be explicitly computed. In practice, a circular aperture is assumed, leading to the following diffraction kernel (Proposition 2 [15]):

$$k_{\text{diff}}(x, y) = C \cdot \left(\frac{2J_1(r)}{r}\right)^2, \qquad \text{with } r = \frac{\pi D\sqrt{x^2 + y^2}}{\lambda f}, C = \frac{\pi^2 D^4}{32 f^2}.$$

The function $J_1(r) = \frac{1}{\pi}\int_0^\pi \cos(\theta - t\sin\theta)d\theta$ is the Bessel function of first kind and order 1, f is the focal length, D the aperture diameter, and λ the wavelength.

Notice that in this case of circular aperture, the radius of the diffraction kernel (approximated as the first zero crossing of $J_1(r)$) is $r_a \approx 1.22\frac{\lambda f}{D}$. This is called *airy pattern* and gives a reasonable estimate of the optical system resolution. Note, that the size of the central spot depends only on the so called F-number $= \frac{f}{D}$ and the wavelength λ. Figure 4 shows a real camera aperture: the circular aperture hypothesis is doubtfully reasonable.

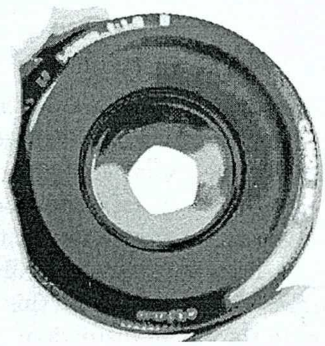


Figure 4: Real camera aperture. Image taken from [13]

The second consequence of a non-punctual aperture is the out of focus. The out of focus effect appears when the scene presents several objects at different depths and the picture cannot capture all of them in perfect focus. If the imaged scene is a plane parallel to the camera plane, we can always take the photograph in perfect focus. For different ways of describing the out of focus see [15] section 1.2.3 where a mathematical description in terms of the aperture shape is presented or [12] where a simple circular disk kernel is considered.

Since both problems are consequence of the same part of the acquisition system process, the aperture kernel can be modeled as

$$k_{\text{aperture}} = k_{\text{diff}} * k_{\text{defocus}}$$

However, we point out that the goal of this work is to estimate the PSF only due to the camera hardware and not to take into account extrinsic effects such as the out of focus.

The Sampling Process

The digitization process is performed by a rectangular grid of photo-sensors (CCD or CMOS) located in the focal plane. Each photo-sensor integrates the light arriving at a particular exposure time. Since all the analysis presented here considers monochromatic light we work independently with each of the RGB camera channels. Most digital cameras have only one CCD array, so in order to acquire color information, each photo-sensor is filtered to capture only wavelengths of a particular band for the red, green, or blue channels. This is done by the Bayer filter mosaic, which covers the sensor plane with 50% of green filters and 25% of blue and red filters respectively (see Figure 5). The image formed by the data as it comes directly off the sensor array is called RAW image. Then the camera built-in software interpolates these patterns to get information of the three colors in each pixel. This process is called *demosaicking*.

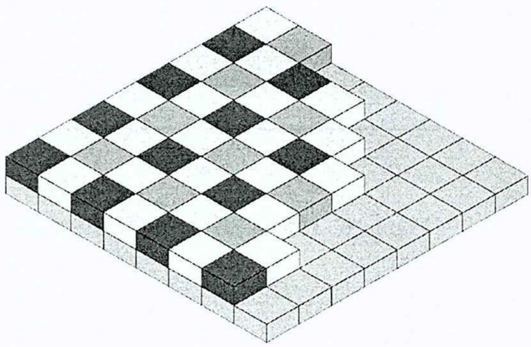


Figure 5: Typical Bayer pattern. Image taken from [16].

Sensor light integration can be modeled by a convolution with a kernel $k_{\text{sensor}} = \mathbf{1}_C$, the indicator function of the photo-sensor region C. Also, as the sensor does not have a linear response for low and high energies, instead of acquiring the digital image I_d we will acquire the image $g(I_d)$ where $g(\cdot)$ is a unknown non linear increasing function. In Figure 6 we show an example. For that reason, it should be avoided the work in dark places as well as very illuminated ones.

If relative motion between the camera and the scene during image exposure time exists, the acquired image will present motion blur. In the case that the relative camera-scene motion is constant all over the scene, the motion blur can be modeled as a convolution with a single kernel. In a general situation, the motion blur is not translation invariant therefore it cannot be expressed as a convolution. Nevertheless, if the motion is constant by regions, it can be locally described as a convolution with a particular kernel.

Since the photo-sensors are electronic devices they suffer from random noise. Also as the recorded intensities are quantized on a finite number of levels, this will also produce quantization noise. Finally instead of acquiring the digital image I_d, the camera captures the image $g(I_d) + n$, where n models the random noise. Typically the noise variance increases with the

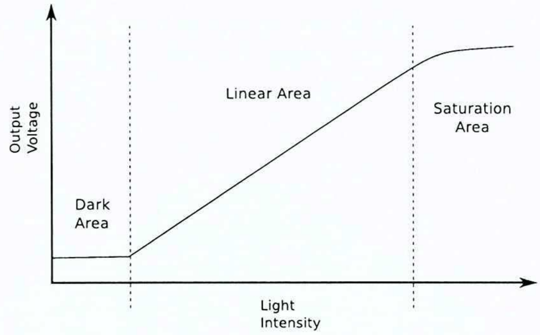


Figure 6: Typical light sensor Response.

intensity level.

In order to avoid the effect of the demosaicking algorithm, we access directly the RAW image file, which contents the Bayer image. Then we can compute a PSF for each of the channels. Notice that due the conformation of the Bayer pattern, we have the double of samples in the green channel. Consequently, the green channel PSF estimation should be the most accurate one.

Adopted Model

In this work, the goal is not to make a parametric camera model, however in order to check if it is possible to estimate a single PSF we need to define the order in which each of the acquisition system stages occurs.

It seems reasonable to consider: first a projective transformation which maps the 3D world scene into a 2D image plane, then a geometric distortion due to the lens system, followed by diffraction phenomena due to finite aperture, then a perspective transformation which maps the lens image plane into the CCD camera plane and finally the sampling process in the CCD array.

This can be formally described by the following model:

$$(M_0) \quad I_d = \Pi_{\mathbb{Z}^2} g(H_2(R(H_1(I)) * k_{\text{aperture}})) * k_{\text{CCD}}) + n,$$

where:

- $H_1(\cdot)$ is the projective transform from 3D to 2D world.
- $H_2(\cdot)$ is the projective transform from 2D lens image plane to 2D CCD image plane.
- $R(\cdot)$ is the distortion function due to the lens system.

- k_{aperture} is a convolution kernel due to the diffraction in the aperture.

- k_{CCD} is a convolution kernel due to the light integration in the sensor CCD.

- $g(\cdot)$ is a monotone increasing function due to the not-linear CCD response.

- $\Pi_{\mathbb{Z}^2}$ is the bi-dimensional ideal sampling operator due to the CCD array.

- n models all the noise present during the acquisition process.

This model is too complex to work with, and also it cannot be described by using only one convolution kernel. So we will consider an approximation model, in which we will concentrate all the PSF like effects into a single PSF, that is

$$(M_1) \quad I_d = \Pi_{\mathbb{Z}^2} g\left(R\left(H(I)\right) * k\right) + n$$

where, $H(\cdot)$ is the projective transform from 3D to 2D world and k is a convolution kernel due to all PSF like effects.

Another simplified model can be derived if we consider the elementary stages in a different order:

$$(M_2) \quad I_d = \Pi_{\mathbb{Z}^2} g\left(R\left(H\left(I * k\right)\right)\right) + n$$

where we imposed that the first process is the convolution with the PSF and then the rest. The model (M_1) seems more realistic than (M_2), as the order of the stages seems to be more appropriate. In Section 4.5, where we present our PSF estimation approach, we will comment on the differences between considering each of these model.

4 Proposed Approach to the PSF estimation

There exist several methods to estimate the PSF, in Section 2 we mentioned some of them. As we have previously said, the goal of this work is to do an accurate subpixel PSF estimation, and with that in mind the most reasonable is to do a non-blind estimation. More specifically, our work is strongly based in [2, 3] where the authors proposed a novel method to do a superresolved PSF estimation in blind and non-blind conditions.

In Section 3 we presented an image formation model that takes into account all phenomena involved the acquisition of a digital image. The purpose of this Section is to analyze how accurately we can estimate the PSF by following the image formation model. First we introduce our non-blind approach for PSF estimation, then we discuss how to choose a suitable grid pattern. Next, we analyze all the necessary steps to do a correct estimation of the PSF, always bearing in mind that the observed images are aliased.

The idea behind our approach is to solve an inverse problem in order to find the PSF. By using prior information about the smoothness of the PSF we can make the inverse problem well posed. This can be formulated by considering one of the image formation model, for example (M_2) and choosing k to minimize the functional:

$$L = \|g(R(I * k))) - I_d\|^2 + \lambda\|\nabla k\|^2,$$

where I is the sharp grid pattern image and I_d the blurred degraded observation. We have included a regularization term which penalizes kernels with large gradient. The regularization parameter λ is related to the noise level but also to how over/under determined is the system.

As inferred by the above problem, if we want to estimate the PSF by a non-blind method, we will have to face (explicitly or implicitly) the following problems:

- Choosing a good grid pattern.
- Getting rid of the non-linear CCD reponse - $g(\cdot)$ function.
- Getting rid of the geometric transformation -$R(\cdot)$ function.
- Estimation of the sharp grid pattern image I. We know the pattern as it is not-blind estimation but we do not know its intensity value and also we do not know its alignment to the observed image.
- Numerical algorithms for solving the PSF.

4.1 Choosing a grid pattern

There are several ways of producing a good grid pattern. Suppose, we could do a perfect delta-like grid pattern (i.e. an image containing just a single point of zero measure), then its Fourier transform would be constant, and finally the observed image would be exactly the PSF. However, in practice it is impossible to do this as we cannot print such an image.

One easy way of implementing the idea behind this is considering a pattern with perfect step-edges. In this case, the observed image will give us information about the PSF response in the direction orthogonal to the edge. Then, we need the pattern having edges in all directions. Also, if we want to do a local PSF estimation we need a pattern having edges in all directions of every location.

As a previous step to tackle the PSF estimation, we need to align the pattern and correct the geometrical transformation due to perspective and lens distortion. For that purpose, the pattern has to be locally alignable. For example, it has to contain some local features that are easily detected.

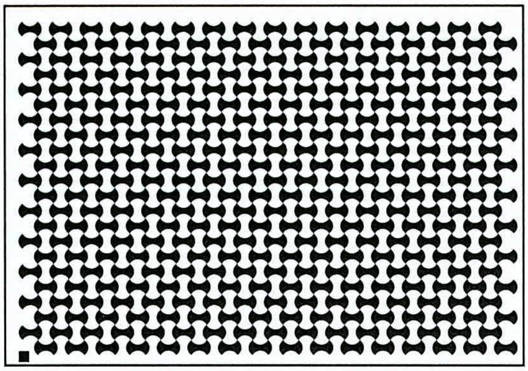
In this work, we used the grid pattern proposed by Joshi et al. [2] which consists of perfect 120° arcs connected in a way that they produce checkerboard like corners (see Figure 7). We have drawn the grid in vectorial form (postscript language) to be able to rasterize it at any resolution.

We leave for future research the study of other ways to produce good pattern grids. In particular, we would like to study the generation of white-noise-like grid patterns.

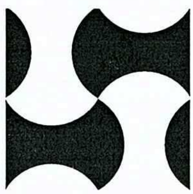
4.2 Grid Alignment

The approach adopted here assumes that we know the input to the system. As we know beforehand the pattern that the camera is acquiring, we only need to align it to the observation. The main problem is that we do not know how the acquisition system (i.e. digital camera) degrades the pattern image, since this is what we want to measure.

Following either models (M_1) or (M_2), presented in Section 3, in order to deal with geometric distortions the ideal pattern and its observation have to be locally aligned. Notice that actually, if the PSF does not exhibit radial symmetry misalignment due to its influence can occur. For that purpose we detect the checkerboard like corners, and if we suppose that the PSF is symmetric then this x-corners will not suffer from shrinkage. Several methods to detect corners have been reported in the Computer Vision literature,



(a) grid pattern



(b) local grid pattern

Figure 7: Gird pattern for local PSF estimation

ranging from differential operators such as Harris detector to more specific correlation methods.

4.2.1 Harris Corner Detector

Harris and Stephens [17] analyzed this problem based on the local autocorrelation function of an image. The local autocorrelation function measures the local changes withing an image by shifting patches a small distance in all directions. Given a shift (x, y) is defined as:

$$E(x, y) = \sum_{u,v} w(u, v) \left(I(u + x, v + y) - I(u, v) \right)^2 ,$$

where $w(u, v)$ is a smoothing window (e.g. a Gaussian) centered in (x, y) and $I(\cdot, \cdot)$ denotes the image intensity. Then approximating $I(u + x, v + y)$ by its first order Taylor expansion

$$I(u + x, v + y) \approx I(u, v) + (I_u(u, v), I_v(u, v)) \cdot [x, y]^t .$$

and replacing this in $E(x, y)$ we have

$$E(x, y) \approx [x, y]M[x, y]^t$$

where $M = \sum_{u,v} w(u, v) \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix}$ captures the intensity behaviour of the local (x, y) neighborhood.

Then observing the eigenvalues of M it is possible to construct a rotationally invariant descriptor:

- Both eigenvalues are small meaning the image region has constant intensity.
- One eigenvalue is high and the other low, then there is an abrupt change in the image in one direction and little change in the orthogonal direction. This indicates the presence of an edge.
- Both eigenvalues are high, meaning that shifts in any direction will result in a significant change in the intensity image. This indicates the presence of a corner.

Harris-Stephens proposed to measure this by, $R = \det(M) - k\text{trace}(M)^2$ where $k \in [0.04, 0.15]$.

Then big R values will mean a presence of a corner in the point (x, y).

There exist multi-scale and affine invariant generalizations of the Harris-Stephens corner approach [18], but none of them were considered in this work. As we work under a totally controlled situation, we can always know exactly the scale at which we are working. We concentrated our work in studying how we can refine the corner detector to get subpixel accuracy.

4.2.2 Sub-pixel Corner Detection

The procedure presented in last Section does not give subpixel precision as we are computing the R value only in the grid defined by image pixels. In order to get subpixel accuracy we studied three different variants to refine the initial position given by the Harris-Stephens algorithm.

Image Interpolation In [19] the authors proposed to first apply a Harris-Stephens corner detector at pixel level and then interpolate the image intensity in neighbours where corners were detected. Finally we could apply the corner detector to each of the interpolated neighbours and if the interpolation is precise we could get subpixel accuracy. The main drawback is that if the image is aliased then we do not know how to correctly interpolate it. In practice, we used an iterative algorithm where we smooth the image (as an antialias preprocessing) before interpolating it, then we run the Harris-Stephens algorithm. By doing this iteratively, we can refine the corner position by decreasing the smoothing filter action.

Saddle-points Detector In [20] the authors proposed a method for extracting X-junctions with subpixel precision. First, a Harris-like method is run to detect neighbours that contain X-junctions, then a quadratic function is fit to the intensity profile for each of the detected points. This is done by solving a linear least squares problem,

$$\arg\min_{a,b,c,d,e,f} \|ax^2 + bxy + cy^2 + dx + ey + f - I_{N(x_0,y_0)}(x,y)\|^2$$

where $I_{N(x_0,y_0)}$ represent a neighbourhood of the image $I(x,y)$ centered in the point (x_0, y_0) where the Harris method detected a X-junction. As the critical point of the quadratic function is a saddle point, it is given by the intersection of two lines,

$$\begin{cases} 2ax + bx + d = 0 \\ bx + 2cy + e = 0 \end{cases}$$

And finally the subpixel X-junction (x, y) is located at:

$$x = \frac{be - 2cd}{4ac - b^2}, y = \frac{bd - 2ae}{4ac - b^2}.$$

X-checkerboard Detector In [21] the authors proposed a new method for detecting X-corners. Once they get an initial position (x, y) they refine the localization by using a second order Taylor approximation, that is by assuming that the real position is $(x + s, y + t)$ then,

$$I(x+s, y+t) \approx I(x, y) + [s, t][I_x, I_y]' + \frac{1}{2}[s, t] \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix} [s, t]' \tag{3}$$

where I_{xy} is the second derivate of I respect x and y, and the same for the others.

Then as an X-corner is just a saddle point of the intensity image, we can find it by setting the first derivate of Eq.3 to zero. That is solving,

$$\begin{cases} I_{xx}s + I_{xy}t + I_x = 0 \\ I_{xy}s + I_{yy}t + I_y = 0 \end{cases}$$

Finally the subpixel X-corner position is $(x + s, y + t)$ where:

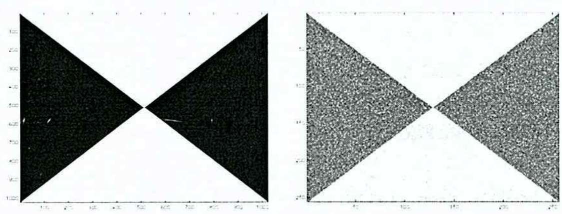
$$s = \frac{I_y I_{xy} - I_x I_{yy}}{I_{xx} I_{yy} - Ixy^2}, t = \frac{I_x I_{xy} - I_y I_{xx}}{I_{xx} I_{yy} - Ixy^2}$$

4.2.3 Evaluation

We tried to simulate the same conditions as how the grid would be acquired. For that purpose we built a checkerboard corner image with perfectly known position. Then we filter the image with a small Gaussian filter and downsampled it 16×. Adjusting the variance σ of the Gaussian filter we can control the amount of alias in the downsampled image. Finally some Gaussian white noise was added to get the `test` image. We ran all tested algorithms in several images varying the alias level and noise level. Figure 8 shows an example image and its degraded version.

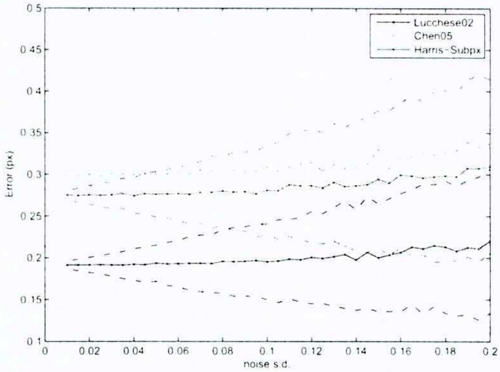
Figure 9 shows the performance of the three evaluated detectors in three different situations: high alias, middle alias, low alias. We repeated the experiment several times for each variance noise and computed the mean error and the standard deviation. In solid lines it is shown the mean value and in dot lines the mean value $\pm\sigma$ stripe. Note that if alias is middle-low all algorithms have an accuracy greater than 0.1 px for medium-low noise. Also, Chen and Harris subpixel algorithm have a very similar performance, and if alias is low Lucchese algorithm works worse. In Figure 10 we show the accuracy vs. σ (small σ high alias - large σ low alias) for additive Gaussian noise of power 0.03 (low noise) and 0.1 (medium noise). In this case, we observed that for $\sigma > 0.6 - 0.7$ all algorithms have a high accuracy (error less than 0.1 px). For highly aliased images the Lucchese algorithm has a better performance while in low alias situations Chen or Harris subpixel algorithm reach a greater accuracy.

Images taken from typical digital cameras are normally middle-aliased ($\sigma = 1 - 1.6$), and as are working in a laboratory we can control the acquisition process to have low noise level. Also, as we pretend to estimate the PSF superresolved at 16× we need a corner detector with accuracy less than 0.1 px. From this evaluation we conclude that in this conditions all algorithms have similar performances, so for simplicity we adopt to utilize the Harris subpixel detector algorithm.

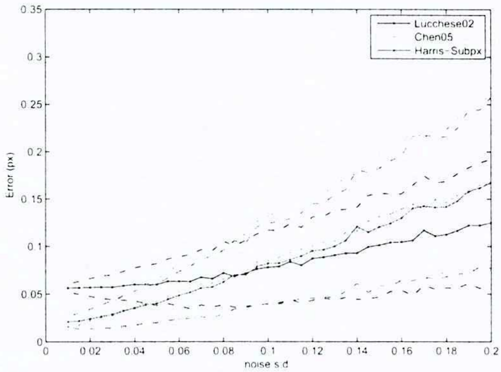


(a) (b)

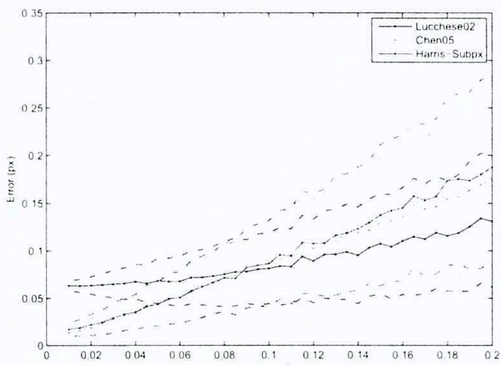
Figure 8: checkerboard corner image example (a) and Aliased 16× downsampled noise corner image example (b).



(a) $\sigma = 0.3$ - high alias

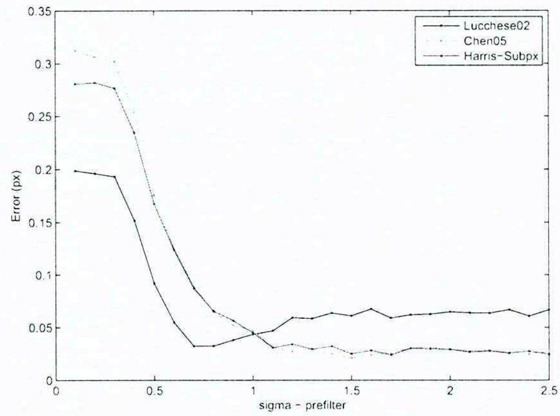


(b) $\sigma = 1.3$ - middle alias

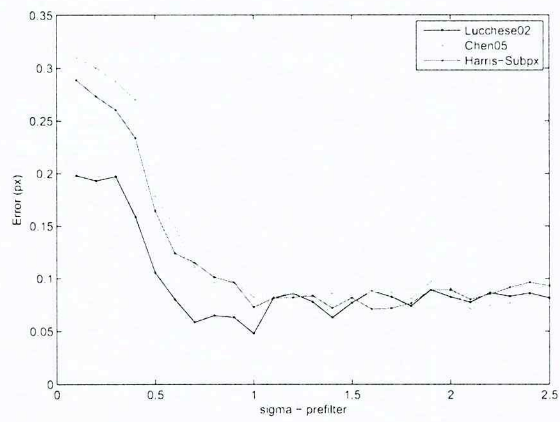


(c) $\sigma = 2.3$ - low alias

Figure 9: Aliased checkerboard corner detector evaluation I. Accuracy (in px. from the downsampled low resolution image) vs. noise standard deviation, prefiltered by a Gaussian of: $\sigma = 0.3$ - high alias (top), $\sigma = 1.3$ - middle alias (center), $\sigma = 2.3$ - low alias (bottom).



(a) noise std = 0.03 - low noise



(b) noise std = 0.1 - medium noise

Figure 10: Aliased checkerboard corner detector evaluation II. Accuracy (in px. from the downsampled low resolution image) vs. Gaussian standard deviation (prefilter). with additive white Gaussian noise : std= 0.03 - low noise (top). std= 0.1 - medium noise (bottom).

4.3 Geometric Transformation and Distortion Estimation

The method studied in this work assumes that we know a subpixel correspondence between digital blurred image pixels and original image. However, there exists a geometric transformation between real world scene and the associated digital image captured in the CCD plane. In practice, from the model that we assumed in Section 3 the mapping between a point in a 3D plane $P = [X, Y, Z]$ and its projection $p = [x, y]$ is given by $p = R(H(P))$, where $H(\cdot)$ model the perspective transformation as a 2D homography and $R(\cdot)$ models the non-projective distortion.

We have studied two different approaches for computing the geometric mapping. First, the classical one (based on the work by Zhang [5]) estimating a homography and a radial distortion, and a general, non-parametric one that directly approximates the geometrical transformation by thin plates splines. Both techniques require to know a set of correspondences; in our case we used the checkerboard detected features as we know their original location in the pattern image.

4.3.1 Homography + Radial Distortion

This geometric transformation is divided into a planar perspective projection of coplanar detected features P_i onto the image plane, $p_i = HP_i$, and radial distortion $\tilde{p}_i = d(p_i)$. In this approach we have not considered tangential distortion while most authors consider it as usually negligible [22], if our model/estimation is not accurate enough we can introduce some artifacts in the estimation of the PSF.

Radial distortion is usually modeled as a polynomial,

$$\tilde{p} = p + (p - p_0)(d_1 r + d_2 r^2 + d_3 r^3 + d_4 r^4) \tag{4}$$

where p_0 is the distortion center $p_0 \in \mathbb{R}^2$, $r = \|p - p_0\|$ and $d_1, d_2, d_3, d_4 \in \mathbb{R}$.

A homography between points lying in a plane (the calibration pattern) to points lying in another plane (CCD) is determined by 4 pairs of corresponding points (8 unknowns). If $P = [X, Y]$ and $p = [x, y]$, and h_{ij} denote the matrix homography entries:

$$x = \frac{h_{11}X + h_{12}Y + h_{13}}{h_{31}X + h_{32}Y + 1}$$

$$y = \frac{h_{21}X + h_{22}Y + h_{23}}{h_{31}X + h_{32}Y + 1}$$

Then, we can rewrite this equation to get a linear system:

$$h_{31}Xx + h_{32}Yx + x - h_{11}X - h_{12}Y - h_{13} = 0$$

$$h_{31}Xy + h_{32}Yy + y - h_{21}X - h_{22}Y - h_{23} = 0$$

Suppose that we know the corresponding pairs $(P_i, p_i), i = 1, 2, \dots, n_c$, of 3D world points in the grid pattern and points to image plane without radial distortion. Then a linear system on (h_{ij}) can be conformed and the problem reduces to solve a linear least square problem:

$$\begin{cases} h_{31}X_1x_1 + h_{32}Y_1x_1 + x_1 - h_{11}X_1 - h_{12}Y_1 - h_{13} = 0 \\ h_{31}X_1y_1 + h_{32}Y_1y_1 + y_1 - h_{21}X_1 - h_{22}Y_1 - h_{23} = 0 \\ \vdots \\ h_{31}X_Nx_{n_c} + h_{32}Y_{n_c}x_{n_c} + x_{n_c} - h_{11}X_{n_c} - h_{12}Y_{n_c} - h_{13} = 0 \\ h_{31}X_Ny_{n_c} + h_{32}Y_{n_c}y_{n_c} + y_{n_c} - h_{21}X_{n_c} - h_{22}Y_{n_c} - h_{23} = 0 \end{cases}$$

From the adopted radial model (Eq. 4), distortion near the center is minimal, so we utilize feature points located at the center of the image to estimate an initial homography. However, notice that this minimization does not minimize the euclidean distance between feature points $\sum \|p_i - HP_i\|^2$.

Instead of simultaneously estimating both the homography and radial distortion parameters that minimize the euclidean distance between feature points, we proceed in an iterative way. Given a homography we estimate the radial distortion that best explains the difference between the observed feature points and the points projected through the homography. Then, by the inverse radial distortion we compute a new set of points and estimate a new homography. We proceed iteratively till the difference between parameters is less than a small threshold. We do this in order to accelerate the convergence of the algorithm as estimating at the same time the whole set of parameters is really slow.

To estimate an initial radial distortion, we compute the projection of all detected feature points P_i through the homography already estimated p_i and solve for the pairs $(\tilde{p}_i, p_i), i = 1, \dots, n$

$$R = \arg\min \sum_{i=1}^{N} \|\tilde{p}_i - R(p_i)\|^2$$

Observe that R is characterized by $(x_0, y_0, d_1, d_2, d_3, d_4)$. This problem is non-linear and we solve it using the Levenberg-Marquart algorithm.

Once we have computed an initial H_0 and R_0, we can proceed in an iterative way. For doing this we need to invert the radial distortion. This can be done by a Newton fixed-point algorithm. Consider r_0 the distance of the distorted pixel to the distortion centre and $d(r) = (d_1r + d_2r^2 + d_3r^3 + d_4r^4)$. Then following the assumed radial model,

$$r_0 = (\tilde{p}_i - p_0) = (p_i - p_0)(1 + d(r)) = r(1 + d(r))$$

and then we can invert $r\,(1 + d(r)) = r_0$ by the following iterative process:

$$r_{n+1} = \frac{d_1 r_n^2 + 2d_2 r_n^3 + 3d_3 r_n^4 + 4d_4 r_n^5 - r_0}{1 + 2d_1 r_n + 3d_2 r_n^2 + 4d_3 r_n^3 + 5d_4 r_n^4}.$$

Once we have estimated the radial distortion parameters we compute $p_i = R^{-1}(\tilde{p}_i)$ and from these points we compute the new homography proceeding iteratively.

4.3.2 Thin-plate smoothing spline

For our purpose of PSF estimation we do not need to separate the distortion in homography and non-homography distortion. The idea behind using thin-plate splines is to avoid that computation and to utilize a more general model. Since we have previously detected the $\{\tilde{p}_i\}$ checkerboard corners from the grid and we know exactly their corresponding points $\{P_i\}$ we can use these correspondences to find a smooth mapping from the non-distorted to the distorted space.

Although thin-plates were originally used as an exact interpolation method [23] they can be easily extended to the approximation problem [24]. Also, it is considered that the problem can be subdivided in two problems, one for each component of the transformation. One way of weaken the interpolation condition is by minimizing the functional:

$$E = \sum_i \|f(P_i) - \tilde{p}_i\|^2 + \lambda \iint \left(f_{xx}^2 + 2f_{xy} + f_{yy}^2\right) dx dy.$$

Then, the solution of this functional is of the form

$$f(x, y) = a_0 + a_1 x + a_2 y + \sum_{i=1}^{n} w_i U_i(\|P_i - (x, y)\|).$$

where $U(r) = r^2 \log r^2$, the coefficients $a_0, a_1, a_2, (w_i)_1^n$ can be found by solving a linear system (see [24]).

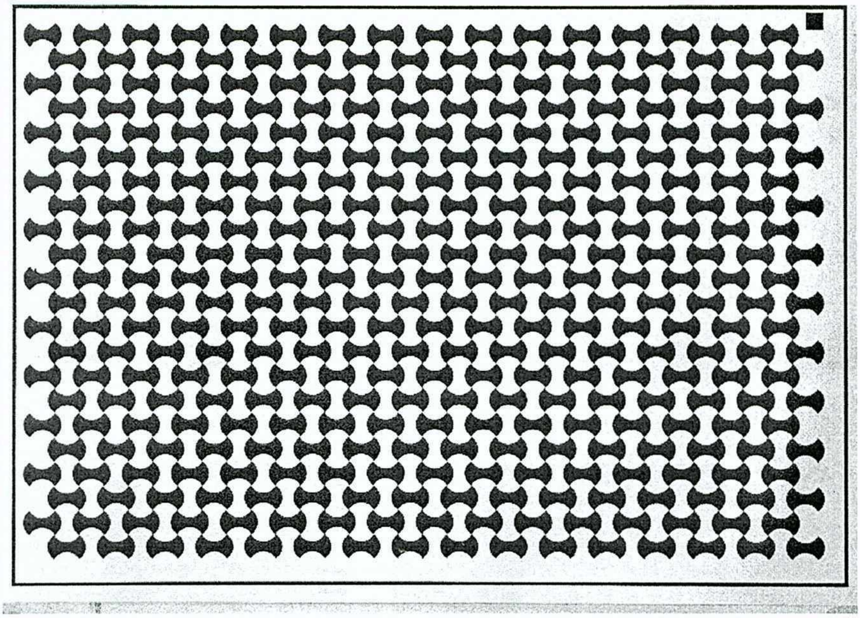
We utilized the Matlab Spline Toolbox™ for computing the thin-plate smoothing approximation.

4.3.3 Evaluation: Straight lines rectification

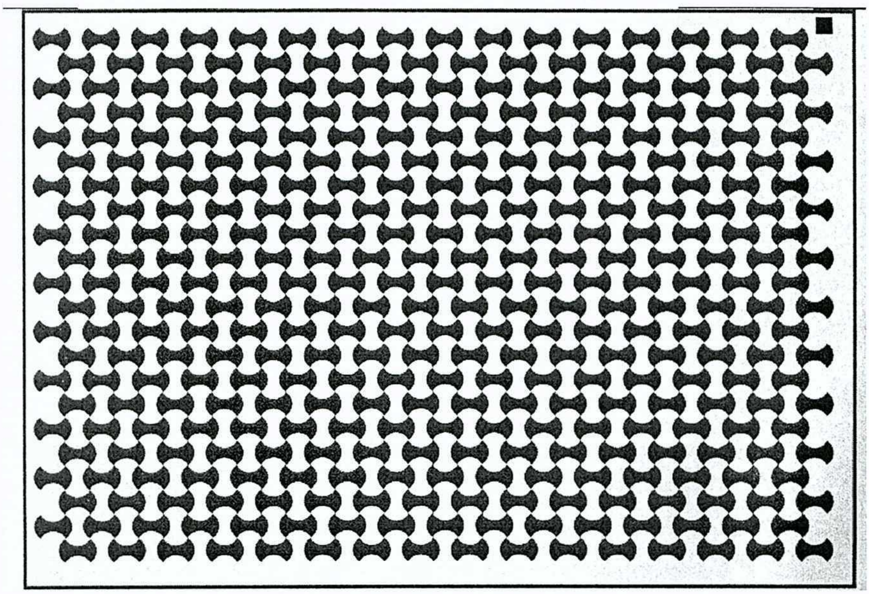
In order to evaluate both approaches, we added a rectangular frame to our pattern grid. Then we proceeded to correct the digital images by the two methods. If the transformation was correctly estimated then all sides of the rectangular frame should be rectified (corrected to be straight lines) and the rectangle should be parallel to the horizontal and vertical axes respectively.

In this experiment we used the blue channel of a raw image taken from a Pentax K-M camera using a 40mm focal length at f/5.6. Figures 11 and 12 show that in all cases the best correction is obtained by the thin-plates method.

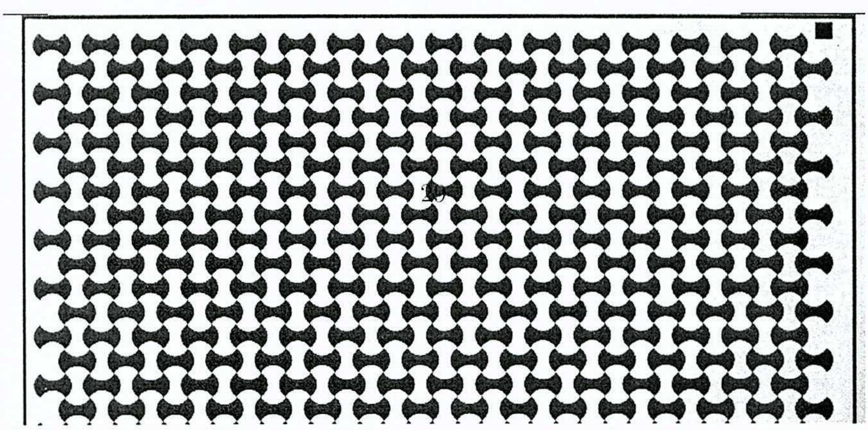
Although the distortion does not seem to be large, it is important to model it as better as possible to be close to the proposed image formation model. For that and also as we do not need to decompose the distortion, we decided to use the thin-plate approach.

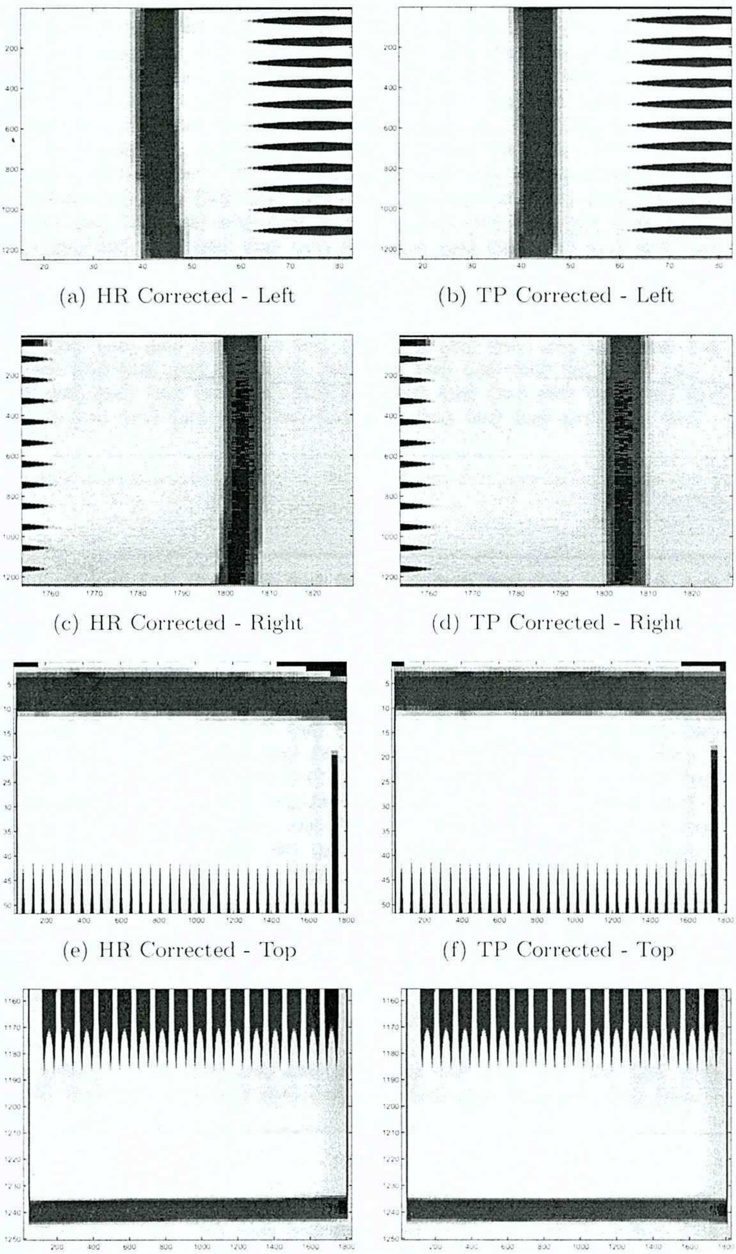


(a) Original



(b) HR Corrected





(a) HR Corrected - Left

(b) TP Corrected - Left

(c) HR Corrected - Right

(d) TP Corrected - Right

(e) HR Corrected - Top

(f) TP Corrected - Top

(g) HR Corrected - Bottom

(h) TP Corrected - Bottom

Figure 12: Distortion Correction as a validation procedure for geometric transformation estimation *II*. Homography+Radial correction (left side) and Thin-plates correction (right side).

4.4 Sharp image reconstruction

Once the blurred image is aligned to the grid pattern, only rests to estimate the *black* and *white* pixel values of the sharp pattern image. The estimation is performed locally, i.e. for every local grid pattern, in order to make it more robust and deal with non-uniform illumination. This can be formulated as estimating a and b for corresponding *black* and *white* pixel values, according to:

$$I_c(x,y) = (b-a)I(x,y) + a$$

where $I_c(x,y)$ is the adjusted contrast sharp local image pattern, and $I(x,y)$ is the normalized contrast local sharp image pattern with 0 for *black* pixel values and 1 for *white* ones.

If we suppose that the PSF support is not too large (in comparison to the size of the local grid pattern), which in our estimation of in-focus-no-movement case is a reasonable hypothesis, we can estimate the *black* and *white* values by taking the mean of pixels in *black* and *white* regions respectively. This is a direct consequence of the adopted model, suppose (x_c, y_c) is the central pixel of the *black* flat region, and W_{x_c,y_c} a window centered in pixel (x_c, y_c) and totally included in the *black* region. Then the estimation of a, $\hat{a}$ can be written:

$$\begin{aligned}
\hat{a} &= \mu(W_{x_c,y_c})^{-1} \sum_{(x,y)\in W_{x_c,y_c}} B(x,y) \\
&= \mu(W_{x_c,y_c})^{-1} \sum_{(x,y)\in W_{x_c,y_c}} \int_{(s,t)\in \mathrm{supp}(h)} h(s,t) I_c(x,y)\,dsdt \\
&= \mu(W_{x_c,y_c})^{-1} \sum_{(x,y)\in W_{x_c,y_c}} a \int h(s,t)\,dsdt \\
&= a \int h(s,t)\,dsdt.
\end{aligned}$$

For the sake of clarity in this analysis we have omitted the noise, sampling and distortion operators. Also, we have supposed that the size of W is small enough to keep constant the image value.

Then, if we assume that the PSF is normalized, i.e. $\int h(s,t)dsdt = 1$, assuming random noised of zero mean our estimator is unbiased (i.e. the expected value is equal the true value). In practice, this will not be strictly true and in Section 5 we show by simulations how a small error on the estimation of a and b will affect the PSF estimation by introducing some artifacts. As we have found that the precise estimation of this values are a critical step for an accurate PSF estimation, in Section 4.5.4 we propose a more robust way of estimating the PSF, without a precise knowledge of these values.

4.5 Local PSF Estimation

Once we have an estimate of the sharp image, we can solve the PSF estimation by solving an inverse problem. By using prior information about the smoothness of the PSF we can make the inverse problem well posed. This can be formulated by choosing k to minimize the functional

$$L = ||I * k - B||^2 + \lambda||\nabla k||^2, \tag{5}$$

where I is the estimated sharp image and B the blurred degraded observation. We have included a regularization term which penalizes kernels with large gradient. The regularization parameter λ is related to the noise level but also to how over/under determined is the system.

In order to be consistent with the acquisition process that we have adopted, we reformulate the problem described in Eq. 5 by considering the already introduced model (M_2) as:

$$k = \arg\min_k ||\Pi_{\mathbb{Z}^2} \left(g(R(I * k))\right) - B||^2 + \lambda||\nabla k||^2, \tag{6}$$

where we have included: $g(\cdot)$ the non-linear increasing function modeling the non-linearity response of the sensor, $R(\cdot)$ the geometrical transformation, and Π the sampling operator.

As we have already mentioned we cannot get rid of the non-linearity of the sensor response. However, if we work in middle range intensities then its response is almost linear. Due to our freedom to set up the laboratory experiment, we can choose the illumination conditions and the grid contrast value to work in the mentioned situation. So, from now on we will omit the function $g(\cdot)$ always remembering that we have to be careful in how we carry out the experiment.

We have two different ways of taking into account the geometrical transformation to solve the problem. We could directly correct the observation B, by applying the inverse transformation. However, as the observation is aliased we do not know how to correctly interpolate the image. Instead, we prefer to solve directly the problem as it is originally formulated. Although, this is possible to do as it is a convex functional, we prefer to write down a simplified linear version of the problem. By doing this, we can use fast algorithms to solve the problem. Suppose that we can approximate the geometrical distortion by an affine transformation by its first order Taylor expansion:

$$R(\mathbf{x} + \mathbf{h}) = R(\mathbf{x}) + J(\mathbf{x})\mathbf{h} + o(||\mathbf{h}||^2) \tag{7}$$

Then,

$$\begin{aligned}
R(I * h(\mathbf{x})) &= \int I(\mathbf{s}) h\left(R(\mathbf{x}) - \mathbf{s}\right) d\mathbf{s} \\
&= \int I\left(R\left(\mathbf{s}'\right)\right) h\left(R(\mathbf{x}) - R(\mathbf{s}')\right) |J| d\mathbf{s}' \\
&\approx \int I\left(R\left(\mathbf{s}'\right)\right) h\left(J(\mathbf{x})\left(\mathbf{x} - \mathbf{s}'\right)\right) |J| d\mathbf{s}' \\
&\approx \int I_R(\mathbf{s}') h_J(\mathbf{x} - \mathbf{s}') d\mathbf{s}' \\
&= I_R * h_J(\mathbf{x})
\end{aligned}$$

where we have applied Eq. 7 and a change of variable $\mathbf{s}' = R^{-1}(\mathbf{s})$. Observe, that by holding this assumption, we can modify the sharp image I by the geometrical distortion (to get I_R) and then solve for the kernel h_J. However, we have to take into account that the solution h_J is perturbed by the linear transformation J due to the jacobian of the distortion.

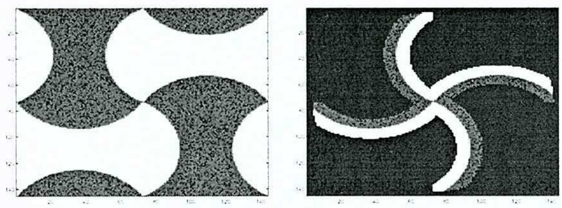
The assumption does not generally hold, due to the non-affine distortion introduced by the lens. Nevertheless, if we solve for local point spread functions, then locally the distortion can be well approximated by an affine transformation, and everything holds.

Note that if we assume the other image formation model (M_2) the PSF estimation can be done in the same way. However, the results follows directly from the model where we can write $I_R * h(\mathbf{x})$ so the kernel will not be modified by the geometrical distortion. This difference is due to the adopted model but conceptually in both cases we are estimating a point spread function that will characterize the digital camera.

Finally we can rewrite Eq. 6 by approximating the samples of the continuous convolution $I_R * h_J(\mathbf{x})$ by the discrete convolution. Note that this is true, if the sharp image and the kernel are sampled at frequency higher than the Nyquist frequency. In the case of the sharp image this cannot be true, as it has infinite support. Then

$$k = \arg\min_k \|\mathbf{MI}k - \mathbf{M}B\|^2 + \lambda\|\nabla k\|^2. \tag{8}$$

We have written the convolution as a linear operator by the convolution matrix $\mathbf{I}$ formed from I_R. The $\mathbf{M}$ operator is a mask that only evaluates the functional in those pixels that add information. It is not used to calculate the difference in every pixel, as the pixels deeply inside the flat regions do not incorporate any information apart from adding noise. In practice, the binary mask operator consist of a band of ones around the circular edges (see figure 13).



(a) (b)

Figure 13: Masking operator. Unmasked observation (a), masked observation (b).

4.5.1 Superresolved PSF

With the aim of computing a superresolved PSF, we can take advantage of the a priori analytic knowledge of the grid pattern. As we have studied in Section 4.2 we can align the grid pattern and the blurred observation with subpixel accuracy. Then, we can rasterize the grid pattern at the resolution we want to estimate the PSF. Let us call this rasterized image I_H. Next, we introduce a down-sampling operator $\mathbf{D}$, that takes a high resolution image I_H and generates I_L as $I_L(m,n) = I_H(sm, sn)$ where $s = 2^k$ ($k = 0, 1, ..$). Observe that we can replace the old data misfit term in Eq. 6 by $\|\mathbf{MDI}_H k_H - \mathbf{M}B\|^2$ and solve for k_H to find a superresolved PSF. Finally the minimization problem can be written as

$$(P) \qquad k = \arg\min_{k_H} \|\mathbf{MDI}_H k_H - \mathbf{M}B\|^2 + \lambda\|\nabla k_H\|^2$$

This formulation also improves the approximation of the continuous convolution as it is done in a higher resolution domain.

4.5.2 Choosing the regularization parameter λ

As we have already mentioned in the previous Section, the parameter λ should be an increasing function of the noise level but also on how much determined is the liner system due to the data misfit term. In a Bayesian interpretation, λ is the ratio of data and model variance.

In the original work [2] the λ parameter was fixed only taking into account the noise level and the size of the PSF support. In our opinion, the λ parameter should be at least a function of:

- How local we are doing the estimation (i.e. number of local grid patterns), this will have an impact in how well the observed step-edges response is *sampled*.

- How much we are increasing the resolution of the estimated PSF (i.e. s parameter).

- Noise level.

There exist two popular ways to give an estimation of the regularization parameter: the L-curve [25] and Generalized Cross Validation (GCV) [26]. The L-curve approach consists of selecting as the regularization parameter the point of largest curvature (called "L" point) in the plot Euclidean norm data misfit error vs. norm of regularization term.

In this work, we opted for GCV for determining λ. In Section 4.5.3, we will also explore a different way of posing the PSF estimation in a parameterless way. We leave for future investigations the performance of other λ estimators like the L-curve approach.

Generalized Cross Validation The argument behind Generalized Cross Validation is that if λ is a good choice, then the λ estimated from a part of the data should be a good estimation for regularizing the problem and doing a prediction for the other part of the unseen data. We used *leave-one-out cross validation* (LOOCV) which means to compute the model by removing one observation and then calculate the residual between the removed value and the value predicted by the model.

Consider the general regularized least squares problem, where A is a m by n matrix (m data values), and b is a m by 1 vector (observations), D is a n by n regularization matrix,

$$\arg\min_x ||Ax - b||^2 + \lambda||Dx||^2$$

The cross validation residual [26] can be written as,

$$\begin{aligned} V(\lambda) &= \frac{1}{n}\sum_{i=1}^{n}\left(b_i - \hat{b}_{(i)}\right)^2 \\ &= \frac{\frac{1}{n}||I - N(\lambda)b||^2}{\left[\frac{1}{n}\text{trace}(I - N(\lambda)\right]^2} \end{aligned}$$

where $N(\lambda) = (A^tA + n\lambda D^tD)^{-1}A^t$, b_i is the i-th observation and $\hat{b}_{(i)}$ is the predicted value by the model estimated with the other $(n-1)$ observations.

Finally we choose as the regularization parameter the λ^* minimum of function $V(\lambda)$.

4.5.3 Parameterless minimization - Noise estimation

In Section 4.5.2 we presented the PSF estimation problem as the minimization of a functional composed by a data misfit term plus a regularization penalty term. Although we commented a way for choosing the weight of the penalty term (by a generalized cross validation approach), here we present a different way to estimate the PSF that only involves the estimation of the noise level.

Suppose that we know that the image noise follows an additive zero mean Gaussian white noise model, whose variance σ^2 is known. Then we can recast the problem (P_0) as:

$$(P_\sigma) \qquad k = \arg\min_k \|\mathbf{R}k\|^2$$
$$\text{subject to}$$
$$\|\mathbf{MDI}k - \mathbf{M}B\|^2 \leq |\mathbf{M}|\sigma^2$$

Notice, that this problem selects the most regularized of the kernels from all the feasible solutions. We consider a feasible solution if it can explain the observation due to the adopted noise model.

For this we need to give an estimation of the noise level. Due to the pattern grid we used, this can be easily done as it has several constant intensity parts. If we suppose that changes in illumination are smooth, then we can proceed as follows. Take a small window in the center part of the *black* region (analogous for the white region) and compute the intensity variance inside the window. Then, the estimated variance will be close to the noise level per *black* pixel.

In practice, we found that image noise depends on the intensity value, and higher intensity regions (white regions) have more noise than lower ones. We solved this by doing an average of the noise level. As for generalizing this to solve for a non-local kernel by using K local grid patterns, we can rewrite (P_σ) as:

$$(P_{\sigma_G}) \qquad k = \arg\min_k \|\mathbf{R}k\|^2$$
$$\text{subject to}$$
$$\begin{aligned} \|\mathbf{M}_1\mathbf{DI}_1k - \mathbf{M}_1B_1\|^2 &\leq |\mathbf{M}_1|\sigma_1^2 \\ \|\mathbf{M}_2\mathbf{DI}_2k - \mathbf{M}_2B_2\|^2 &\leq |\mathbf{M}_2|\sigma_2^2 \\ &\vdots \\ \|\mathbf{M}_K\mathbf{DI}_Kk - \mathbf{M}_KB_K\|^2 &\leq |\mathbf{M}_K|\sigma_K^2 \end{aligned}$$

Section 5 shows some simulations and real camera results by using this approach to solve for a local PSF kernel.

4.5.4 Minimization without contrast estimation

In Section 4.4 we presented an approach for sharp image reconstruction. We mentioned that in order to estimate the PSF it was important to estimate the *black* and *white* intensity values of the sharp image pattern. Also, in Section 4.5.1 we discussed how from the sharp image and the observed blurred image, we can find a superresolved PSF that satisfies the adopted image formation model.

Here we present a different approach, in which the *black* and *white* sharp image values do not need to be estimated. Therefore, this novel method should be more robust than the previous one.

Consider the original problem presented in Eq. 5 (we have left out the regularization term for readability but it does not change anything), remembering $I_c = (b-a)I + a$, where I is the binary sharp image,

$$\begin{aligned}\min_k \|I_c * k - B\|^2 &= \min_k \min_{a,b} \|((b-a)I + a) * k - B\|^2 \\ &= \min_k \min_{a',b'} \|b'I * k - a' - B\|^2 \\ &= \min_k \min_{a'} \|I * k - a' - B\|^2\end{aligned}$$

Observe that we have included the scale term $(b-a)$ inside the kernel, so we can get the best kernel k upto a scale factor. This is not important, as we know that the kernel should be normalized. Next, we can apply

$$\min_{a'} \|I * k - a' - B\|^2 = \|I * k - B - \overline{I * k - B}\|^2$$

which is a direct consequence that the best estimator for a vector in a least squares sense is its mean value. Finally we have

$$\min_k \|I_c * k - B\|^2 = \min_k \|I * k - \overline{I * k} - (B - \overline{B})\|^2,$$

where we have substracted to each term its respective mean value. Then, we can rewrite the minimization problem (P) as:

$$(P_{C'}) \quad k = \arg\min_{k_H} \|\mathbf{MCDI}_H k_H - \mathbf{M}B_c\|^2 + \lambda\|\mathbf{R}k_H\|^2$$

where $\mathbf{C}$ is the centering operator defined as $\mathbf{C} = \mathbf{I} - \frac{1}{m \times n}\mathbf{E}$, $\mathbf{E}$ is a matrix of ones of size $m \times n$ by $m \times n$, $\mathbf{I}$ is the identity matrix of same size, and $B_c = B - \overline{B}$ are the centered observed values. Also, we can reformulate the non local kernel estimation problem in the same way.

In Section 5 we show some results obtained by solving this approach instead of the original one.

4.6 Numerical methods to estimate the PSF

In Section 3 we gave a list of hypotheses that the PSF should satisfy. One of them mentions that the kernel must be positive as negative light does not exist. This implies that our solution must be non negative, so we can impose this to shrink the set of possible solutions.

Suppose that the local grid pattern observation B is of size $m \times n$ and we want to estimate a PSF at $s\times$. Also suppose that the estimated support of the PSF is inside a $p \times q$ image. Then, the problem to be solved can be formally written as

$$(P_0) \quad k = \arg\min_{k_H} \|\mathbf{MDI}_H k_H - \mathbf{M}B\|^2 + \lambda\|\mathbf{R}k_H\|^2$$
$$\text{subject to} \quad k_{H_i} \geq 0$$

where $\mathbf{MDI}_H$ is a $m \times n$ by $p \times q$ matrix, which in practice can be too large. $\mathbf{R}$ is the matrix associated to the gradient operator.

If we want to estimate a PSF but taking into account K local grid patterns, then we would have to concatenate the associated $\mathbf{MDI}_{H_i}$ matrices and B_i observations for $i = 1, \ldots, K$. This would require a huge amount of memory. Let us call $\mathbf{P}_i = \mathbf{MDI}_{H_i}$ and $Q_i = MB_i$ for each local grid pattern. Then we can re-write the problem (P_0) as

$$(P_1) \quad k = \arg\min_{k_H} \left\| \left(\sum_i \rho_i \mathbf{P}_i^t \mathbf{P}_i + \mathbf{R}^t\mathbf{R} \right) k_H - \sum_i \rho_i \mathbf{P}_i^t Q_i \right\|^2$$
$$\text{subject to} \quad k_{H_i} \geq 0$$

Notice, that in this case all matrices are of the size pxq by pxq which is considerably smaller than the case before. We have substituted the parameter λ by a parameter ρ_i which performs the same (opposite) role but in a local way.

4.6.1 Studied Methods

With the aim of solving the problem (P_1) subject to the non-negativity constraint, we studied several different methods:

- A Gradient descent with projection method [27]
- Two-Point Step size gradient method with projection (based on [28])
- An Uzawa's Method [29]
- A Newton interior point method [30]

- Disciplined convex programming - Semidefinite-quadratic-linear programming implemented in CVX [31]

For a general survey of the development of algorithms for nonnegativity constraints see [32] and for more general convex optimization [33].

Although a simple gradient descent with projection manages to get the correct solution, its convergence time is really slow. We also studied the two-point step size gradient algorithm described in [28]. This algorithm was modified by projecting the actual point, at the end of each iteration, in order to satisfy the non-negativity constraint. However, we could not manage to get it to converge.

We also implemented an Uzawa's based method. Although this method achieved good results for finely tuned parameters, we found that the results were very sensitive to the step-size chosen.

The general framework CVX [31] in which all problems are rewritten in an standard way, also succeeded in getting a solution of the problem. This disciplined convex programming toolbox, uses solvers based on predictor-corrector variants of interior-point methods [33].

As the interior-point method described in [30] gets similar results to CVX, besides of being simpler and direct to implement we decided to use it to solve our problem (P_1).

A Newton interior point method described in [30] The goal is to recover a non-negative vector x which explains the observation b as well as possible in a least squares sense,

$$(P) \quad \arg\min_{x} \|Ax - b\|^2 \qquad \text{subject to} \quad x \geq 0$$

This problem, can be rewritten as a quadratic programming (QP),

$$(QP) \quad \arg\min_{x} \quad A^t Ax - 2A^t b \qquad \text{subject to} \quad x \geq 0$$

According to the Karush-Kuhn-Tucker optimality conditions, if the vector x is a minimizer for (P), there exists a vector y such that

$$(LCP) \quad y = A^t Ax - A^t b, \quad y \geq 0, x \geq o, x^t y = 0$$

This problem is called the linear complementarity problem (LCP) associated to the (P) problem.

The algorithm we used is based in a Newton approach to solve the LCP. Consider the following non-linear equation,

$$F(x, y) = \begin{bmatrix} XYe \\ A^t AXe - Ye - A^t b \end{bmatrix} = 0$$

where $X, Y \in \mathbb{R}^{n\times n}$ are square diagonal matrix whose diagonal elements are the components of x and y respectively, and $e \in \mathbb{R}^N$ is a vector of ones. The idea is to solve for function $F(x, y) = 0$ in the set $S = \{(x, y) : x \geq 0, y \geq 0\}$.

For that purpose, consider a sequence of points (x^k) by,

$$(x^{k+1}, y^{k+1}) = (x^k, y^k) + \theta_k(u^k, v^k)$$

where θ_k is a positive stepsize and (u_k, v_k) is the Newton descent direction given by the solution of,

$$\begin{bmatrix} Y^k & X^k \\ A^t A & -I \end{bmatrix} \begin{bmatrix} u^k \\ v^k \end{bmatrix} = \begin{bmatrix} -X^k Y^k e + \mu_k e \\ -A^t AX^k e + Y^k e + A^t b \end{bmatrix} \tag{9}$$

The value of μ_k must be positive to assure all the variables to be positive.

Let θ_k^* be the largest value of θ_k such that $(x^{k+1}, y^{k+1}) \in S$. In fact, we need $x^k + \theta_k u^k > 0$ and $y^k + \theta_k v^k > 0$, so $\theta_k^* = \min\{\theta_k^1, \theta_k^2\}$ where,

$$\theta_k^1 = \min\left\{ \frac{-u_i^k}{x_n^i} : i = 1, \dots, n \text{ and } u_i^k < 0 \right\}$$

$$\theta_k^2 = \min\left\{ \frac{-v_i^k}{y_n^i} : i = 1, \dots, n \text{ and } v_i^k < 0 \right\}.$$

Also, we want that $g(x, y) = x^t y$ decreases in each iteration. Then,

$$\begin{aligned} g(x^{k+1}, x^{k+1}) - g(x^k, x^k) &= (x^k + \theta_k u^k)^t (y^k + \theta_k v^k) - (x^k)^t y^k \\ &= ((u^k)^t v^k)\theta_k^2 + ((x^k)^t v^k + (y^k)^t u^k)\theta_k \\ &= ((u^k)^t v^k)\theta_k^2 + (-(x^k)^t y^k + n\mu_k)\theta_k \end{aligned}$$

where we want this expression to be negative. Two cases can be differentiated,

$$\begin{array}{lll} \text{if } (u^k)^t v^k \leq 0 \text{ then} & 0 < \theta_k \leq \theta_k^*, & 0 < \mu_k < \tilde{\mu}_k \\ \text{if } (u^k)^t v^k > 0 \text{ then} & 0 < \theta_k \leq \min\{\theta_k^*, \tilde{\theta_k}\}, & 0 < \mu_k < \tilde{\mu}_k \end{array}$$

where,

$$\tilde{\theta}_k = \frac{(x^k)^t y^k - n\mu_k}{(u^k)^t v^k}$$

$$\tilde{\mu}_k = \frac{(x^k)^t y^k}{n}.$$

In [30] the authors give some ideas, base on their computational experience on how to set the parameters θ_k and μ_k. Finally the Newton based algorithm can be summarized in the following steps:

1. Initialization

   - Choose: $tol1$ and $tol2$ two tolerances for zero, and $(x_0, y_0) > 0$ initial point.
   - Set $k = 0$.

2. Main Loop

   Step 1 Compute u^k and v^k given by Eq. 9.

   Step 2 Choose an appropriate θ_k by respecting all the conditions given and update (x^{k+1}, y^{k+1}).

   Step 3 If $(x^{k+1})^t y^{k+1} < tol1$ and $\|A^t A x^{k+1} - A^t b - y^{k+1}\|^2 < tol2$ stop and set $x^* = x^{k+1}$ as the solution of the given problem. Otherwise update $k = k + 1$ and return to step 1.

We do not investigate different numerical methods to solve the problem (P_σ). Instead we use the already introduced convex optimization toolbox CVX which can manage to solve this kind of quadratic constraint problems.

5 Experimental Results: psf estimation

5.1 Simulations for objective evaluation

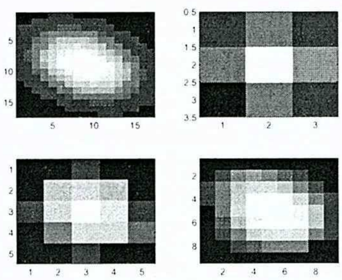
Considering that we do not know the PSF camera ground truth (i.e. the *real* camera PSF) we propose an evaluation method that includes different simulations trying to recreate the camera acquisition process. This way, the proposed subpixel PSF estimation methodology is evaluated using data simulated under different conditions and representing different outcomes of the acquisition process. We pay particular attention to the aliasing effect caused by sampling under the Nyquist frequency.

Manual λ, known real contrast

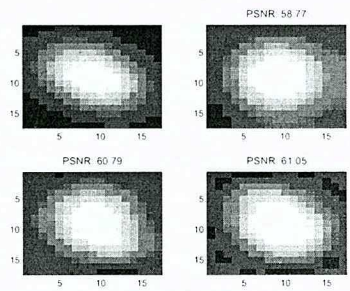
This experiment is probably the most important one. It validates the possibility of performing a subpixel PSF estimation with the proposed approach. For that purpose we rasterize the grid pattern at a high resolution, we convolve it with a PSF like kernel (not necessary anisotropic) and we down sample it to get the observed digital image. The image is down sampled at a rate $16\times$ (i.e. one pixel from a 16×16 block) and the kernel is chosen so that the low resolution image presents aliasing artifacts. We also add white Gaussian noise of s.d. $\sigma = 0.05$. We call this experiment the *base* test. We ran the basic algorithm manually setting the regularization parameter λ and the contrast values: a and b. The alignment was done automatically. We estimated $1\times$, $2\times$ and $4\times$ PSF.

The estimation results, performed with one local grid pattern and a global estimation using 81 local patterns, are presented in Figure 14. We interpolated (using a Lanczos window) the estimated kernels in order to compare them with the original one. As a performance measure we decided to use the PSNR between the interpolated kernel and the original one. Although a high PSNR could indicate a good estimation, it is important to notice that the kernel could still present artifacts. For that reason we also give a qualitative description of the estimation. In this example, as it is shown in Figure 14, none of the estimated kernels present strange artifacts. In both cases, the estimation done at $2\times$ or $4\times$, seem to capture the shape of the kernel (eccentricity at 45 degrees) while the estimation done at $1\times$ does not.

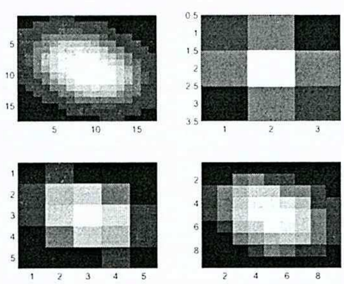
At the end of this section we present a table summarizing all the PSNR values and the observations for every of the following experiments.



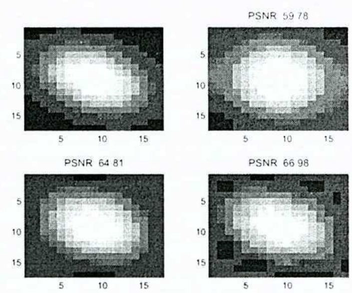
(a) original estimation - local



(b) interpolated estimation - local



(c) original estimation - global



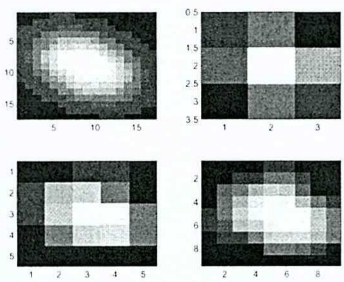
(d) interpolated estimation - global

Figure 14: Manual λ, known real contrast. For all estimations: original kernel (top left), $1\times$ estimation (top left), $2\times$ estimation (bottom left), $4\times$ estimation (bottom right). Local estimations are performed using only one local pattern; global ones using 81 local patterns.

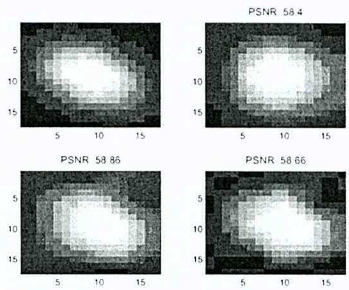
Manual λ, known real contrast, alignment perturbed 0.25 pixel

This experiment tries to validate the motivation to develop and include an accurate subpixel corner detector in the PSF estimation process. We repeated the *base* test but we perturbed the alignment done by the subpixel corner detector with a random variable uniformly distributed in $(-0.25, 0.25)$ pixels. Once again we ran the basic algorithm manually setting the the regularization parameter and contrast values. We estimated the $1\times$, $2\times$ and $4\times$ PSF.

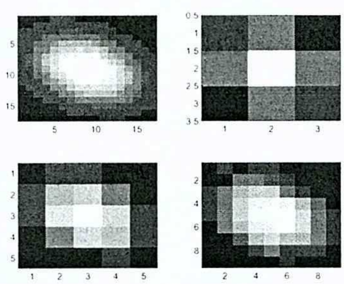
Results are shown in figure 15. As expected, the performance is lower than for the *base* test, in particular for the local estimations. Besides some artifacts appear in the global $4\times$ estimation.



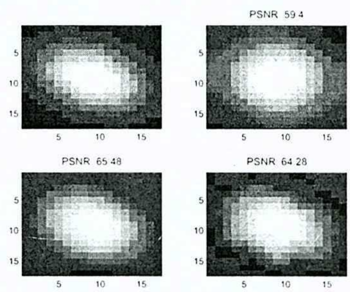
(a) original estimation - local



(b) interpolated estimation - local



(c) original estimation - global



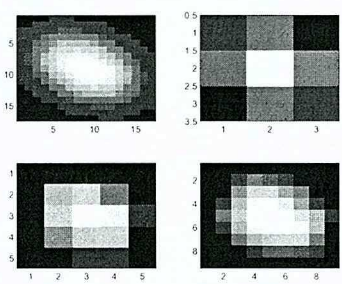
(d) interpolated estimation - global

Figure 15: Manual λ, known real contrast, alignment perturbed $U(-1/4, 1/4)$ pixel. For all estimations: original kernel (top left), $1\times$ estimation (top right), $2\times$ estimation (bottom left), $4\times$ estimation (right). Local estimations are performed using only one local pattern; global ones using 81 local patterns.

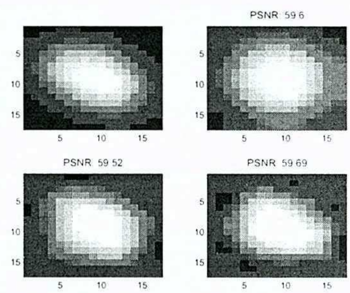
Manual λ, real contrast perturbed

This experiment shows the sensibility of the proposed method to the estimation of the printed pattern contrast level (black and white pixel values). We repeated the *base* test but we perturbed the real contrast values by a adding a random variable uniformily distributed in $(5, 5)$ (image values range $[0, 255]$). We ran the basic algorithm manually setting the regularization parameter. We estimated the $1\times$, $2\times$ and $4\times$ PSF.

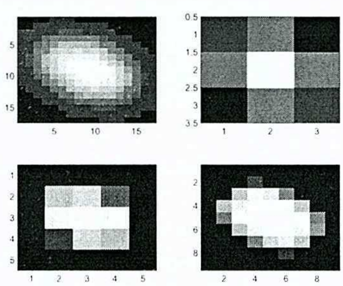
Results are shown in figure 16. As we expected the performance is lower than for the *base* test, in particular for the $2\times$ and $4\times$ estimations. We can also appreciate that the Gaussian shape is not correctly estimated in the local/global $4\times$ estimations.



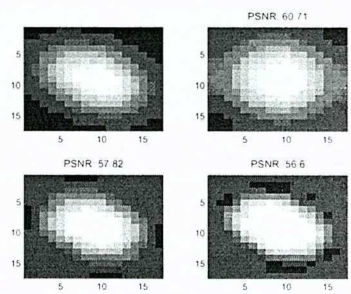
(a) original estimation - local



(b) interpolated estimation - local



(c) original estimation - global



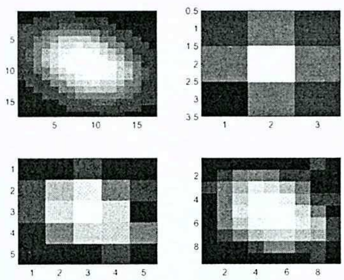
(d) interpolated estimation - global

Figure 16: Manual λ, real contrast perturbed. For all estimations: original kernel (top left), $1\times$ estimation (top right), $2\times$ estimation (bottom left), $4\times$ estimation (bottom right). Local estimations are performed using only one local pattern; global ones using 81 local patterns.

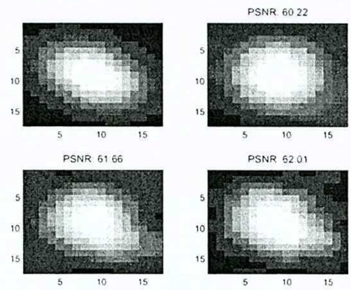
Manual λ, unknown real contrast

This experiment shows the stability of the PSF estimation by automatically finding the contrast levels (black and white pixel values). We repeat the *base* test without setting the contrast values. Again, we ran the basic algorithm manually setting the regularization parameter. We estimated the $1\times$, $2\times$ and $4\times$ PSF.

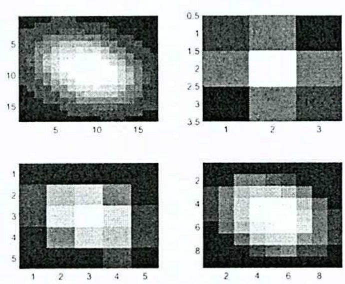
Results are shown in figure 17. The performance is similar to that of the *base* test.



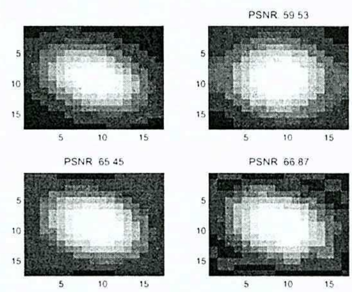
(a) original estimation - local



(b) interpolated estimation - local



(c) original estimation - global



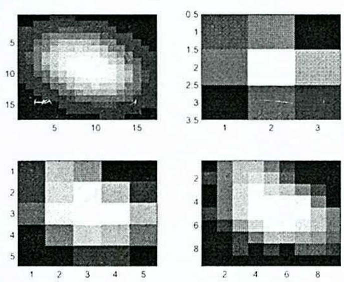
(d) interpolated estimation - global

Figure 17: Manual λ, unknown real contrast. For all estimations: original kernel (top left), $1\times$ estimation (top right), $2\times$ estimation (bottom left), $4\times$ estimation (bottom right). Local estimations are performed using only one local pattern; global ones using 81 local patterns.

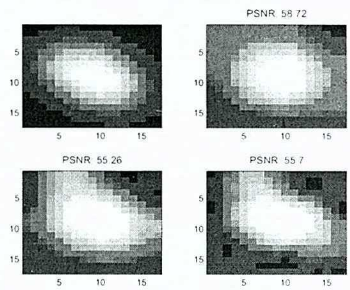
Manual λ, unknown real contrast, very noisy image

This experiment shows the robustness of the PSF estimation algorithm to additive Gaussian white noise. We repeat the *base* test without setting the contrast values but in this case we added white Gaussian noise of standard deviation $\sigma = 0.15$. We manually set the regularization parameter λ and estimate the $1\times$, $2\times$ and $4\times$ PSF.

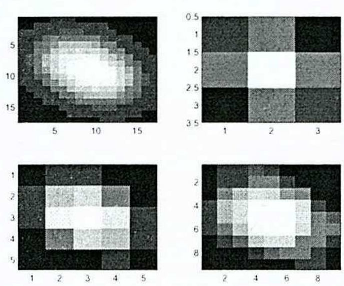
Results are shown in figure 18. As expected the performance is lower than for the *base* test. Besides some artifacts appear in the local/global $4\times$ estimation and the local $2\times$ estimation.



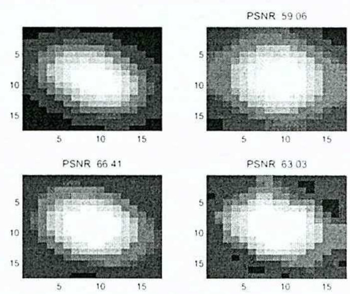
(a) original estimation - local



(b) interpolated estimation - local



(c) original estimation - global



(d) interpolated estimation - global

Figure 18: Manual λ, unknown real contrast, very noisy image. For all estimations: original kernel (top left), $1\times$ estimation (top right), $2\times$ estimation (bottom left), $4\times$ estimation (bottom right). Local estimations are performed using only one local pattern; global ones using 81 local patterns.

Performance Comparison Summary

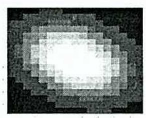
Experiment	Resolution	Mode	PSNR	Observations
known real contrast	×1	local	58.8	shape not captured
		global	61.1	shape not captured
	×2	local	60.9	-
		global	64.8	-
	×4	local	61.1	-
		global	67.0	-
alignment perturbed known real contrast	×1	local	58.4	shape not captured
		global	59.4	shape not captured
	×2	local	58.9	-
		global	65.5	-
	×4	local	58.7	some artifacts
		global	64.3	some artifacts
real contrast perturbed	×1	local	59.6	shape not captured
		global	60.7	shape not captured
	×2	local	59.5	-
		global	57.8	-
	×4	local	59.7	wrong shape
		global	56.6	wrong shape
unkwnown real contrast	×1	local	60.2	shape not captured
		global	59.5	shape not captured
	×2	local	61.7	-
		global	65.5	-
	×4	local	62.0	-
		global	66.9	some artifacts
unkwnown real contrast very noisy image	×1	local	58.7	shape not captured
		global	59.1	shape not captured
	×2	local	55.3	some artifacts
		global	66.4	-
	×4	local	55.7	some artifacts
		global	63.0	some artifacts

Table 1: Performance Comparison Summary

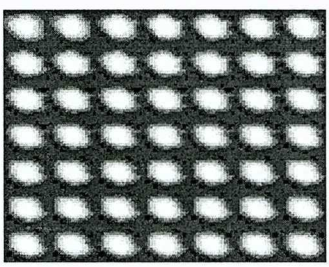
Changing the considered number of local grid patterns

This experiment shows how the PSF estimate varies when considering more local grid patterns in the estimation. We repeated the *base* test, this time increasing by one the number of considered local grid patterns when performing the PSF estimation. We manually set the regularization parameter λ.

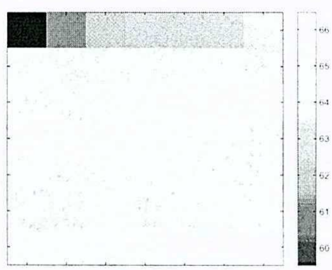
Results for the $4\times$ PSF estimation are shown in figure 19. As we expected the performance is better than for the *base* test as more local grid patterns are considered. However, this is a synthetic example and in real situations, where the PSF can spatially vary, we will have a trade off between averaging due to the spatial variation of the PSF and the accuracy of a local PSF estimation caused by considering less observations.



(a) original



(b) PSF at $4\times$

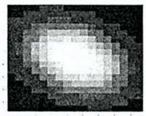


(c) PSNR

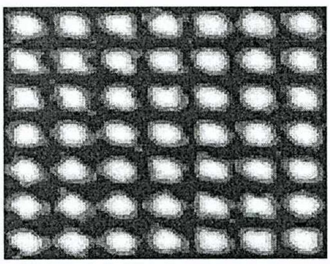
Figure 19: Changing the considered number of local grid patterns. (a) Original Kernel. (b) Estimated kernel at resolution $\times 4$ and interpolated to the original kernel size. From left to right top to bottom, increasing by one the number of considered local grid patterns used for the PSF estimation. (c) PSNR between each estimated PSF (interpolated by Lanczos) and the original kernel.

λ set by Generalized Crossvalidation

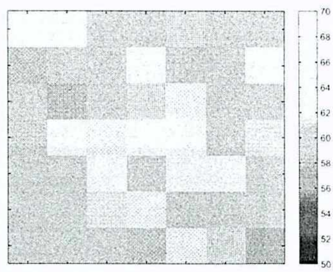
In this experiment we analyze the algorithm performance when fixing the regularization parameter λ by generalized cross validation (introduced in Section 4.5.2). We repeated the *base* test several times to evaluate the performance for different noise realizations. Results are shown in Figure 20. Unfortunately the performance is not always good, indicating that this method is not appropiate for choosing the regularization parameter.



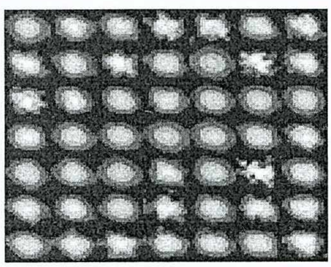
(a) original



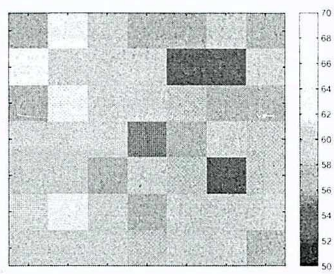
(b) estimation at $2\times$



(c) PSNR estimation at $2\times$



(d) estimation at $4\times$

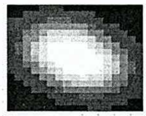


(e) PSNR estimation at $4\times$

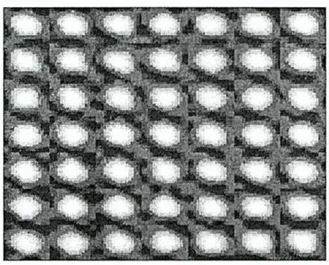
Figure 20: λ set by GCV. (a) Original Kernel, (b) estimated kernels at resolution $\times 2$ (interpolated) for several realizations and their respective PSNR (c). (d) estimated kernels at resolution $4\times$ (interpolated) for several realizations and their respective PSNR (e).

Minimization without contrast estimation

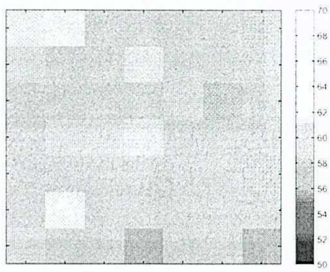
In this experiment we analyze the algorithm performance when solving for the alternative functional presented in Section 4.5.4. In this case, it is not necessary to do an accurate image contrast (*black* and *white* pixel values) estimation. We repeated the *base* test several times to see the performance for different noise realizations. Results are shown in Figure 21. The performance seems to be a little lower than for the optimal case.



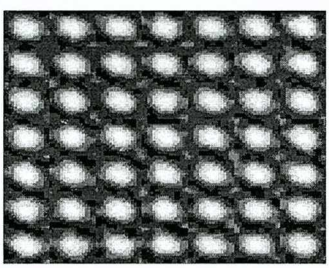
(a) original



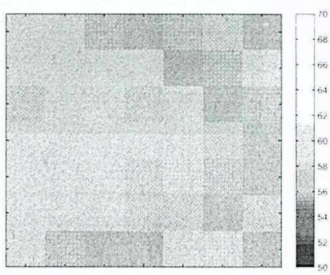
(b) estimation at $2\times$



(c) PSNR estimation at $2\times$



(d) estimation at $4\times$



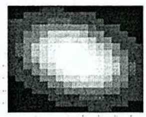
(e) PSNR estimation at $4\times$

Figure 21: Minimization without contrast estimation. (a) Original Kernel. (b) estimated kernels at resolution $\times 2$ (interpolated) for several realizations and their respective PSNR (c). (d) estimated kernels at resolution $4\times$ (interpolated) for several realizations and their respective PSNR (e).

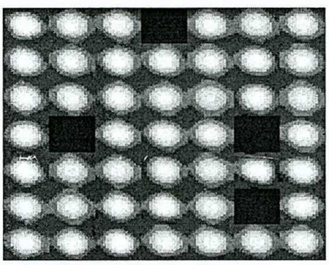
Minimization with noise level estimation

In this experiment we analyze the algorithm performance when solving for the alternative functional presented in Section 4.5.3. The idea is to find the most regular PSF that explains the observed noisy image. For that porpouse we need to estimate the noise level. As in this case we are considering white Gaussian noise, the noise level estimation is performed by estimating the gray level standard deviation in a flat region. We repeated the *base* test several times to evaluate the performance for different noise realizations. In order to get CVX to converge we had to increase the set of feasible solutions. For doing this we added a small value to the estimated noise standard deviation, letting the CVX toolbox solve the problem. This can be justified as considering other sources of error not taken into account during previous analysis. For example, when we presented the minimization problem (P), we approximated the convolution between the PSF and the sharp image as a low resolution discrete convolution. This approximation is one of the possible not previously considered error sources. In that sense, this procedure let us add other sources of error that are not considered in the noise level estimation previously done.

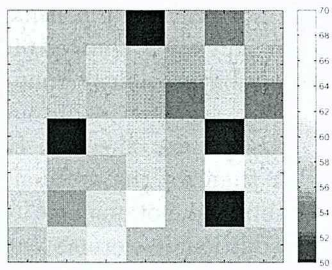
Results are shown in Figure 22. The performance seems to be a little lower than for the optimal case. Also, we show that in some cases the algorithm did not manage to find a feasible solution. The problem is that the additional tolerance, needed for the algorithm to converge, is an extra parameter that prevents the estimation from being parameterless.



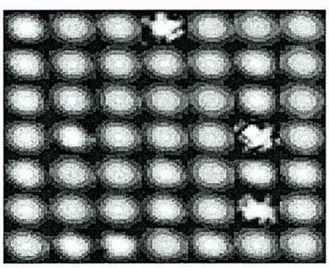
(a) original



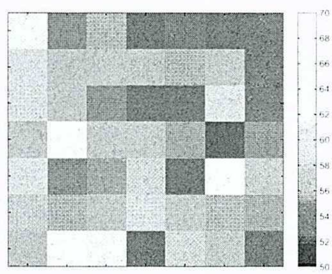
(b) estimation at 2×



(c) PSNR estimation at 4×



(d) estimation at 4×

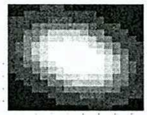


(e) PSNR estimation at 2×

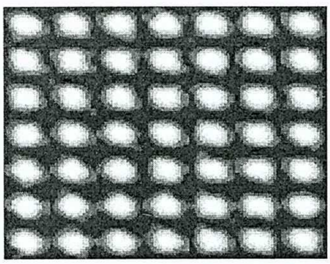
Figure 22: Minimization with noise level estimation. (a) Original Kernel. (b) estimated kernels at resolution ×2 (interpolated) for several realizations (b) and their respective PSNR (c). (d) estimated kernels at resolution 4× (interpolated) for several realizations and their respective PSNR (e).

Base test for different noise realizations

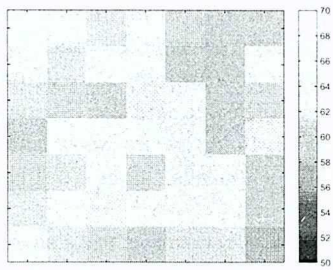
In this experiment we analyze the performance when solving the *base* test automatically estimating the contrast level. We repeated the test several times to evaluate the performance for different noise realizations. Results are shown in Figure 23. The algorithm seems to be stable to Gaussian noise as the performance has not seriously changed between noise realizations.



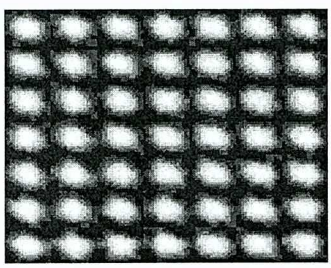
(a) original



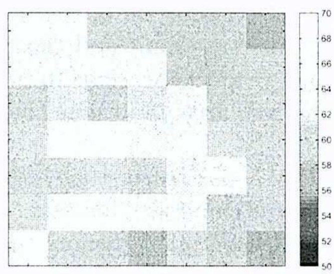
(b) estimation at $2\times$



(c) PSNR estimation at $2\times$



(d) estimation at $4\times$



(e) PSNR estimation at $4\times$

Figure 23: Experiment *base*. (a) Original Kernel. (b) estimated kernels at resolution $\times 2$ (interpolated) for several realizations (b) and their respective PSNR (c). (d) estimated kernels at resolution $4\times$ (interpolated) for several realizations and their respective PSNR (e).

5.2 Real camera images

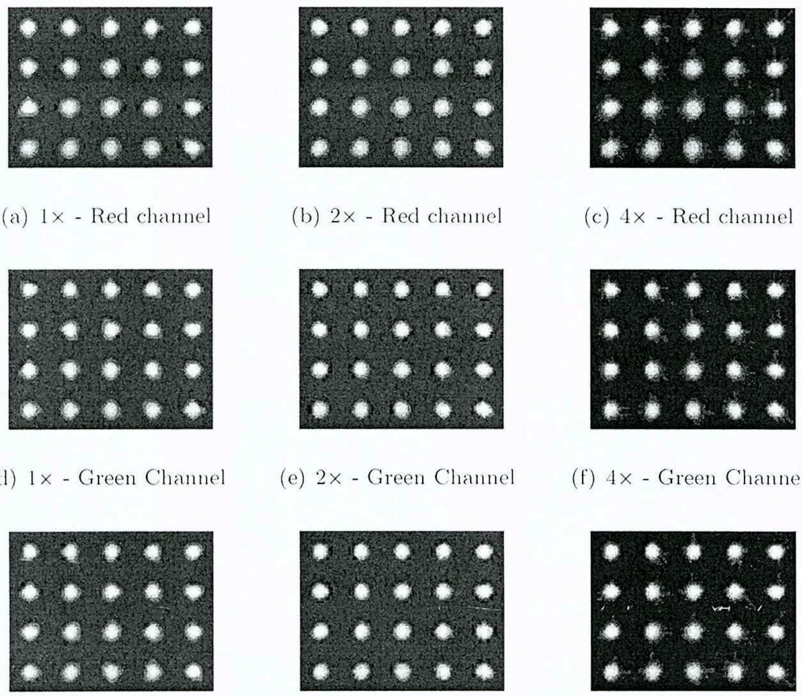
In this Section we present several spatially varying local PSF estimation examples using our basic algorithm. In all cases we have manually set $\lambda = 0, 6, 20$ for $1\times$, $2\times$, $4\times$ PSF estimation respectively.

Figures 24,25,26 show results for our pattern grid captured with a Canon EOS 40D camera provided with a Canon EF 50mm f/1.8 II lens at three different apertures: f/5.6, f/2.8, f/20.0 (at the same 50.0 mm focal length). Also in Figure 27 we show the results with a Pentax K-M camera at f/11.0, 40.0 mm. The estimated local PSF are of size 7×7 pixels and the super-resolved kernels $2\times$ and $4\times$ increase their size in the same proportion. In order to compare all estimations we have interpolated (by a Lanczos window) the $1\times$ and $2\times$ estimated kernels to be the same size as the $4\times$ kernel. In each figure we show the estimated local kernels for the red, green and blue channels taken from the raw image file.

The recovered PSFs show some interesting properties. First, for the apertures 5.6-20.0 distortion does not seem to be very significative while for the aperture 2.8 the PSFs estimated at the image border appears to be distorted with respect to the estimated at the center of the image plane. This could be a consequence of the fact that at 2.8 the lens is practically working at its aperture limit and thus distortion could be more important. This is the only case where seems to be chromatic aberration, as the estimated PSFs for the red, green and blue channel differ a little from each other.

We also notice that PSFs at f/5.6 are smaller than PSFs at f/20 with the Canon camera. This seems to be a direct result of the already described diffraction phenomenon: the radius of the PSF (airy pattern) increases with the f-number. The estimated kernels for the Pentax K-M camera at f/11 are larger than the Canon f/5.6 or f/20.0 PSFs. This apparently indicates that the Pentax K-M is of lower quality.

Finally we show what happens when the captured image is out-of-focus. In this case, as it is expected, PSFs are much larger than the in-focus case and a kind of donut effect appears. These artifacts are not well explained by the typical out-of-focus model.

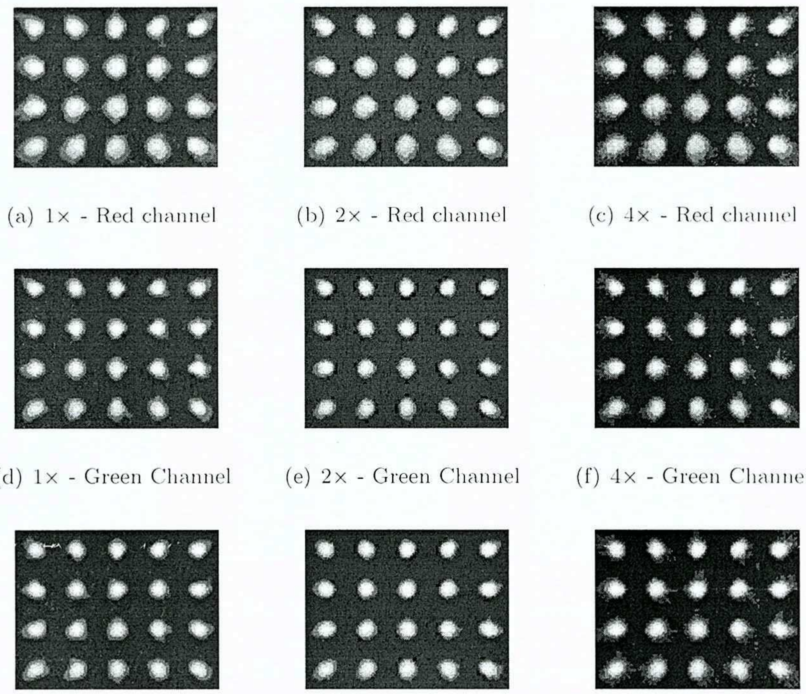


(a) 1× - Red channel (b) 2× - Red channel (c) 4× - Red channel

(d) 1× - Green Channel (e) 2× - Green Channel (f) 4× - Green Channel

(g) 1× - Blue channel (h) 2× - Blue channel (i) 4× - Blue channel

Figure 24: Real camera example. Canon EOS 40D with a lens Canon EF 50mm f/1.8 II. Taken at f/5.6, 1/5 s, 100 ISO, 50mm.

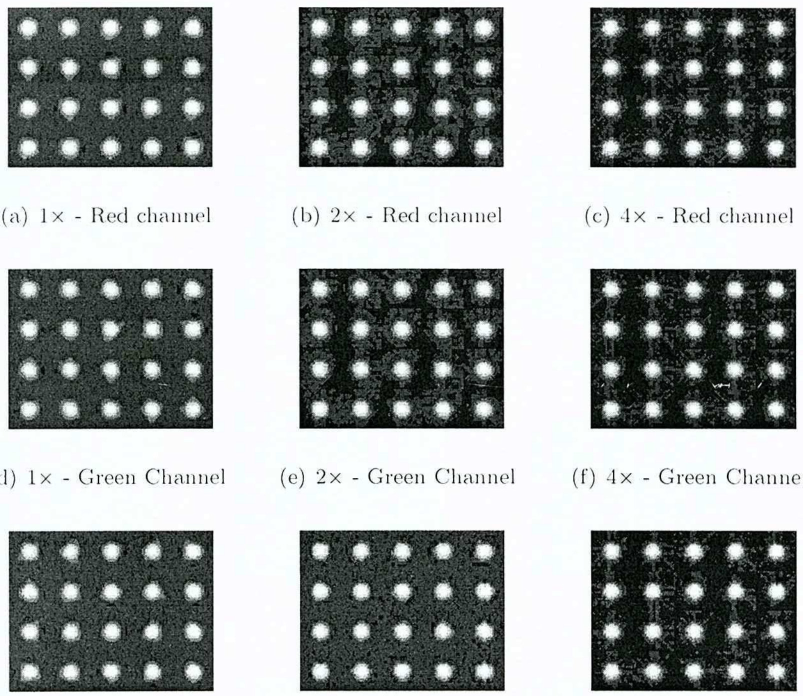


(a) 1× - Red channel (b) 2× - Red channel (c) 4× - Red channel

(d) 1× - Green Channel (e) 2× - Green Channel (f) 4× - Green Channel

(g) 1× - Blue channel (h) 2× - Blue channel (i) 4× - Blue channel

Figure 25: Real camera example. Canon EOS 40D with a lens Canon EF 50mm f/1.8 II. Taken at f/2.8, 1/20 s, 100 ISO, 50mm.

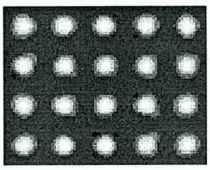


(a) 1× - Red channel (b) 2× - Red channel (c) 4× - Red channel

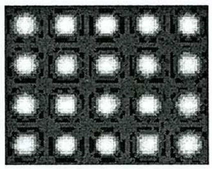
(d) 1× - Green Channel (e) 2× - Green Channel (f) 4× - Green Channel

(g) 1× - Blue channel (h) 2× - Blue channel (i) 4× - Blue channel

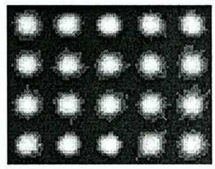
Figure 26: Real camera example. Canon EOS 40D with a lens Canon EF 50mm f/1.8 II. Taken at f/20.0, 3 s, 100 ISO, 50mm.



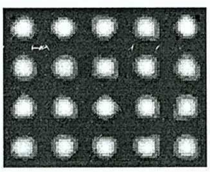
(a) 1× - Red channel



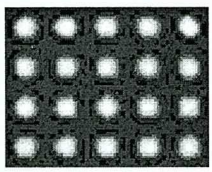
(b) 2× - Red channel



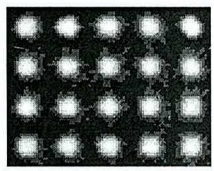
(c) 4× - Red channel



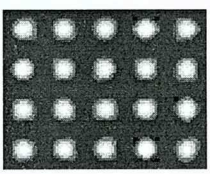
(d) 1× - Green Channel



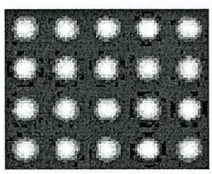
(e) 2× - Green Channel



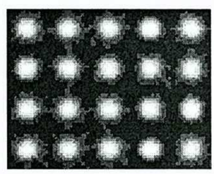
(f) 4× - Green Channel



(g) 1× - Blue channel

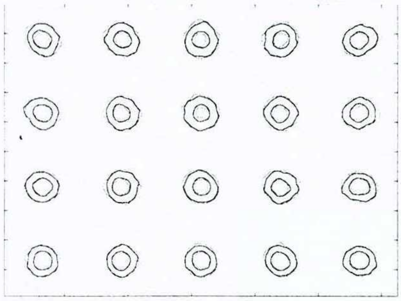


(h) 2× - Blue channel

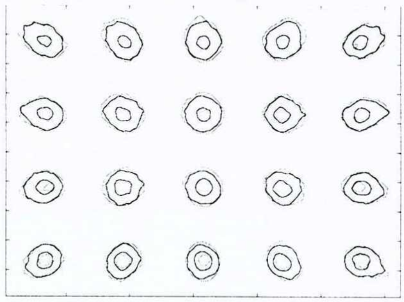


(i) 4× - Blue channel

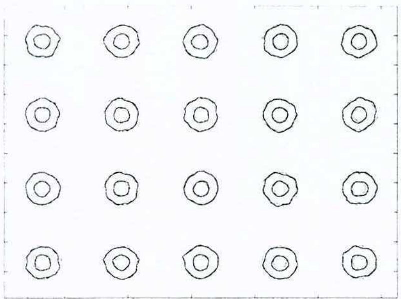
Figure 27: Real camera example. Pentax K-M. Taken at f/11.0, 4 s, 100 ISO, 40mm.



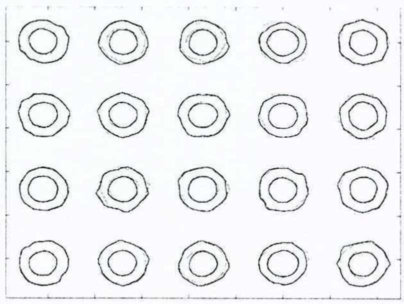
(a) Canon f/5.6, 50mm



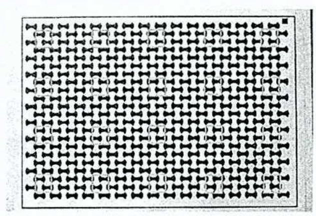
(b) Canon f/2.8, 50mm



(c) Canon f/20, 50mm

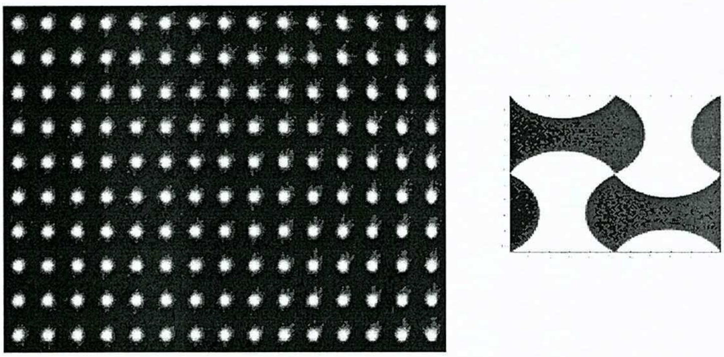


(d) Pentax f/11, 40mm

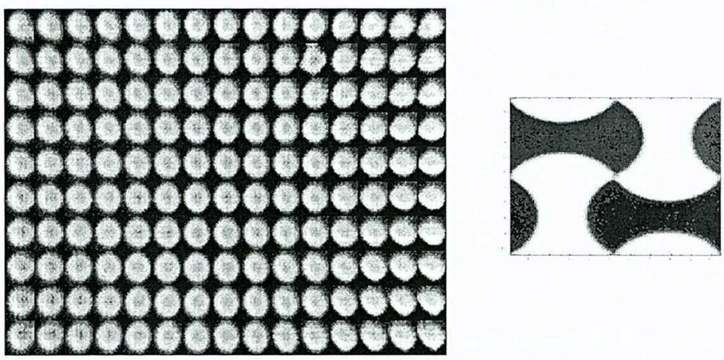


(e) local PSF positions

Figure 28: Real camera example. PSF contour lines at v=0.003 and v=0.013 for red, green (v1) and blue raw data. Canon EOS 40D with a lens Canon EF 50mm f/1.8 II taken at f/5.6 f/2.8 and f/20 with 1/5 s, 1/20 s, 3 s, 100 ISO, 50mm (a), (b) and (c) respectively. Pentax K-M taken at f/11.0, 4 s, 100 ISO, 40mm (d). Sample acquired image with marks where the local PSF is estimated (e).



(a) In focus - local PSFs and one local grid pattern image



(b) Out of focus - local PSFs and one local grid pattern image

Figure 29: Out of focus real camera example. Images taken with a Pentax K-M camera at f/5.6, 1 s, 100 ISO, 40mm. Local $4\times$ PSF s estimated by manually setting the focus distance to be in focus (a) and out of focus (b).

6 PSF Validation through single image superresolution

Having an accurate subpixel PSF estimation is critical to evaluate the properties of a digital optical system. It is also important to develop superresolution algorithms for producing better looking images. A more accurate estimation should improve results for superresolution algorithms using it as input. Based on this idea, we indirectly evaluate our PSF estimation by evaluating the quality of deconvolved/superresolved images acquired with a camera configured equally as for estimating the PSF.

Here we do not pretend to develop a state-of-the-art single image superresolution algorithm but to show it as a feasible application for our PSF estimation. Our PSF estimation algorithm computes a PSF at $\times 2$ and $\times 4$ the digital image resolution. In order to tackle the superresolution/deconvolution problem we can first interpolate the blurry observed values to get a superresolved blurry observation and next apply an image deconvolution algorithm to get the latent superresolved/deconvolved image. This is done in [34] where the authors proposed a blind superresolution algorithm.

This approach has an important drawback. If the image is aliased we do not know how to exactly interpolate it to get a superresolved blurry image, thus we will force the deconvolution algorithm to adjust data which is not the real observation. On the other hand, if the image is not aliased this method is optimal as the problem can be correctly decomposed into two separated steps: superresolution and deconvolution.

In order to avoid this situation, we propose a different approach which includes an image model based on natural images statistics. Both, superresolution and deconvolution problems, are faced simultaneously. No explicit interpolation method is imposed, on the contrary the image model acts as an implicit interpolator. This is presented in Section 6.2. Section 6.1 introduces the principal deconvolution techniques used for the superresolution problem in the case of interpolated blurred images.

6.1 Brief review of main image deconvolution methods

Lucy-Richardson

The Lucy-Richardson iterative algorithm [35, 36] tries to recover the latent image that has been blurred by a known PSF. The basic idea is to calculate the most likely latent image given the blurry noisy observation and the PSF. This maximum-likelihood formulation results in a fixed-point iteration:

$$i_{n+1} = i_n \left[k * \frac{b}{i_n * k} \right]$$

where i is the latent image, b the blurry observation and k the known PSF.

Lucy-Richardson assumes Poisson multiplicative noise, which is not very well suited to the common digital photographic images (for example see [37] where a physical model for the charged-coupled device CCD is proposed). This often generates unwanted artifacts such as ringing.

Wiener deconvolution

Wiener image deconvolution [38] is a direct application of the Wiener filter [39] whose goal is to reduce the amount of noise present on a signal.

Its formulation is obtained in the frequency domain using knowledge of the characteristics of the additive noise and the signal to be recovered:

$$\hat{I}(u,v) = \left[\frac{K^*(u,v)}{|K(u,v)|^2 + \gamma S_n(u,v)/S_i(u,v)}\right]$$

where $\hat{I}(u,v)$ and $K(u,v)$ are the Fourier transforms of the latent image and the PSF respectively, $S_n(u,v)$ and $S_i(u,v)$ are the power spectral density functions of the latent input image and noise.

It can be interpreted as an inverse filter acting only on those frequencies where the signal to noise ratio is significant:

$$\hat{I}(u,v) = \frac{1}{K(u,v)}\left[\frac{|K(u,v)|^2}{|K(u,v)|^2 + \gamma\frac{1}{SNR(u,v)}}\right].$$

Supposing no *a-priori* information of the signal, we can reformulate the general approach as

$$\hat{I}(u,v) = \frac{1}{K(u,v)}\left[\frac{|K(u,v)|^2}{|K(u,v)|^2 + \epsilon^2}\right].$$

One typical problem of the Wiener approach is how to choose the optimal ϵ, which is related to the signal to noise ratio.

6.1.1 MAP

The idea of the maximum a posteriori (MAP) framework is to find the optimal image that maximizes the posterior probability:

$$i^* = \arg\max_i p(i|b) \tag{10}$$

From the Bayes theorem $p(i|b) = \frac{p(b|i)p(i)}{p(b)}$. As maxima (minima) are unaffected by monotone transformations, for simplicity we take minus logarithm of Eq. 10 and solve for

$$i^* = \arg\min_i \left\{ \underbrace{-\log(p(b|i)}_{\text{data}} \quad \underbrace{-\log(p(i))}_{\text{prior}} \right\} \tag{11}$$

If we assume additive white Gaussian noise of zero-mean and σ^2 power, then

$$p(b|i) = Ke^{-\frac{\|b-i*k\|^2}{2\sigma^2}}$$

Thus

$$i^* = \arg\min_i \frac{\|b - i * k\|^2}{2\sigma^2} - \log(p(i)).$$

There is no general image model to choose an appropriate prior image probability distribution $p(i)$. One of the simplest models considers a zero-mean Gaussian distribution on the image gradient:

$$p(i) = Ke^{-\frac{\|\nabla i\|^2}{\sigma_i^2}}$$

Then, the problem can be formulated as a minimisation of,

$$E(i) = \|b - i * k\|^2 + \lambda \|\nabla i\|^2$$

This formulation penalizes image gradient in a ℓ_2-norm sense. This problem can be easily solved by least mean squares as both terms are ℓ_2 norms.

Novel studies in natural image statistics [7, 13, 40, 41] inspired other models for the prior distribution of the latent image. In particular, the work of A. Levin et al. [7, 13] is a significant improvement within this method. They replaced the ℓ_2 norm on the image gradient, which tends to enforce a Gaussian distribution and thus to equally distribute derivatives over the image. Instead, they assume that images are piecewise smooth and thus the image gradient distribution is zero-peaked with high kurtosis. To enforce this property, authors use a gradient sparse prior assumption which tends to enforce the expected edge content of "natural" images. This assumption also helps to remove unwanted image artifacts such as ringing. The image prior is written in a general form as,

$$p(i) = e^{-\alpha \sum_i \rho(g_i * h)}$$

where ρ is the sparse and heavy tailed function $\rho(x) = |x|^p$ with $p = 0.8$. In the simplest approach, g_i are the horizontal and vertical derivative filters $g_1 = [1, -1]$ and $g_2 = [1, -1]^T$. This leads to the minimization of the following functional,

$$E(i) = \|b - i * k\|^2 + \lambda \sum_i \rho(g_i * i) \tag{12}$$

Notice that the ℓ_2-norm on the regularization term has been replaced with a ℓ_p norm with $p = 0.8$. There exists no optimization algorithm that guarantees, in general conditions, to reach the global minimum of Eq. 12 as $E(i)$ is non-convex for $p < 1$. In [13] the authors proposed to use an iterative re-weighted least squares algorithm (IRLS) to solve this problem. Its effectiveness has been demonstrated in [13, 7, 34] where the authors managed to get the algorithm to converge to the apparently right minimum of $E(i)$.

6.2 Proposed approach

Our formulation is based on the MAP framework used in [13, 34] already presented. The idea is to incorporate a down-sampling operator to the data fitting term in order to adjust only to the observed data, as we did in Section 4.5.1 for computing a superresolved PSF. The down-sampling operator $\mathbf{D}$ takes a high resolution image I_H and generates I_L as $I_L(m, n) = I_H(sm, sn)$ where $s = 2^k$ $(k = 0, 1, ..)$. Then we can rewrite the MAP problem presented in Eq. 12 as

$$(P_S) \qquad i^* = \arg\min_i \|b - D(i * k)\|^2 + \lambda \sum_i \rho(g_i * i).$$

Apart from the horizontal and vertical derivative filters $g_1 = [1, -1]$ and $g_2 = [1, -1]^T$, second order derivative filters $g_3 = [1, -2, 1]$,$g_4 = [1, -2, 1]^T$ and $g_5 = [1 - 1; 1 - 1]$ are incorporated. This enforces the resultant image to have both first and second order sparse derivatives.

The main difference between our approach and the work of W. Zhang and W. Cham [34], who proposed a blind single image superresolution method, is that we do not impose any explicit interpolation method to get the superresolved blurry observation. We leave the image model act to get the latent image. This seems to be more reasonable in the case where the observed image b is aliased, and thus no explicit interpolation scheme should be exact.

Finally to solve (P_S) an IRLS algorithm is used. This approach reduces the problem to the solution of a sequence of weighted least squares problem. The main idea is to replace the ℓ_p objective term by a weighted ℓ_2 norm, where the weights are computed from the previous iteration.

$$\|\mathbf{x}\|_p^p = \sum_i |x_i|^p = \sum_i |x_i|^2 |x_i|^{p-2} = \sum_i |x_i|^2 w_i^2 = \|Wx\|_2^2$$

where $W = \operatorname{diag}(|x_i|^{\frac{p-2}{2}})$.

For more information on this algorithm we refer to [42, 13].

7 Experimental Results: Super-resolution

7.1 Simulations for objective evaluation

In order to objectively evaluate the super-resolution/deconvolution algorithms, independently of our PSF estimation, we set up the following experiment with simulated data. A high resolution (HR) known image is convolved with a kernel (simulating the PSF). The filtered image is downsampled x2 to get the observed pixels. We then test the ability of each algorithm to recover the high resolution image from the degraded down-samples and the high resolution kernel. All algorithm parameters are set to get the best results.

As a performance measure we used the PSNR between the restored superresolved image and the original HR one. Although, as we have already indicated, a high PSNR could indicate good performance, it is important to notice that the zoomed image could still present artifacts. For that reason we also give a qualitative description of the estimation.

The studied algorithms for deconvolution/superresolution are the following:

`LR` Lucy-Richardson deconvolution with previous cubic-spline zoom.

`Wiener` Wiener Filter deconvolution with previous cubic-spline zoom.

`L2-L2` MAP with L_2 norm for data fitting and regularization terms with previous cubic-spline zoom.

`L2-L08` MAP with L_2 norm and $L_{0.8}$ for data fitting and regularization terms respectively, with previous cubic-spline zoom.

`L2-L2-D` MAP with L_2 norm for data fitting and regularization term , considering the downsampling operator in data fitting term (no explicit interpolation).

`L2-L08-D` MAP with L_2 norm and $L_{0.8}$ for data fitting and regularization term respectively, considering the down-sampling operator in data fitting term (no explicit interpolation).

Isotropic Gaussian Kernel

In the first experiment a small isotropic Gaussian kernel is used as PSF. Figures 30-38 show the $\times 2$ restored images, difference images and modulus of spectrum difference for commonly used images: *lena, barbara, boat, peppers, pattern*.

Table 7.1 summarizes the PSNR[1] values for every experiment.

For this experiment we find in general similar performance indicators for all the algorithms given a certain image. If the image has high frequency components (as in *barbara* or *boat* images) the restored versions will be of significant lower quality. For images strongly following a prior model, superior results are obtained with algorithms including this prior information. This is the case of *peppers* and *pattern* images where as image gradient is sparse then L2-L08-D algorithm is more consistent.

If the image has no high frequency components the deconvolution algorithms that use cubic spline interpolation have good results. This is a direct consequence of the precise interpolation (e.g. *lena* image).

The norm of the difference image spectrum (original high-resolution - restoration) shows that the algorithm that considers the downsampling operator provides better estimates for the low-frequency components (Figures 37-38). This may indicate that the downsampling operator is acting as a de-aliasing process that in most cases seems to be more appropriate than the interpolation/deconvolution methods.

Figures 39-45 show the results for different values of the Gaussian kernel variance. For small values of the variance the downsampled image will be very aliased and reconstruction will be very difficult. On the other hand, if the kernel is very large, accurate deconvolution will be hard as much information has been lost during the downsampling process. Finally, the best scenario seems to be a medium size kernel, which gives a good compromise between aliasing and accurate deconvolution.

Image	LR	Wiener	L2-L2	L2-L08	L2-L2-D	L2-L08-D
lena	37.46	36.75	37.12	37.21	37.71	37.76
barbara	22.74	23.03	22.86	22.87	22.43	22.58
boat	28.56	28.3	28.42	28.38	28.63	28.64
peppers	31.76	31.49	31.65	31.94	31.93	32.61
pattern	22.32	21.52	21.65	22.77	21.81	23.0

Table 2: Isotropic Gaussian Kernel: Performance Comparison Summary.

[1]All PSNR values were computed in the center region in order to avoid edge effects.

Figure 30: Isotropic Gaussian Kernel: peppers image x2 superresolution/deconvolution results.

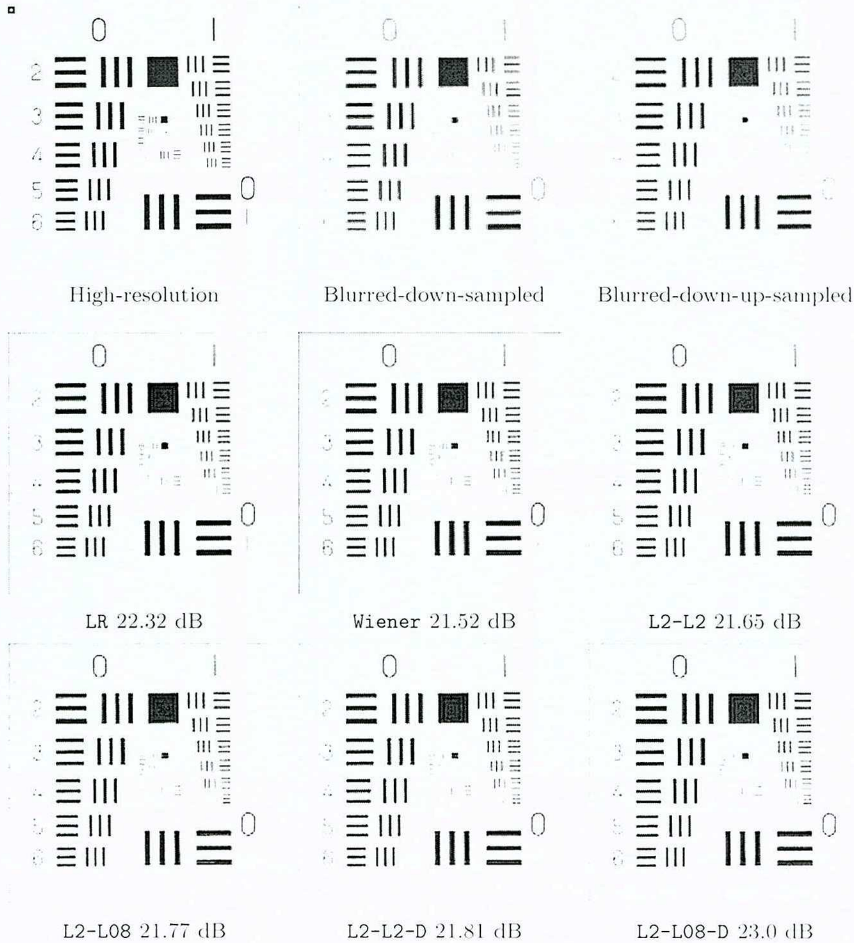


Figure 31: Isotropic Gaussian Kernel: mira image x2 superresolution/deconvolution results.

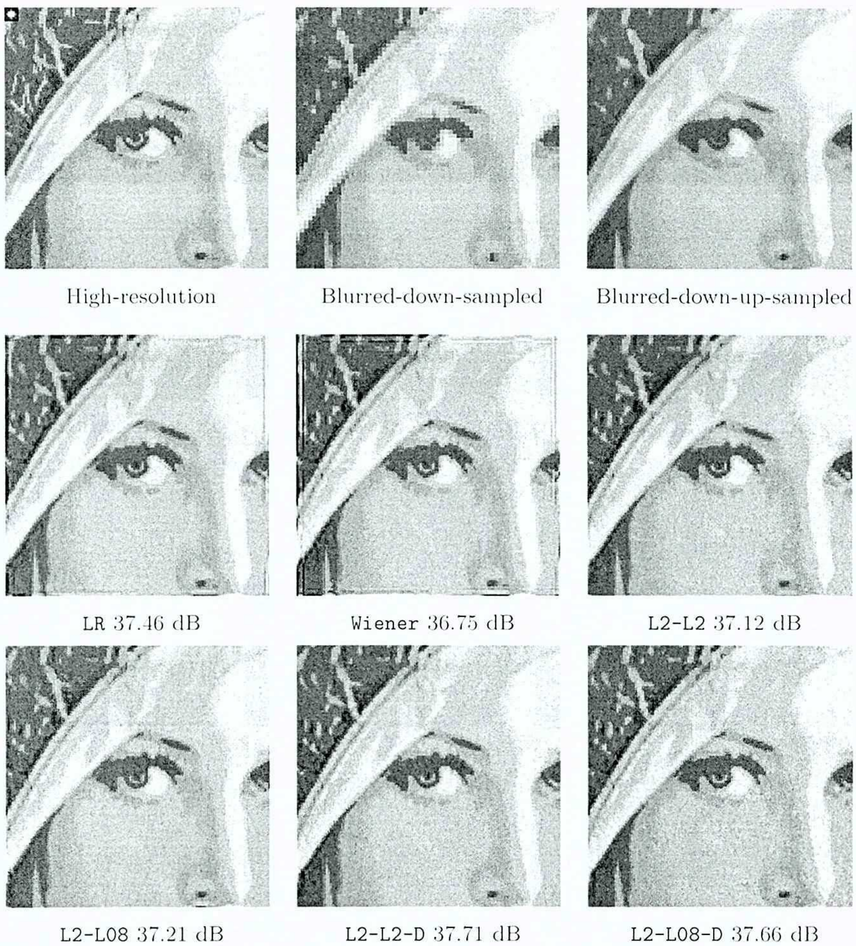


Figure 32: Isotropic Gaussian Kernel: lena image x2 superresolution/deconvolution results.

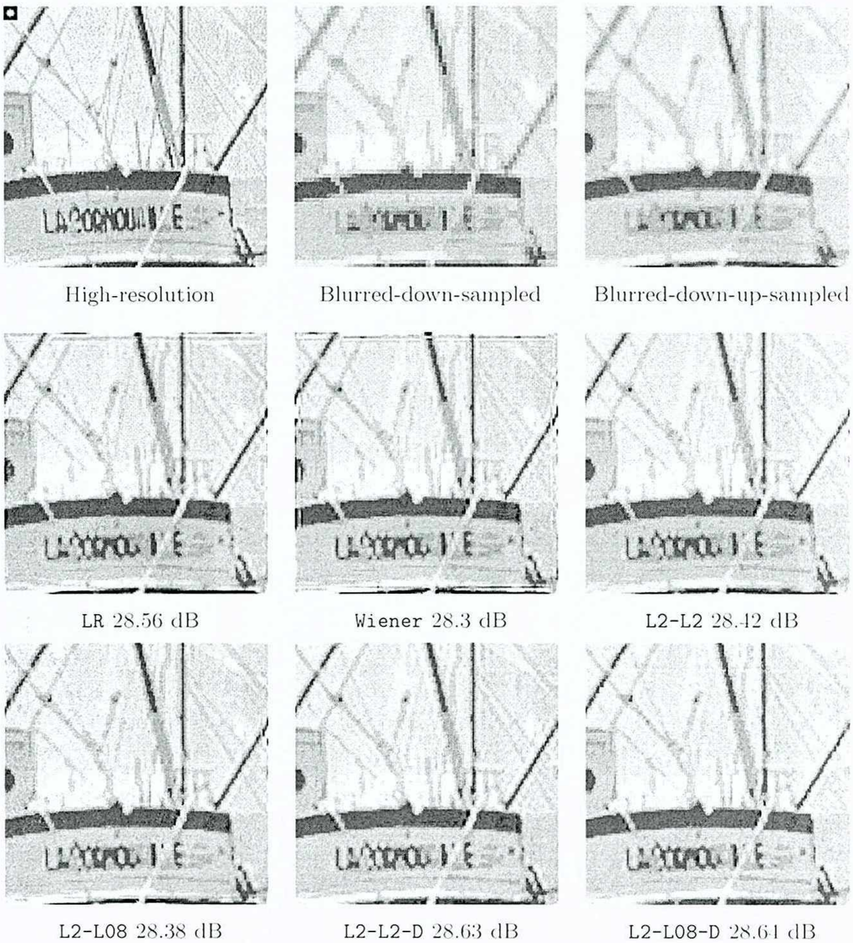
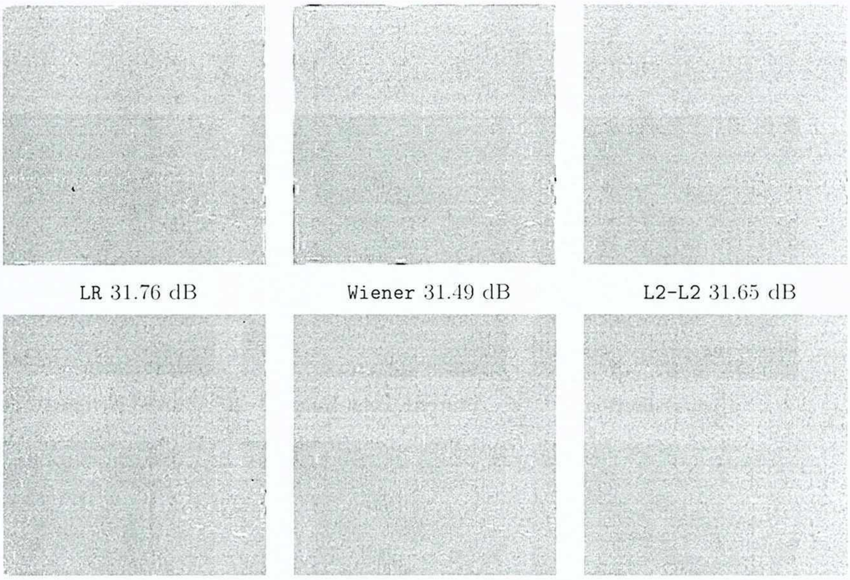


Figure 33: Isotropic Gaussian Kernel: boat image x2 superresolution/deconvolution results.

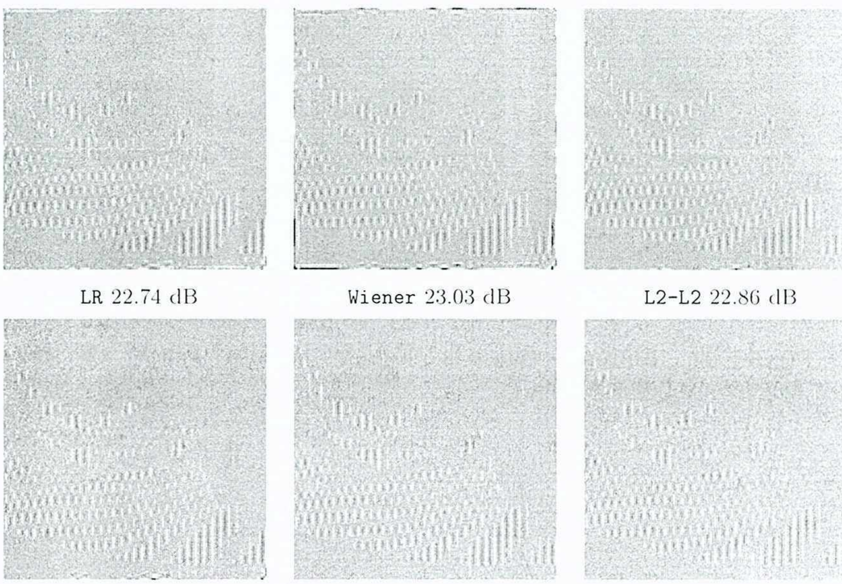
Figure 34: Isotropic Gaussian Kernel: barbara image x2 superresolution/deconvolution results.



LR 31.76 dB | Wiener 31.49 dB | L2-L2 31.65 dB

L2-L08 31.94 dB | L2-L2-D 31.93 dB | L2-L08-D 32.61 dB

Figure 35: Isotropic Gaussian Kernel: peppers image x2 superresolution/deconvolution difference image.



LR 22.74 dB | Wiener 23.03 dB | L2-L2 22.86 dB

L2-L08 22.87 dB | L2-L2-D 22.43 dB | L2-L08-D 22.58 dB

Figure 36: Isotropic Gaussian Kernel: barbara image x2 superresolution/deconvolution difference image.

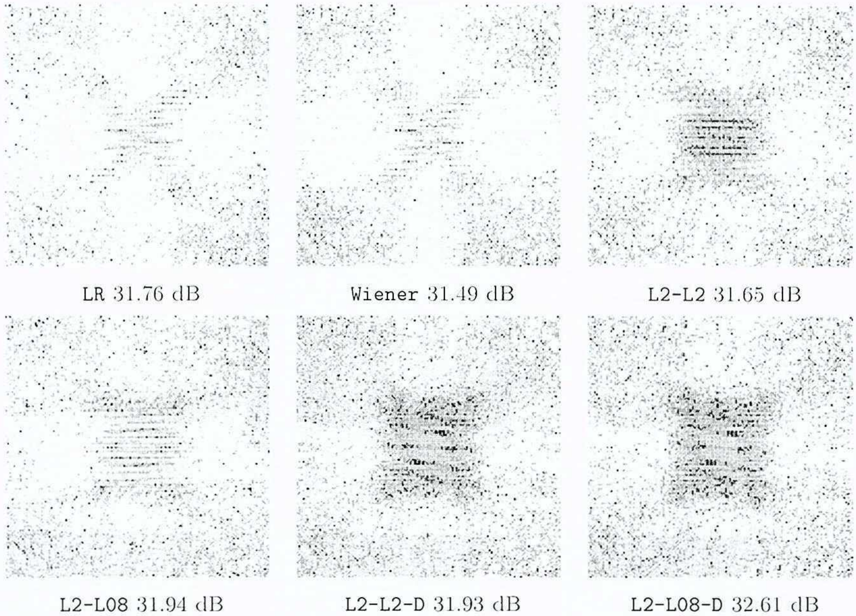


Figure 37: Isotropic Gaussian Kernel: peppers image x2 superresolution/deconvolution norm of the difference image spectrum.

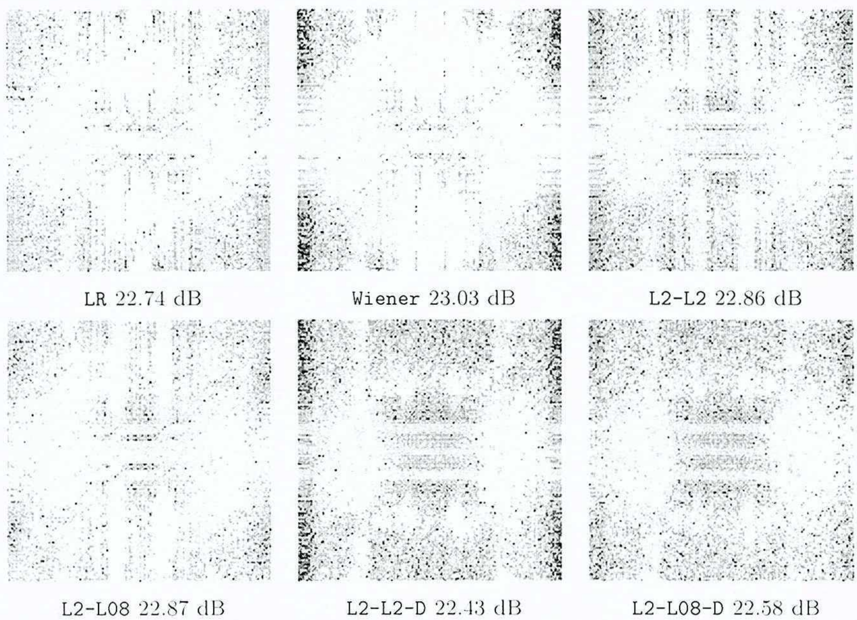


Figure 38: Isotropic Gaussian Kernel: barbara image x2 superresolution/deconvolution norm of the difference image spectrum.

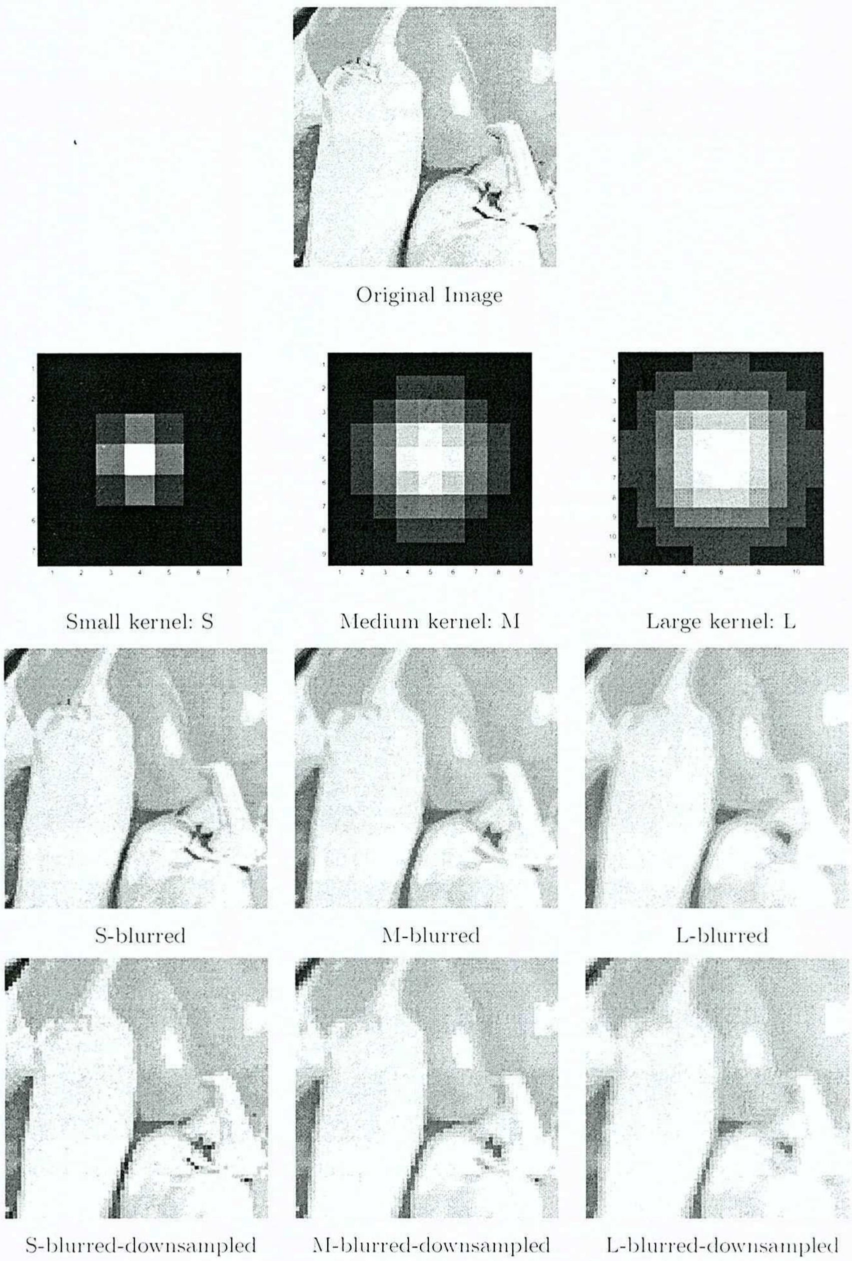


Figure 39: Isotropic Gaussian Kernels: *small-medium-large*, peppers image x2 superresolution/deconvolution image. First

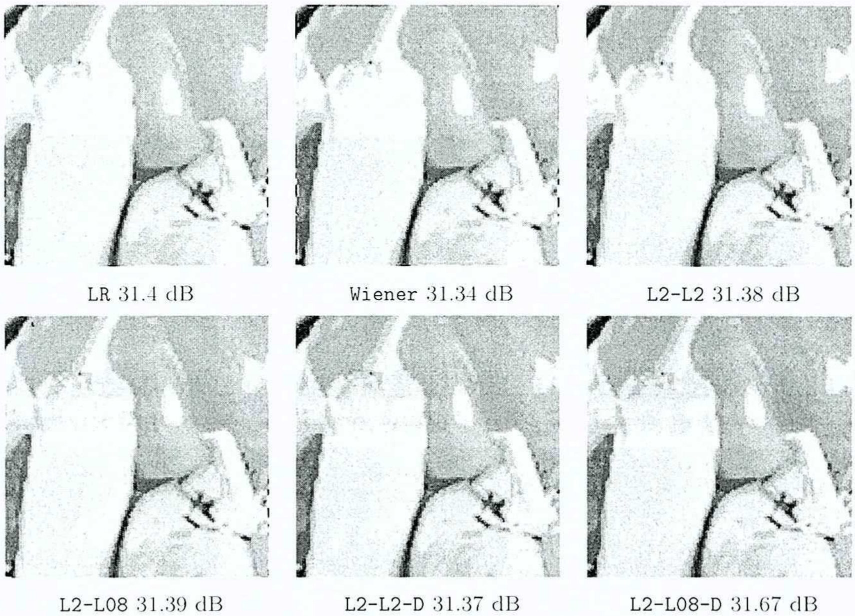


Figure 40: Isotropic Gaussian *small* Kernel: peppers image x2 superresolution/deconvolution image.

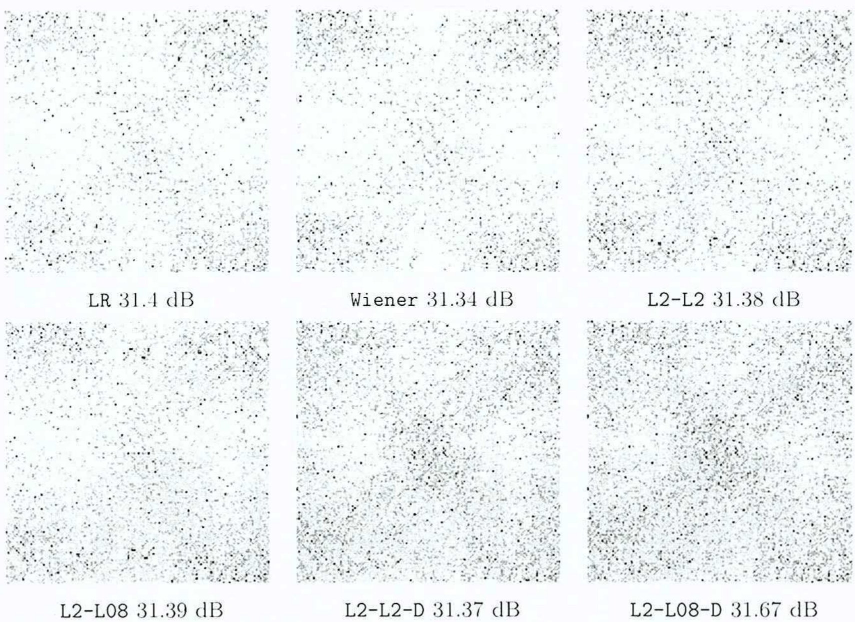
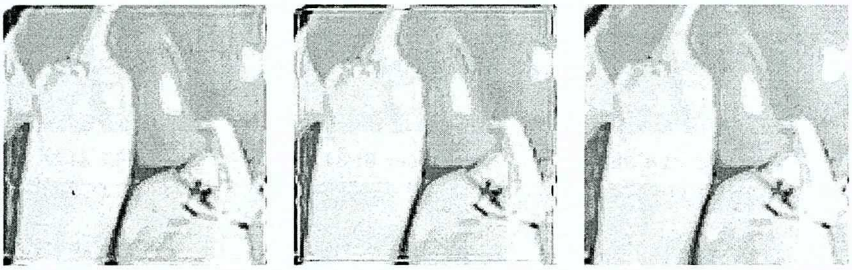


Figure 41: Isotropic Gaussian *small* Kernel: peppers image x2 superresolution/deconvolution difference image spectrum.

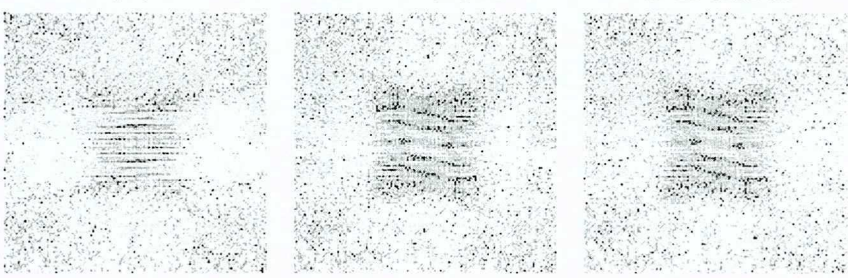


LR 31.34 dB | Wiener 31.3 dB | L2-L2 31.52 dB

L2-L08 31.38 dB | L2-L2-D 31.99 dB | L2-L08-D 32.84 dB

Figure 42: Isotropic Gaussian *medium* size Kernel: peppers image x2 superresolution/deconvolution image.

LR 31.34 dB | Wiener 31.3 dB | L2-L2 31.52 dB



L2-L08 31.38 dB | L2-L2-D 31.99 dB | L2-L08-D 32.84 dB

Figure 43: Isotropic Gaussian *medium* size Kernel: peppers image x2 superresolution/deconvolution difference image spectrum.

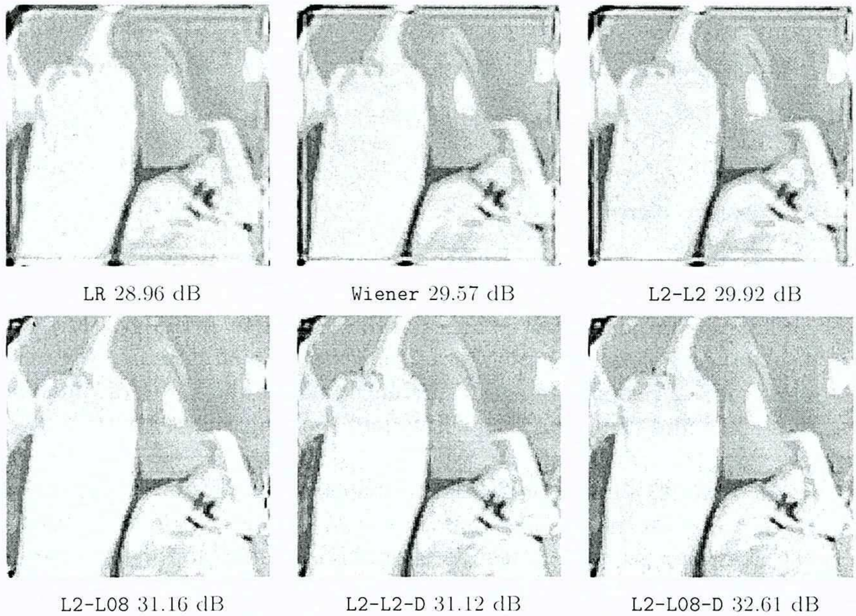


Figure 44: Isotropic Gaussian *large* Kernel: peppers image x2 superresolution/deconvolution image.

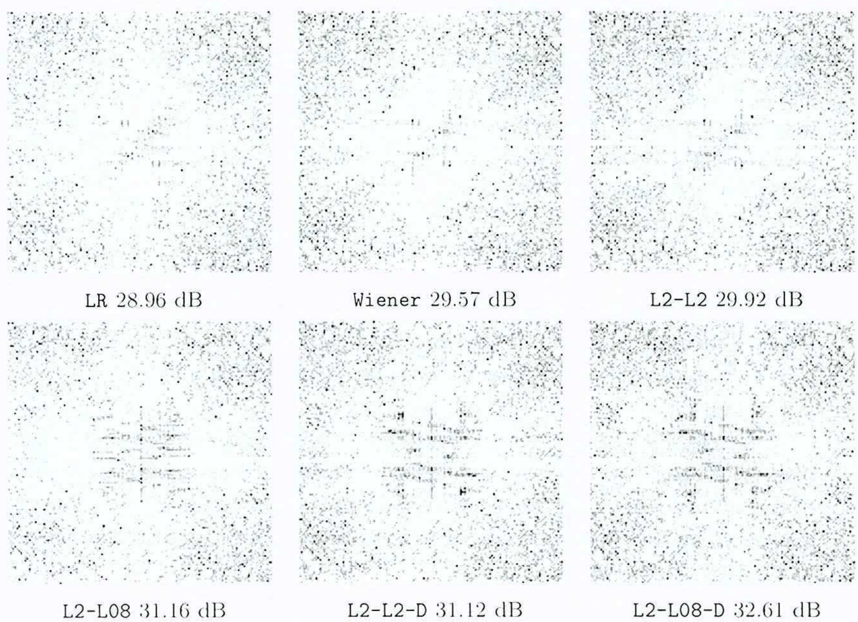


Figure 45: Isotropic Gaussian *large* Kernel: peppers image x2 superresolution/deconvolution difference image spectrum.

Anisotropic Kernel

In this experiment we analyze the importance of having a good estimation of the kernel shape. For that purpose we generate an anisotropic smooth kernel (true) and compute the separable Gaussian kernel (σ_x and σ_y) that best approximates the true kernel in a least squares sense. Figures 46,48 show the $\times 2$ restored images, difference images and norm of difference images spectrum using the true kernel and the Gaussian kernel.

In general, similar performance indicators are found for all algorithms given a certain image. The importance of having a good estimation of the PSF shape to get proper results is verified. Besides leading to lower PSNR, using the Gaussian kernel approximation produces several artifacts along the direction where the kernel estimation mostly differs. A blur effect, similar to camera shake, appears in the resultant superresolved images with the Gaussian approximation.

Super/Sub-Gaussian Kernel

This experiment shows the critical importance of having a good estimation of the kernel decay. We generated a smooth super/sub Gaussian kernel (true) and we computed the separable Gaussian kernel (σ_x and σ_y) that best approximates the true kernel in the least squares sense. Super/sub Gaussian kernels were generated by

$$g(x, y) = Ke^{-(a_1|x|^p + a_2|y|^p)}.$$

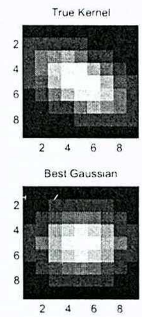
If $p = 2$ then g is a Gaussian kernel, if $p < 2$ is sub-Gaussian and $p > 2$ super-Gaussian. For our experiments we used a sub-Gaussian kernel with $p = 1$ (Laplacian) and a super-Gaussian kernel with $p = 4$.

Figures 49-52 show the considered kernels and their best Gaussian approximations, the x2 restored images, difference images and norm of the difference images spectrum. All algorithms have similar performance for a given image and it is extremely necessary to have a good estimation of the PSF decay to get proper results.

Considering the kernel as Gaussian when is super-Gaussian produces several artifacts. For example, the mast of the ship in the figure 49 is widened significantly. In the case were the true kernel is sub-Gaussian, wrongly considering it as Gaussian produces severe ringing artifacts during deconvolution. In this case, the sparse gradient `L2-L08-D` algorithm seems to be less sensitive to the misestimation of the PSF. This can be shown in Figure 51.

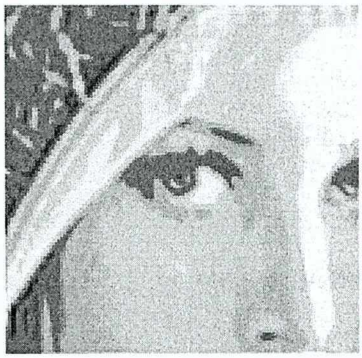
High Resolution

Cubic spline 30.35 dB



True kernel (up)
Gaussian (down)

L2-L2-D (gaussian) 33.57 dB

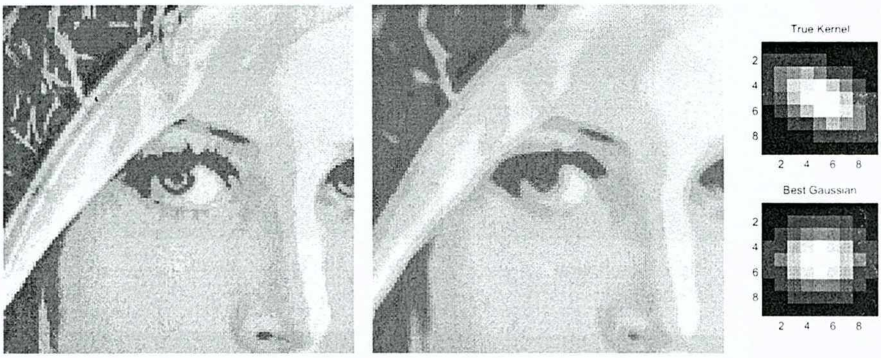


L2-L2-D (true) 35.67 dB

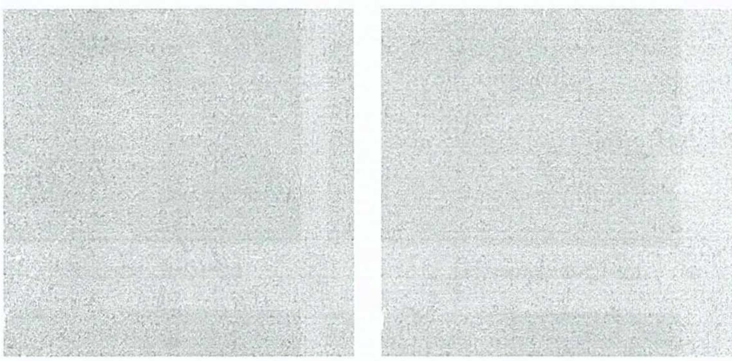
L2-L08-D (gaussian) 33.48 dB

L2-L08-D (true) 35.02 dB

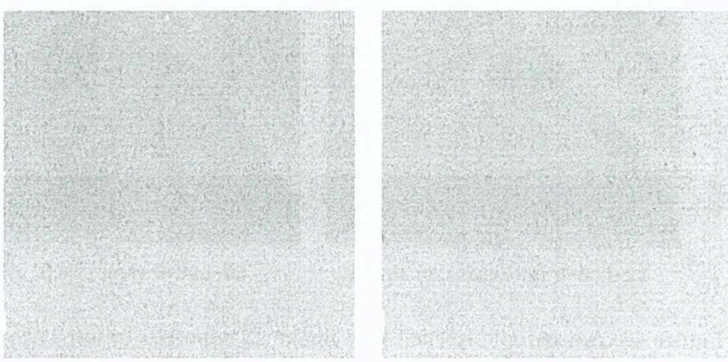
Figure 46: Anisotropic Kernel: lena image x2 superresolution/deconvolution.



High Resolution | Cubic spline 30.35 dB | True kernel (up) Gaussian (down)

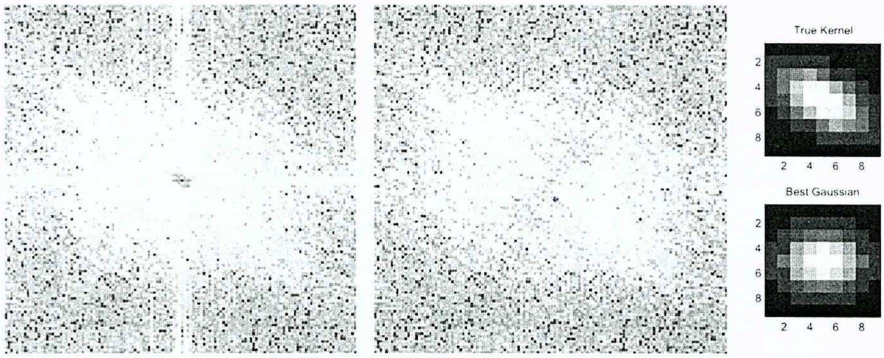


L2-L2-D (gaussian) 33.57 dB | L2-L2-D (true) 35.67 dB

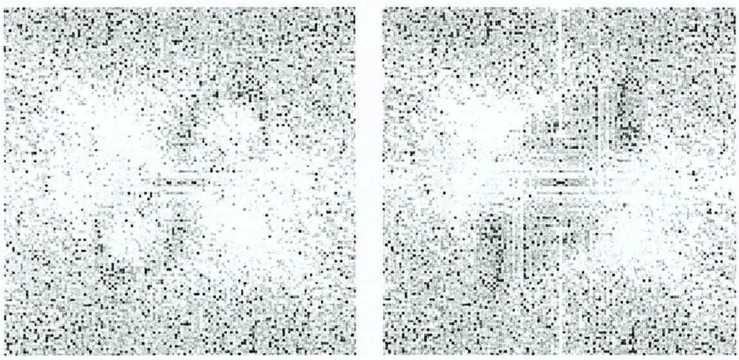


L2-L08-D (gaussian) 33.48 dB | L2-L08-D (true) 35.02 dB

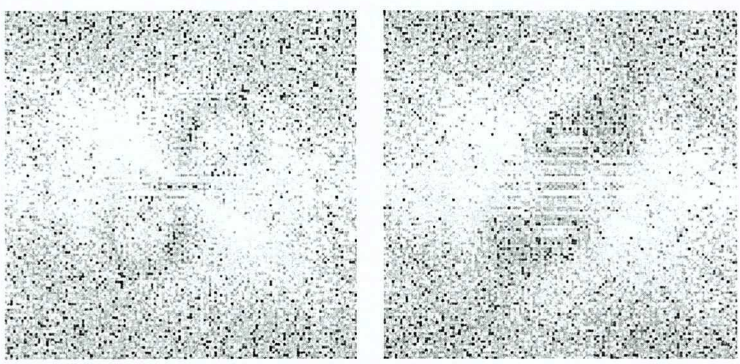
Figure 47: Isotropic Kernel: lena image x2 superresolution/deconvolution difference image.



High Resolution | Cubic spline 30.35 dB | True kernel (up) Gaussian (down)



`L2-L2-D` (gaussian) 33.57 dB | `L2-L2-D` (true) 35.67 dB



`L2-L08-D` (gaussian) 33.48 dB | `L2-L08-D` (true) 35.02 dB

Figure 48: Isotropic Kernel: lena image x2 superresolution/deconvolution difference image spectrum.

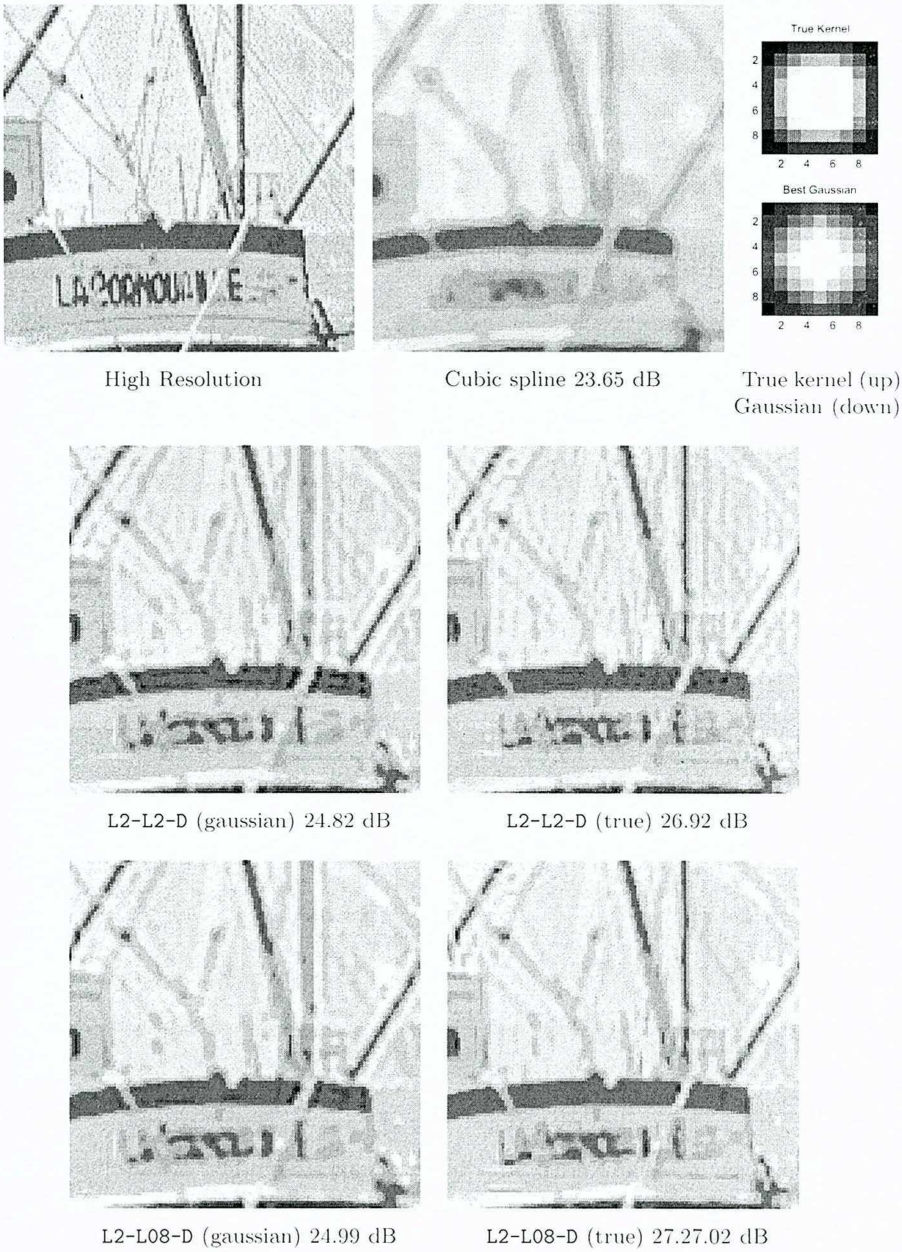
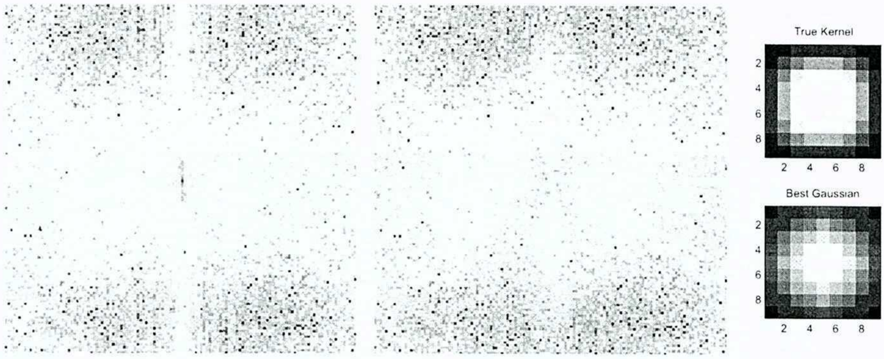
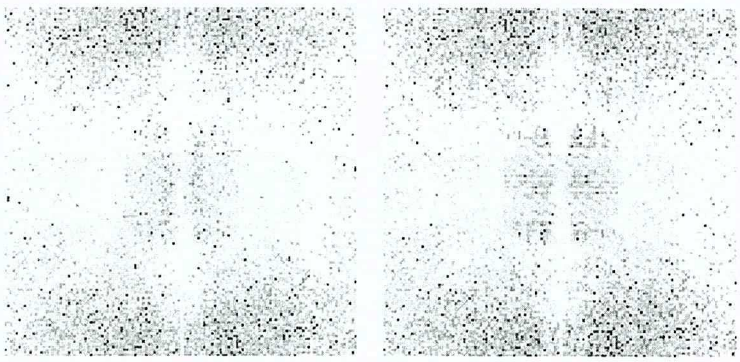


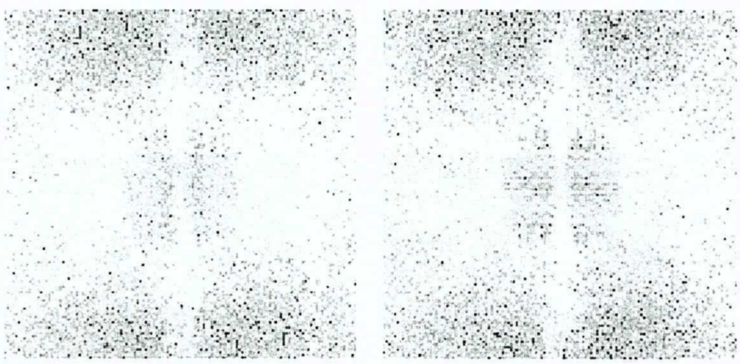
Figure 49: Super-Gaussian Kernel: boat image x2 superresolution/deconvolution. Notice the difference between the images in the mast of the ship.



High Resolution | Cubic spline 23.65 dB | True kernel (up) Gaussian (down)

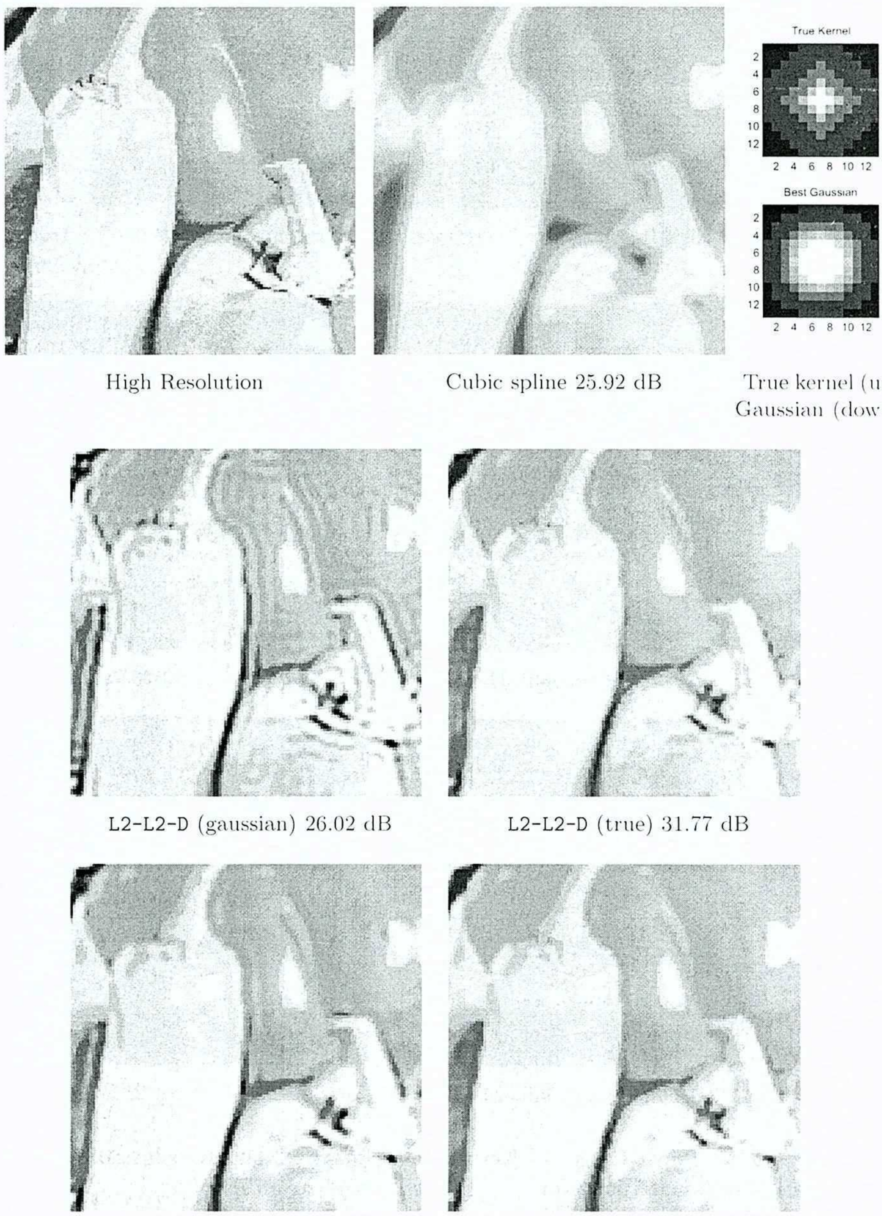


L2-L2-D (gaussian) 24.82 dB | L2-L2-D (true) 26.92 dB



L2-L08-D (gaussian) 24.99 dB | L2-L08-D (true) 27.27 dB

Figure 50: Super-Gaussian Kernel: boat image x2 superresolution/deconvolution difference image spectrum.

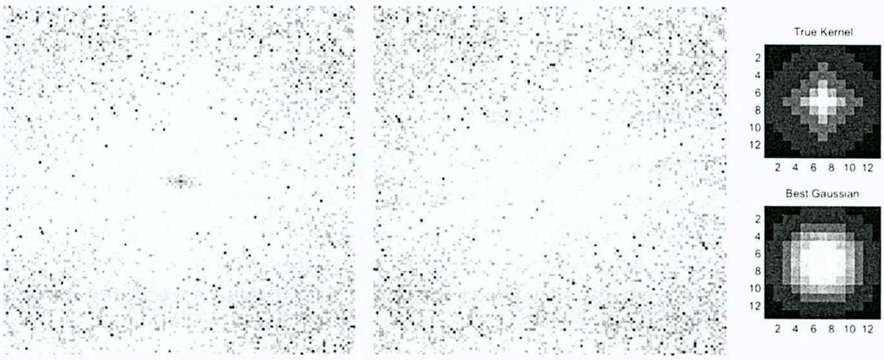


High Resolution — Cubic spline 25.92 dB — True kernel (up) Gaussian (down)

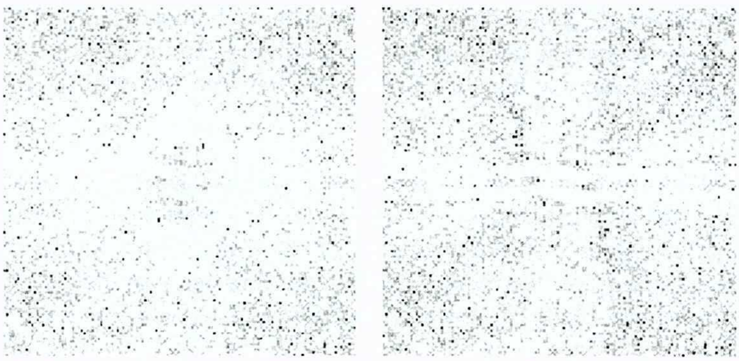
L2-L2-D (gaussian) 26.02 dB — L2-L2-D (true) 31.77 dB

L2-L08-D (gaussian) 29.4 dB — L2-L08-D (true) 32.0 dB

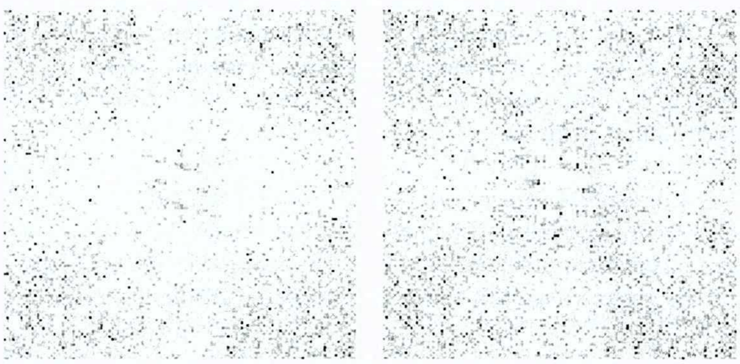
Figure 51: Sub-Gaussian Kernel: peppers image x2 superresolution/deconvolution.



High Resolution | Cubic spline 25.92 dB | True kernel (up) Gaussian (down)



L2-L2-D (gaussian) 26.02 dB | L2-L2-D (true) 31.77 dB



L2-L08-D (gaussian) 29.4 dB | L2-L08-D (true) 32.0 dB

Figure 52: Sub-Gaussian Kernel: lena image x2 superresolution/deconvolution difference image spectrum.

7.2 Real camera images

In this Section we present several examples of PSF usage to zoom images. We used a Canon EOS 400D camera provided with a Tamron SP AF 17-50 mm f/2.8 XR Di II lens at aperture: f/4.5, focal length 40.0 mm and shutter speed 1/50 seconds. The camera was calibrated using the studied PSF estimation algorithm to get $\times 4$ local PSF at different image regions, as it is shown in Figure 53.

A necessary step to compute the PSF is the estimation of the geometrical transformation/distortion introduced by the camera. This can be used to compensate for image distortion (except for a homography). In this particular case, due to the adopted camera configuration, geometrical distortion is minimal so there is hardly no difference between the original and corrected images.

We have computed the separable Gaussian kernel (σ_x and σ_y values) that best fits our subpixel PSF estimation in a least squares sense. This way we intend to show the dependence of the superresolution algorithm to the kernel estimation. We remark that as the Gaussian parameters are estimated from our non-blind PSF estimation this seems to be a best case scenario for Gaussian fitting.

Figures 54-64 show results of super-resolution with the `L2-L2-D` and `L2-L08-D` and the estimated PSF in different regions of the original image in Figure 53. We used the camera green raw channel (as we only intended to validate the proposed approach we discarded half of the green pixels to get a rectangular grid).

Superresolved images are better looking than cubic spline interpolations which tend to over smooth edges. In most images `L2-L08-D` algorithm gives more natural results than `L2-L2-D`. However, this strongly depends on how well the given image fits the assumed sparse gradient model.

We also remark that if the estimated PSF is not very anisotropic, Gaussian approximation gives basically the same results as ours. This can be confirmed in Figures 66-69 where a plot of several particular image lines are shown.

In Figure 64 we show what happens when the kernel is not well approximated by a Gaussian kernel. The difference image in Figure 65 shows differences between the corresponding superresolved images in the direction where the estimated and gaussian approximation kernels mostly differ. However, this difference in the superresolved images seems imperceptible to the naked eye.

Finally we set up a scene to specifically show the effects of using a Gaussian kernel in the case it is not a good approximation (Figure 70). Figure 71 shows the obtained results. For the image restored using the Gaussian kernel, we can see ringing artifacts along the direction where the approximation is less accurate. This can be confirmed in the difference image in Figure 72 and in the peaks that appear in image profiles in Figure 73.

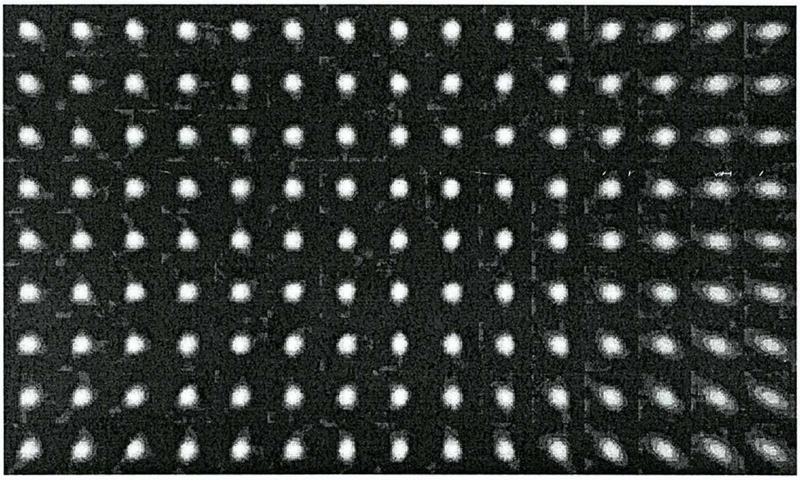


Figure 53: $\times 4$ local PSF estimation for the Canon/Tamron camara/lens at aperture f/4.5, focal length 40.0 mm and shutter speed 1/50 seconds.

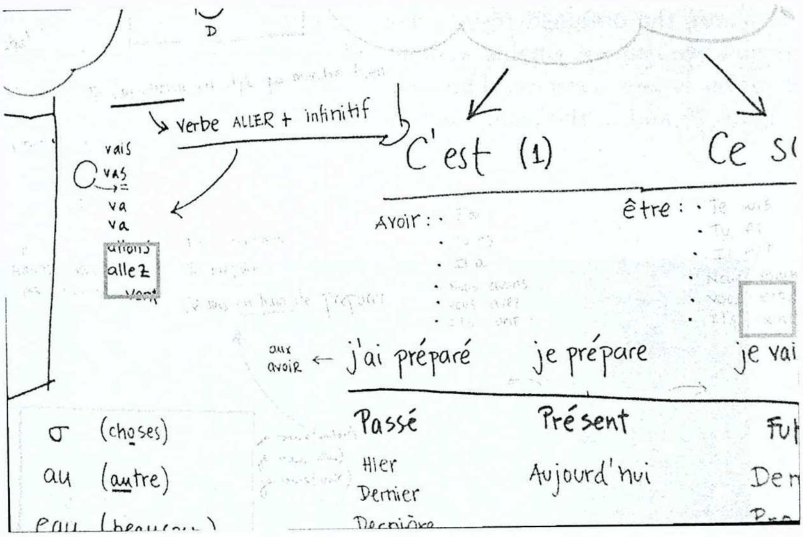
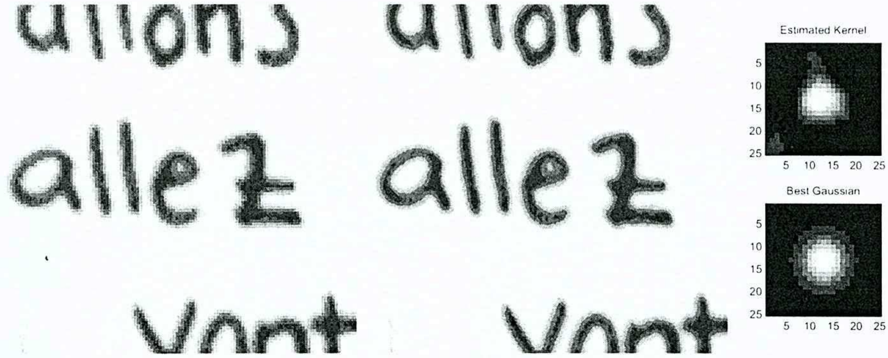
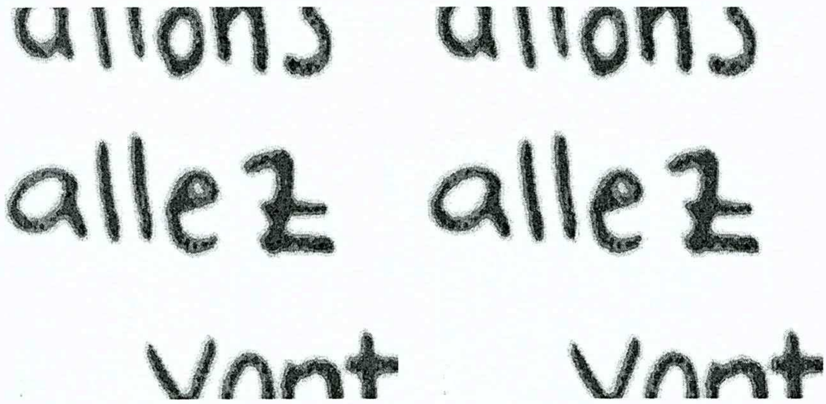


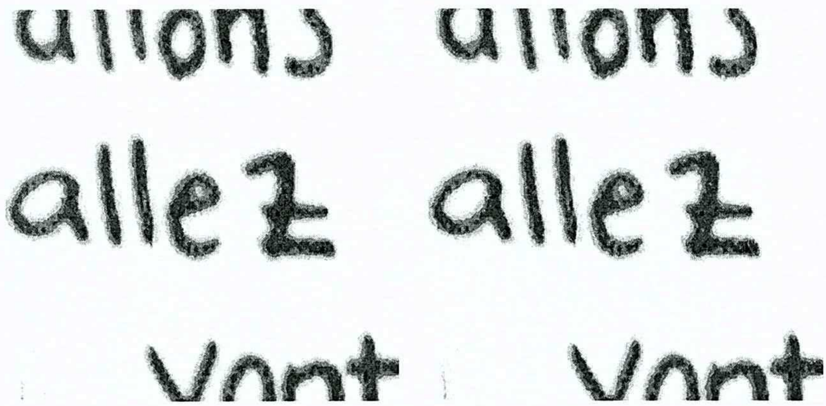
Figure 54: Real image: *papelógrafo*. Boxes indicate where x4 zoom is computed.



Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)

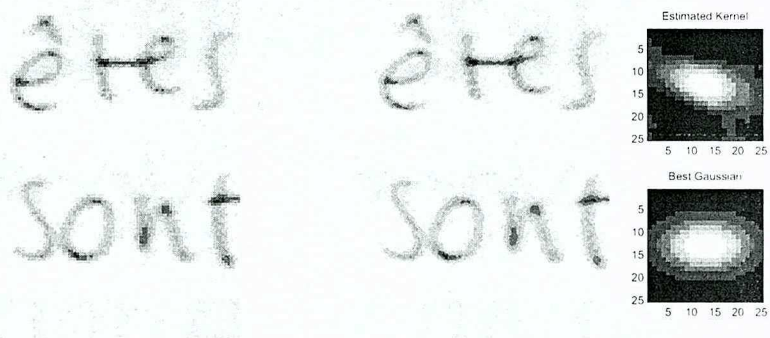


`L2-L2-D` (estimated psf) | `L2-L2-D` (gaussian psf)

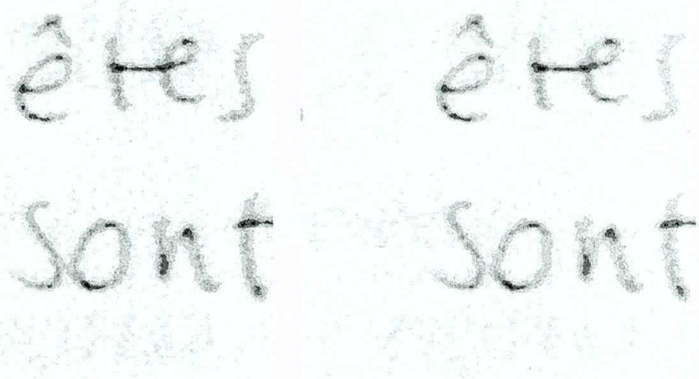


`L2-L08-D` (estimated psf) | `L2-L08-D` (gaussian psf)

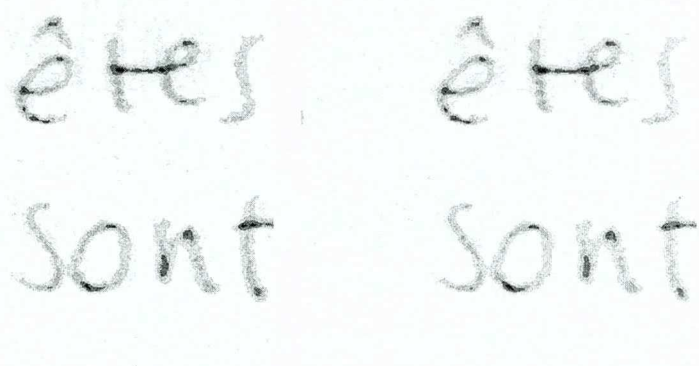
Figure 55: Real image: *papelógrafo CL* x4 superresolution/deconvolution.



Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)



L2-L2-D (estimated psf) | L2-L2-D (gaussian psf)



L2-L08-D (estimated psf) | L2-L08-D (gaussian psf)

Figure 56: Real image: *papelógrafo* x4 superresolution/deconvolution.

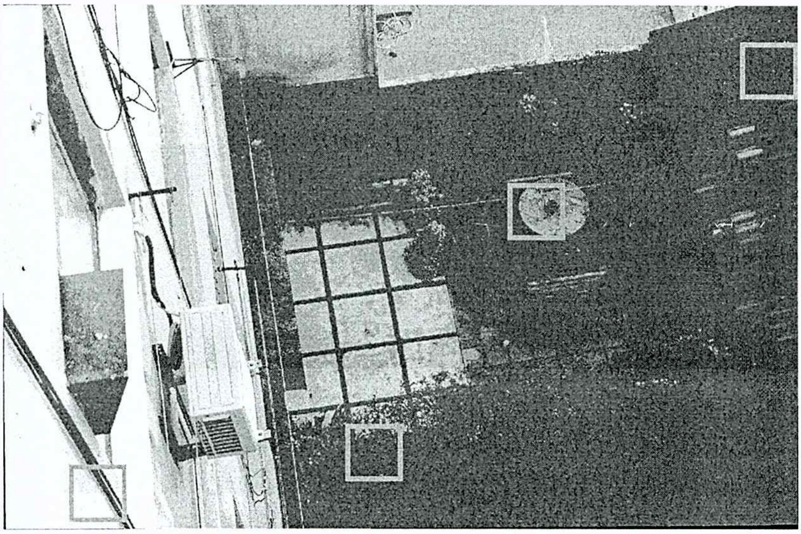
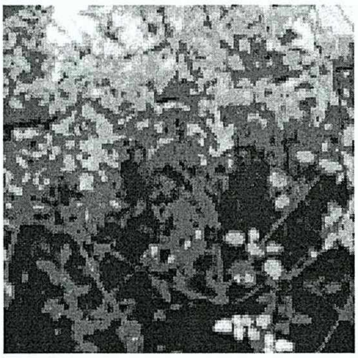
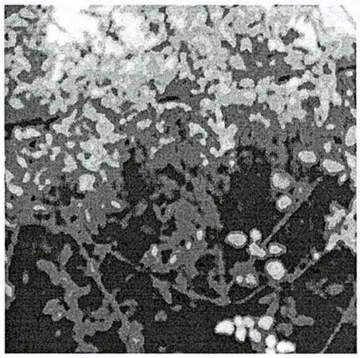


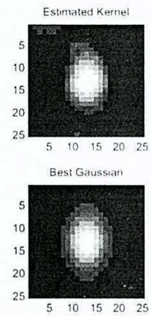
Figure 57: Real image: *planta baja*. Boxes indicate where x4 zoom is computed.



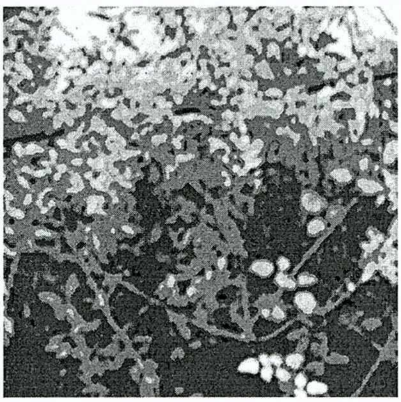
Original Image



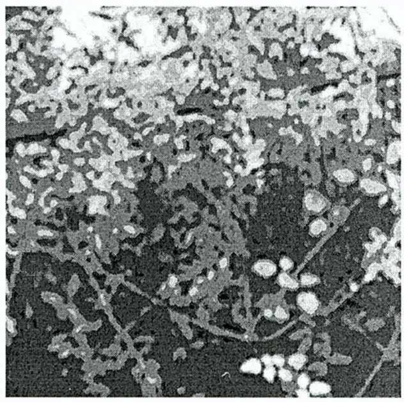
Cubic spline x4



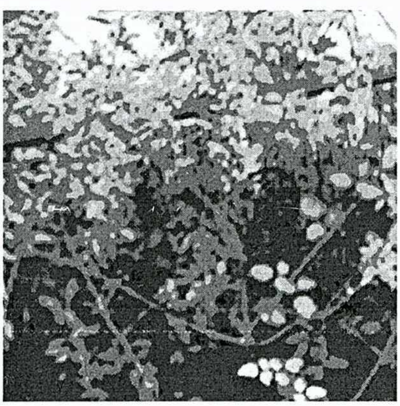
True kernel (up)
Gaussian (down)



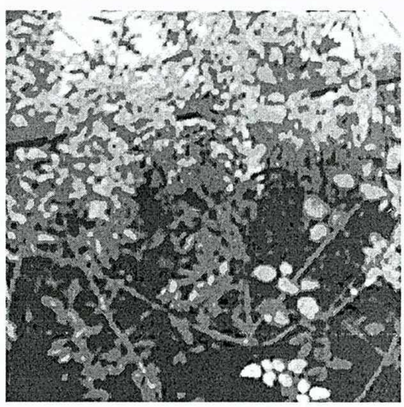
L2-L2-D (estimated psf)



L2-L2-D (gaussian psf)



L2-L08-D (estimated psf)



L2-L08-D (gaussian psf)

Figure 58: Real image: *planta baja BC* x4 superresolution/deconvolution.

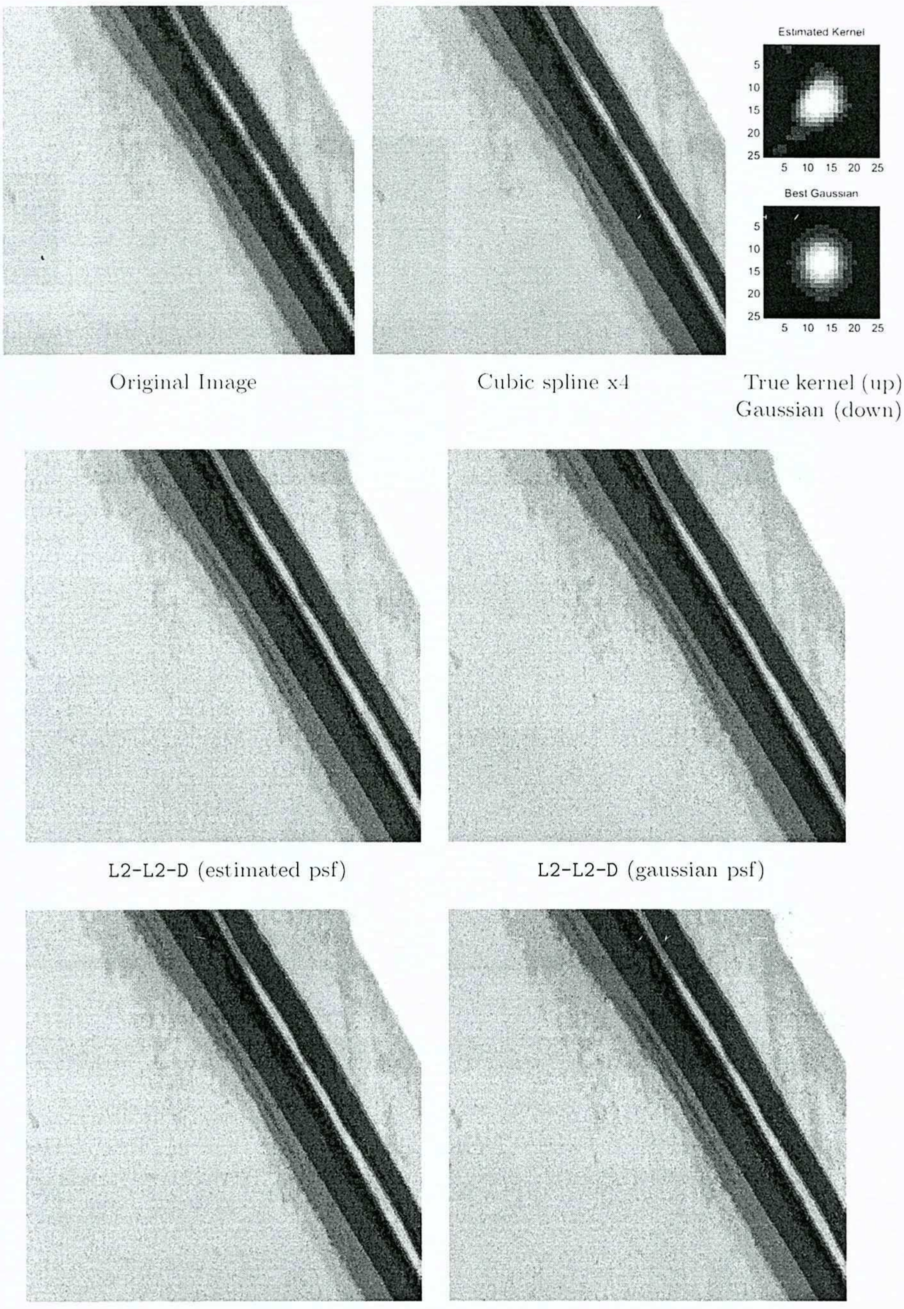
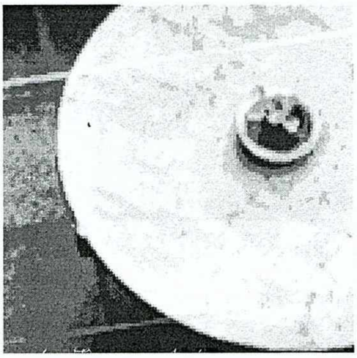
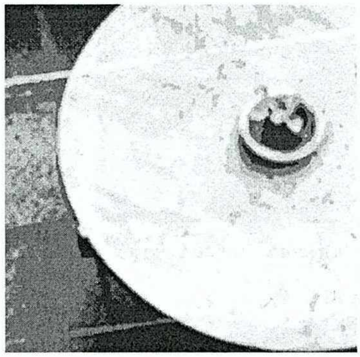
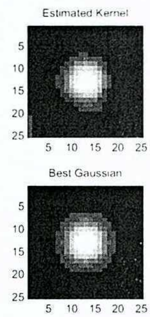


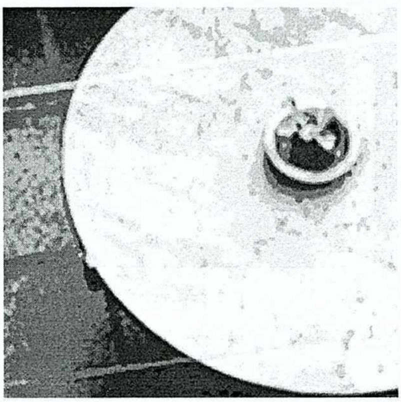
Figure 59: Real image: *planta baja BL* x4 superresolution/deconvolution.

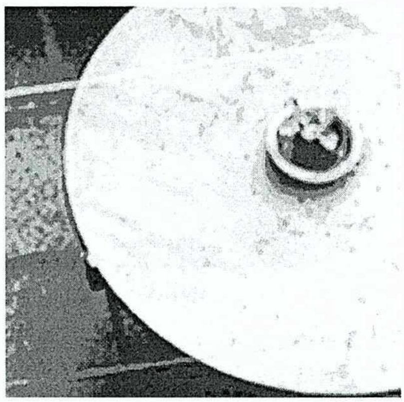




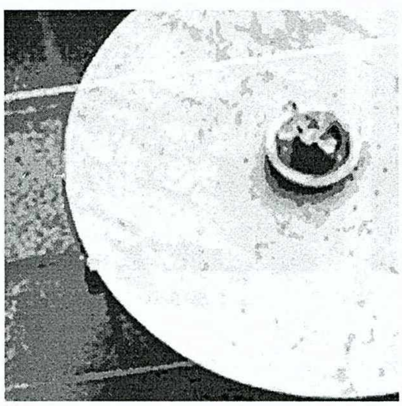


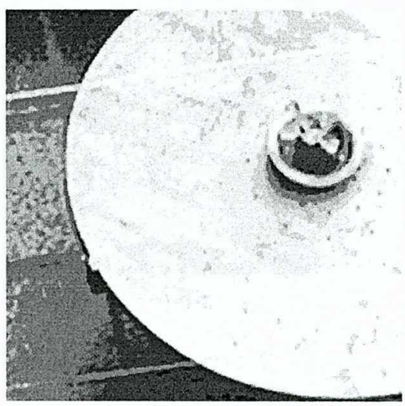
Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)





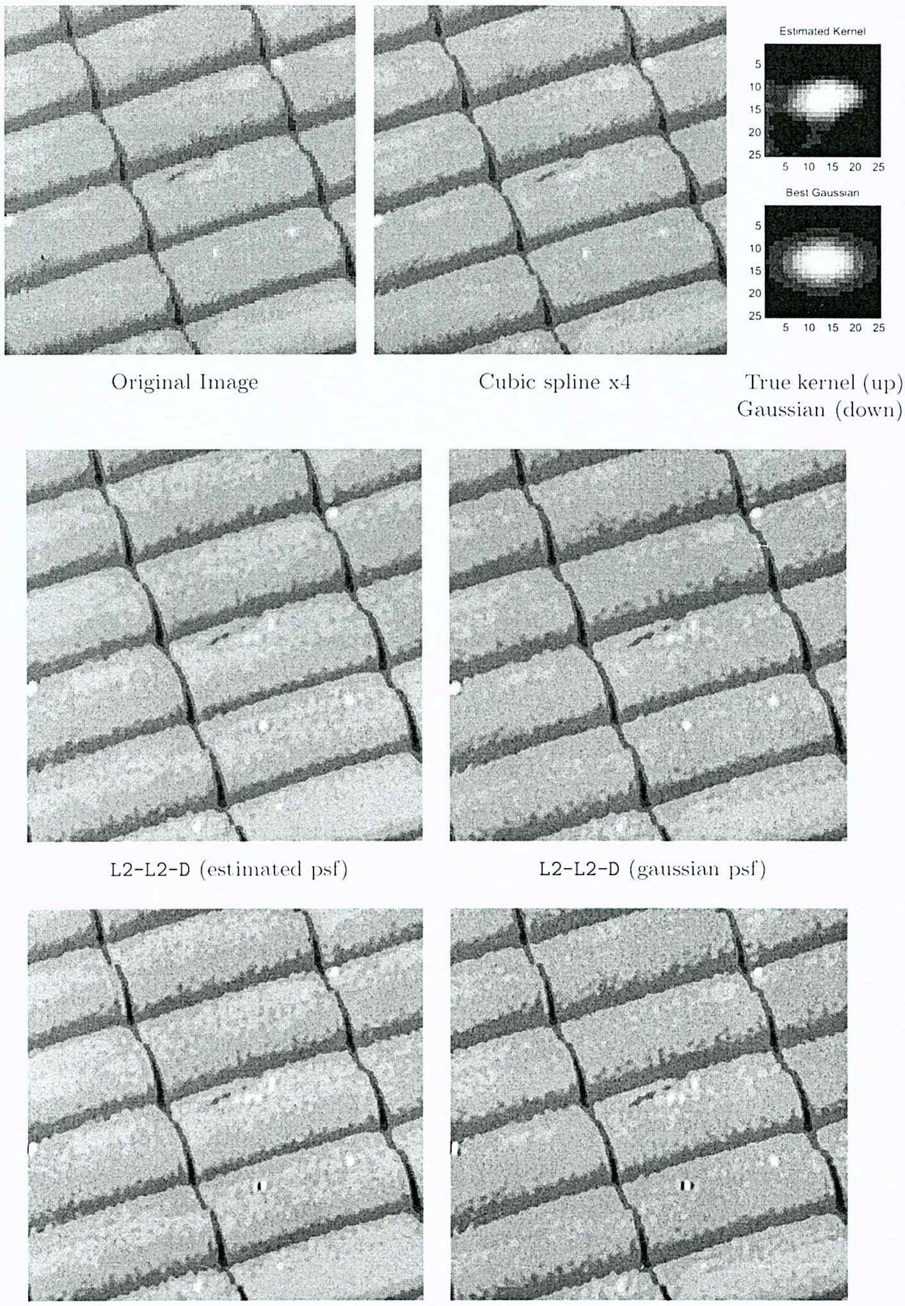
`L2-L2-D` (estimated psf) | `L2-L2-D` (gaussian psf)





`L2-L08-D` (estimated psf) | `L2-L08-D` (gaussian psf)

Figure 60: Real image: *planta baja MM* x4 superresolution/deconvolution.



Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)

L2-L2-D (estimated psf) | L2-L2-D (gaussian psf)

L2-L08-D (estimated psf) | L2-L08-D (gaussian psf)

Figure 61: Real image: *planta baja TR* x4 superresolution/deconvolution.

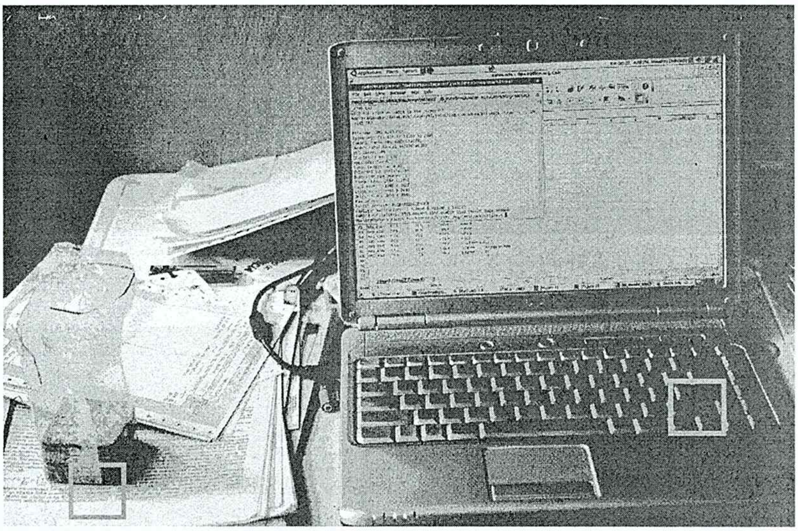
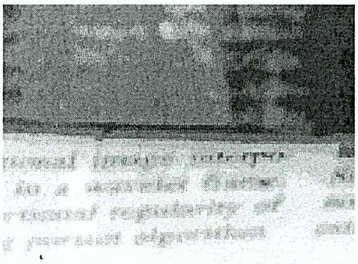
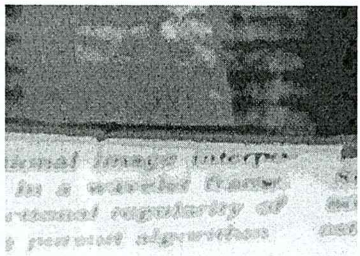


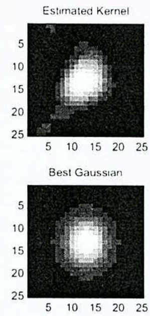
Figure 62: Real image: *laptop*. Boxes indicate where x4 zoom is computed.



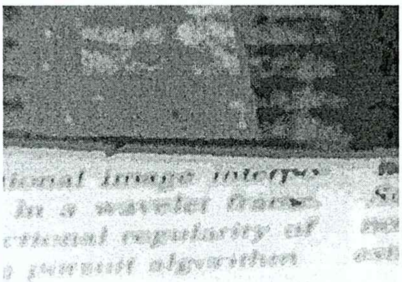
Original Image



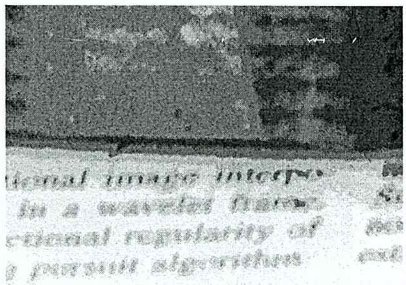
Cubic spline x4



True kernel (up)
Gaussian (down)

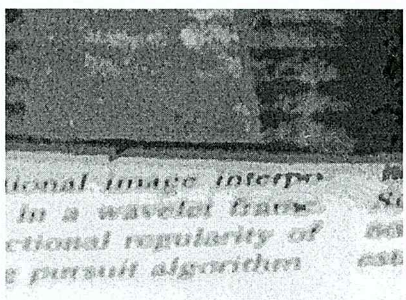


L2-L2-D (estimated psf)



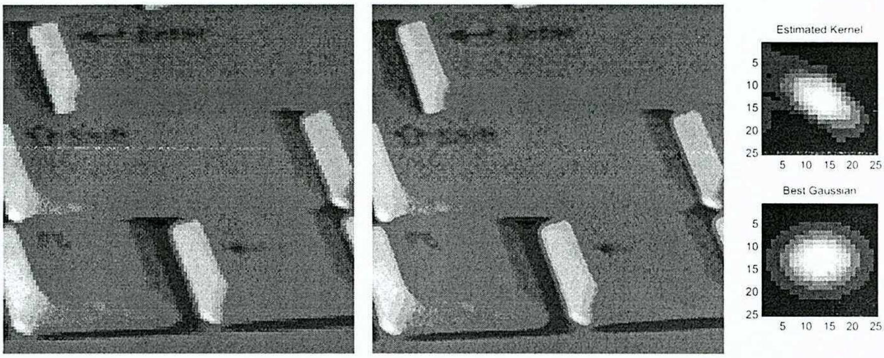
L2-L2-D (gaussian psf)

L2-L08-D (estimated psf)

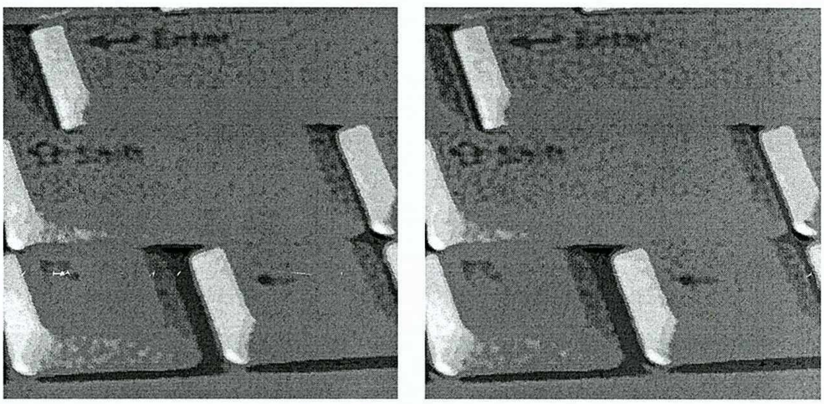


L2-L08-D (gaussian psf)

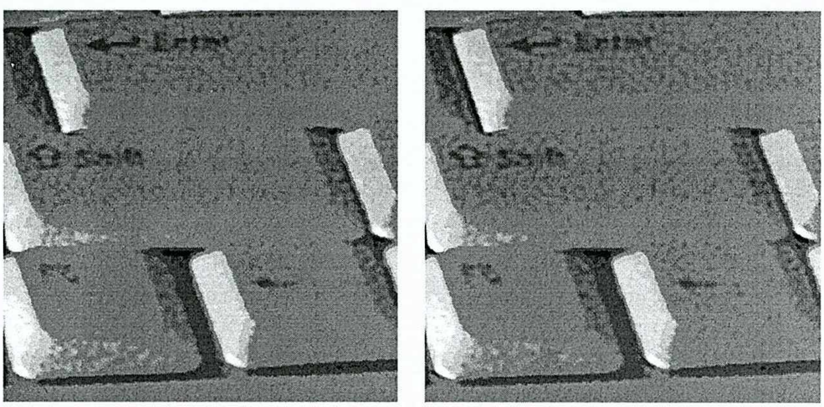
Figure 63: Real image: *laptop BL* x4 superresolution/deconvolution.



Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)

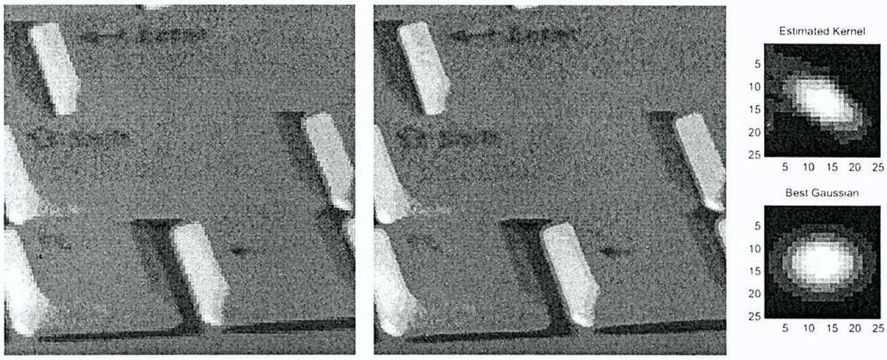


`L2-L2-D` (estimated psf) | `L2-L2-D` (gaussian psf)

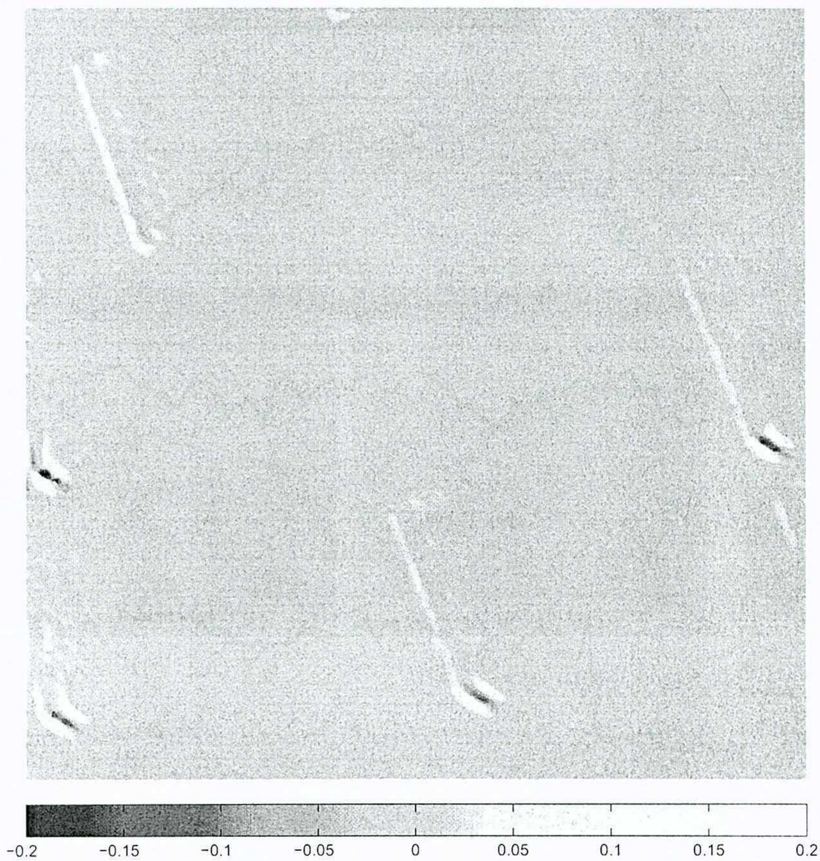


`L2-L08-D` (estimated psf) | `L2-L08-D` (gaussian psf)

Figure 64: Real image: *laptop BR* x4 superresolution/deconvolution.

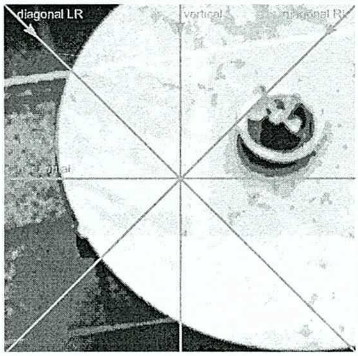


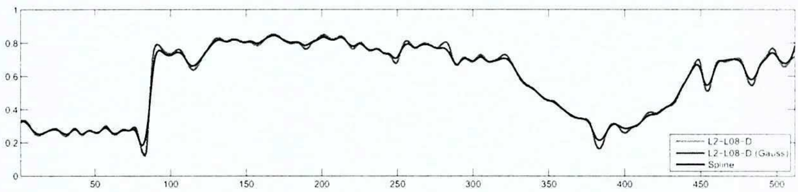
Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)



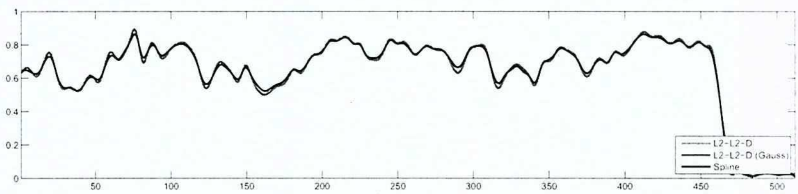
L2-L08-D Estimated vs. Gaussian kernel - Difference image

Figure 65: Real image: *laptop BR* x4 superresolution/deconvolution difference image.

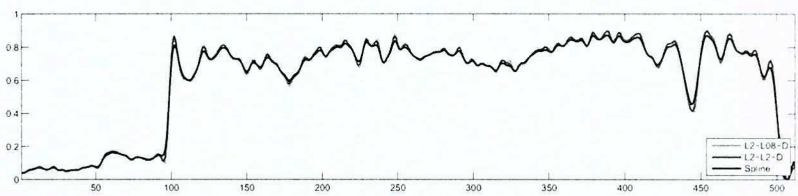




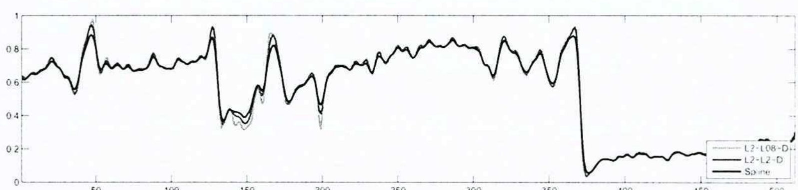
horizontal



vertical

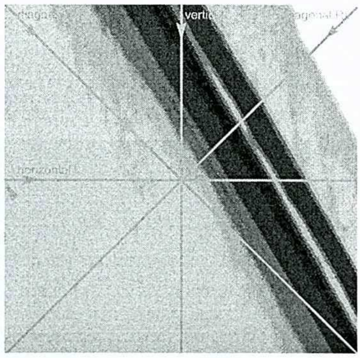


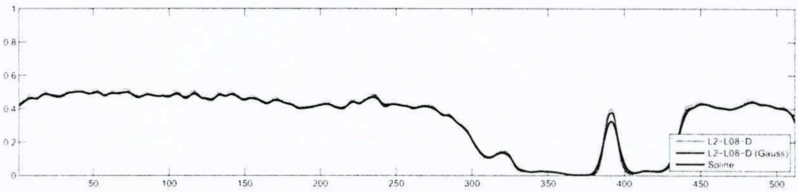
diagonal LR



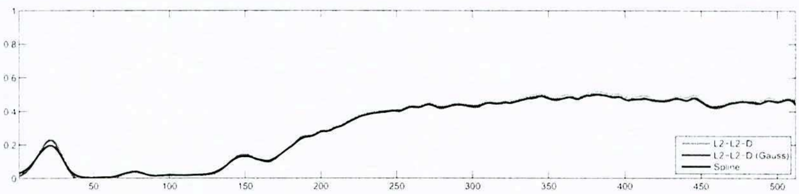
diagonal RL

Figure 66: Real image: *planta baja MM* x4 superresolution/deconvolution profile image.

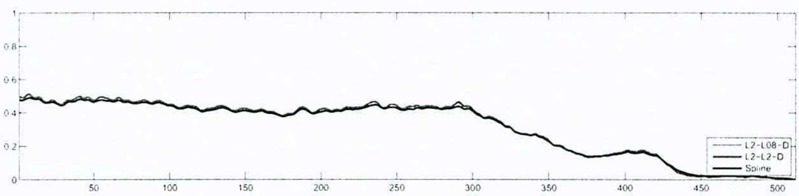




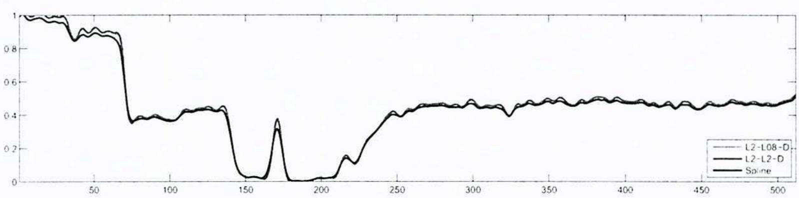
horizontal



vertical

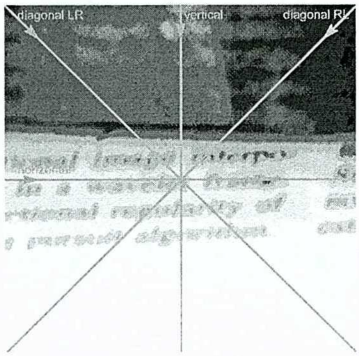


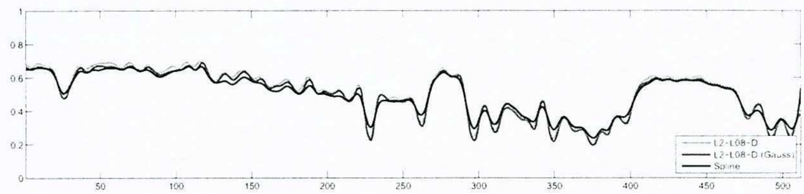
diagonal LR



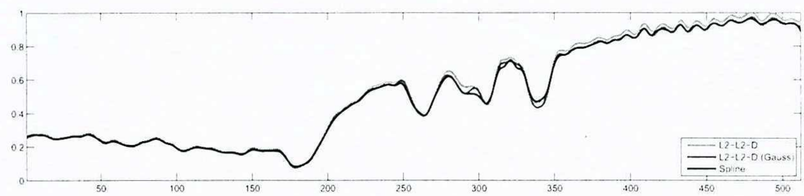
diagonal RL

Figure 67: Real image: *planta baja BL* x4 superresolution/deconvolution profile image.

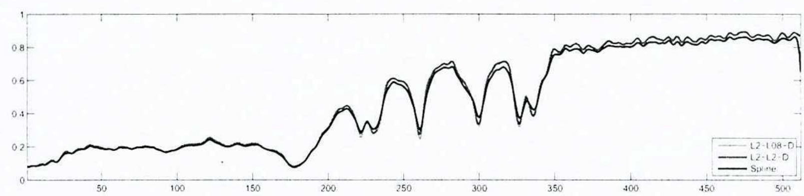




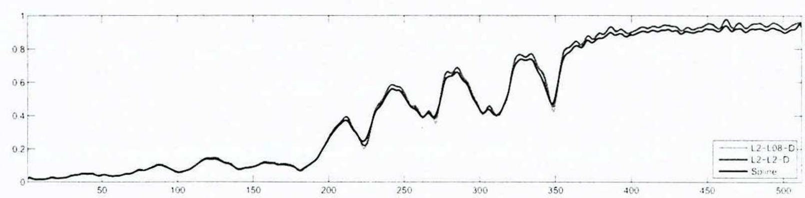
horizontal



vertical

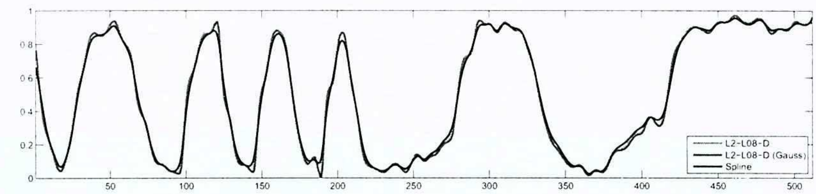


diagonal LR

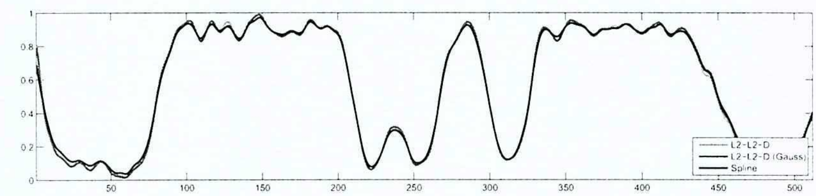


diagonal RL

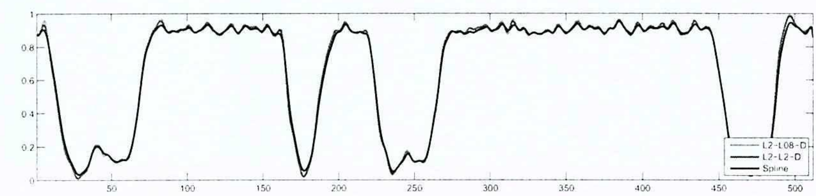
Figure 68: Real image: *laptop BL* x4 superresolution/deconvolution profile image.



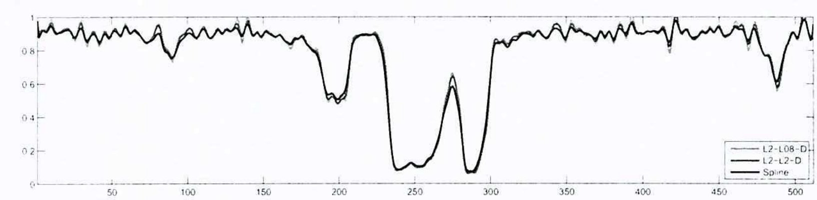
horizontal



vertical



diagonal LR



diagonal RL

Figure 69: Real image: *papelógrafo CL* x4 superresolution/deconvolution profile image.

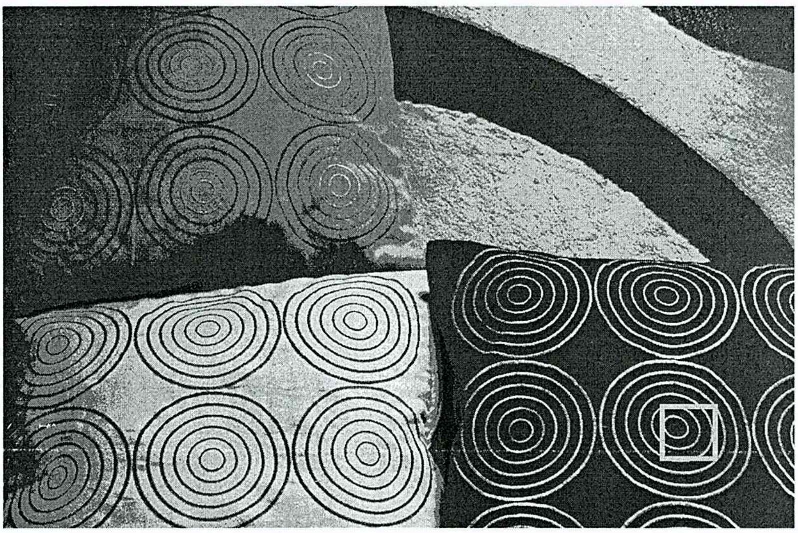
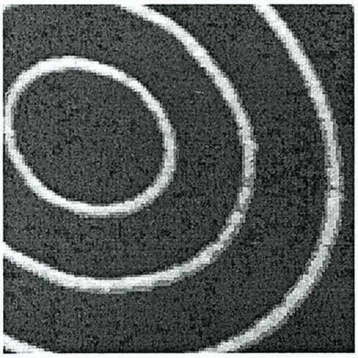
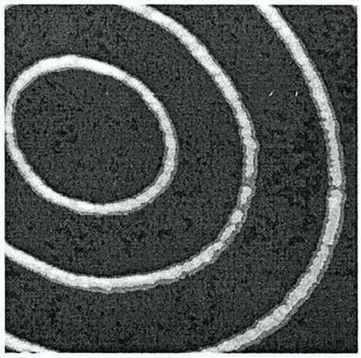


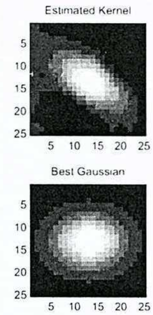
Figure 70: Real image: *círculos*. Boxes indicate where x4 zoom is computed.



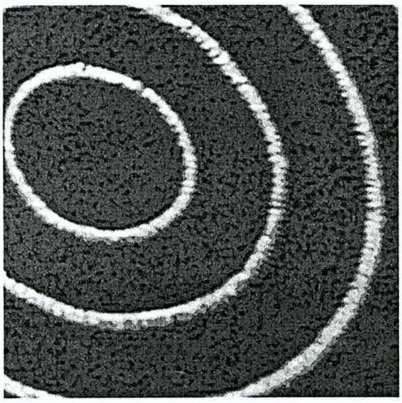
Original Image



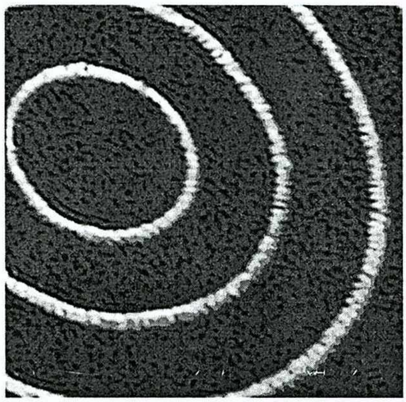
Cubic spline x4



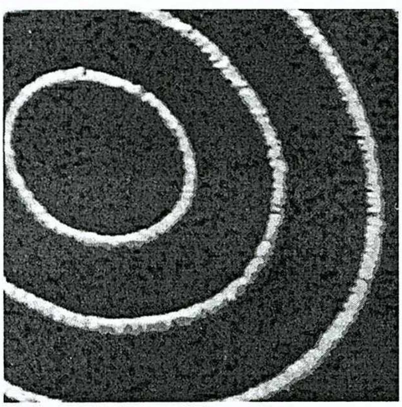
True kernel (up)
Gaussian (down)



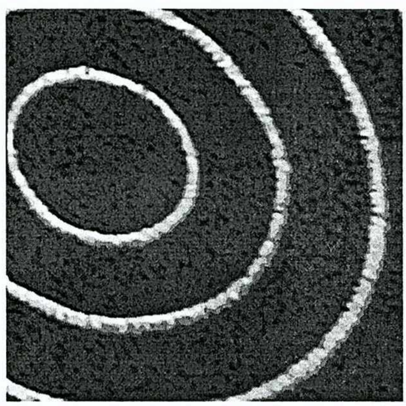
L2-L2-D (estimated psf)



L2-L2-D (gaussian psf)

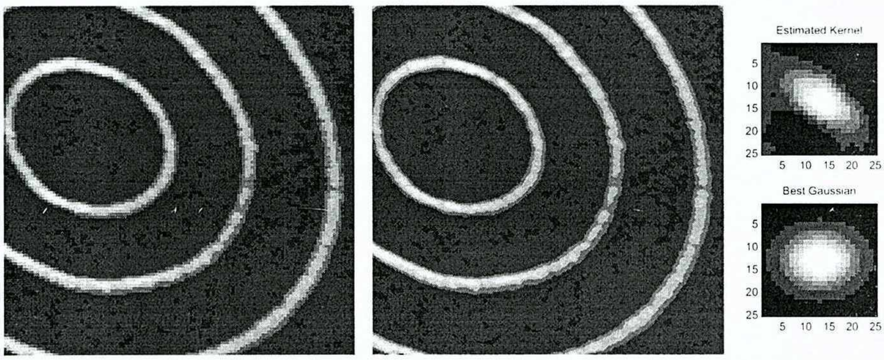


L2-L08-D (estimated psf)

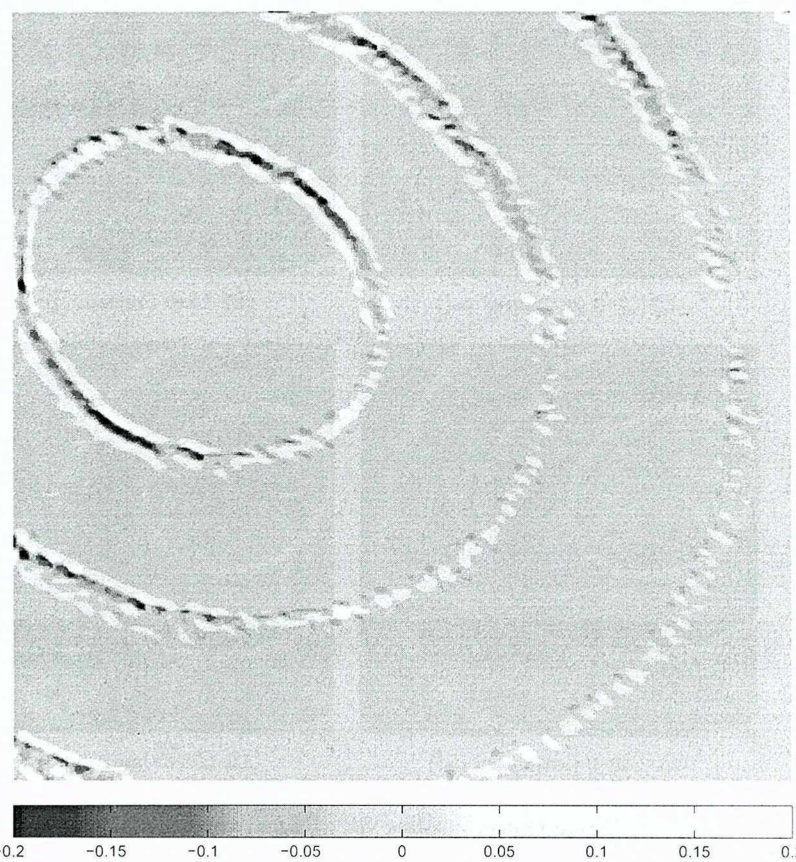


L2-L08-D (gaussian psf)

Figure 71: Real image: *círculos BR* x4 superresolution/deconvolution.



Original Image | Cubic spline x4 | True kernel (up) Gaussian (down)



L2-L08-D Estimated vs. Gaussian kernel - Difference image

Figure 72: Real image: *círculos BR* x4 superresolution/deconvolution difference image.

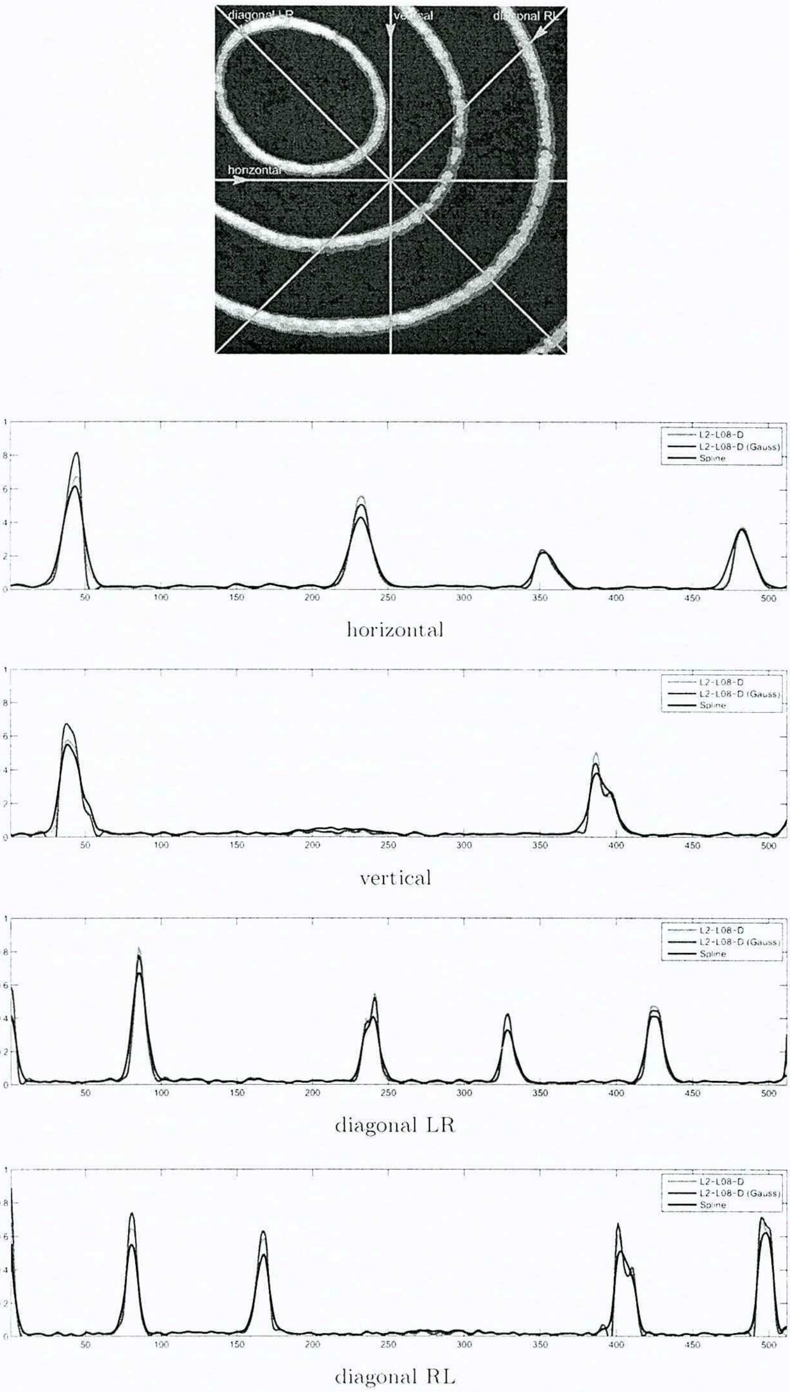


Figure 73: Real image: *círculos BR* x4 superresolution/deconvolution profile image.

8 Discussion

We have analyzed a spatially varying sub-pixel PSF estimation algorithm that captures blur due to intrinsic camera phenomena. The studied algorithm is based on a mathematical digital camera model that takes into account several factors like geometrical distortions due to lens imperfections, diffraction, sensor averaging and out-of-focus.

As part of the analysis of the non-blind PSF approach we studied different methods for aligning the pattern image at sub-pixel precision. This seems to be a key point to really get sub-pixel accuracy. One of the desired characteristics of a good pattern grid is its capability to be easily aligned. Although the chosen pattern presents checkerboard-like corners to facilitate this task, due to aliasing during the acquisition process its detection is not very precise. In this work, we got a precision of about 1/10 pixel in localizing the checkerboard corners at similar digital single-lens reflex camera blur and noise conditions. However, if we are able to reduce the alias, then our checkerboard detector algorithm will get a much higher accuracy. With this aim, we propose to modify the original grid pattern proposed by Joshi et al. and to filter it, only at the checkerboard corners, with a small anti-alias Gaussian kernel. By this, the checkerboard like corners will be easily detectable. Then, we can avoid using the checkerboard region to compute the local PSF by taking advantage of the introduced mask operator. We have a compromise between reducing the number of observations (reducing the mask) and increasing the performance of the alignment stage. We have not tested this yet, but this will be investigated in the future as well as studying other possible grid patterns.

We studied two different ways of modelling the geometrical distortion between the printed grid pattern and the acquired digital image. The non-parametric thin-plate spline seems to outperform the classical polynomial radial distortion approximation considered by most researchers. By this, and the proposed camera model, we can separate the effects of the blur from the distortion. The real camera examples presented in this work, show that the model of a geometrical distortion plus a translation-invariant PSF is consistent in not extreme camera configurations. In particular we show that if the aperture is very wide (small f-number, in our case f/2.8) then a space-variant PSF (local) should be needed to correctly model the camera as the estimation significantly varies from the image center to image borders. It is interesting to study more deeply how the lens focal distance and aperture affect the PSF, in particular for low-price point-and-shoot cameras.

We validated the proposed approach by simulations in which we paid special attention to image alias, trying to simulate real camera acquisitions.

We would like to find a suitable way of choosing the regularization term. For this purpose, we would investigate other ways of stating the mathematical problem of PSF estimation. Instead of directly finding the PSF pixel values, we would like to decompose the PSF in a different base/dictionary and find the best representation over it. By doing this, we can impose other kind of regularizations depending on the base/dictionary chosen.

Precise PSF estimation is of great interest for the evaluation of the camera/lens system. We showed the PSF estimation for different digital single-lens reflex cameras at different apertures, focal distances and shutter time values. By the inspection of the estimated local PSF it is possible to give qualitative/quantitative quality measures of the blur and distortion introduced by the camera.

We also used the super-resolution problem to indirectly evaluate our PSF estimation. Within a Bayesian framework, we proposed a single image deconovolution/super-resolution algorithm that uses the subpixel PSF estimation to find the a posteriori most probable super-resolved image. Our work was based in novel results on natural image statistics which justify a sparse model on the image gradient. Good results are obtained when using our subpixel PSF estimation. In most cases, using a separable Gaussian kernel approximation also gives good results. It is important to notice that the Gaussian kernel parameters (variances in both axes) were fitted using our non-blind subpixel PSF estimation. This seems to be a best case scenario for using a Gaussian kernel parametrization. In regions where the estimated PSF cannot be well approximated by a Gaussian kernel, the superresolved images present some differences and the Gaussian kernel can produce ringing artifacts.

The single image blind super-resolution algorithm proposed in [34] is based in [43] where a justification to choose a Gaussian kernel as a PSF approximation is given. The empirical justification consists in taking a picture of a step edge image in a particular direction and adjust the ideal response to a Gaussian kernel. Their experiments show that for a particular direction the Gaussian approximation fits well. However, this does not mean that the Gaussian kernel is isotropic as used in [34] or even a multivariate kernel. The main advantage of having an accurate low dimensional parametrization of the PSF is that it facilitates the PSF estimation. This is imperative for doing blind PSF estimation and blind image super-resolution.

For that purpose, we would like to find better ways of parameterizing the PSF for the in-focus case studied in this work. One possible way is to take a local three-parameter Gaussian parametrization (i.e. vertical variance, horizontal variance and correlation factor). The camera-objective pair can be calibrated at different apertures, focal distances and shutter time discrete values. Then the estimated Gaussian parameters can be interpolated to get a value for each possible configuration. One problem of taking the Gaussian kernel as approximation is that it does not control the decay speed of the PSF.

Super-resolution real camera experimental results show great dependence on the selected a priori image model. For this reason we propose as future work to research on other kind of super-resolution techniques using more than one image per scene in order to reduce dependence on the image model. It is also of interest to study single image approaches that use non-local multi-scale self examples and need an estimation of the real PSF [44].

A precise knowledge of the subpixel PSF may not be the key element in the natural images superresolution applications. However, if the goal is to get high precision for later use in other applications, e.g. stereo subpixel, the situation might be different. As future work we pretend to evaluate our PSF estimation in such kind of high precision demanding applications.

Lastly, it would be really useful if we could find the highest resolution at which we can give an accurate PSF estimation with a given grid pattern observation. This appears to be associated to the problem of PSF validation.

References

[1] Inc. Canon U.S.A. Canon glossary, 2009. [Online; accessed 28-July-2009].

[2] Neel Joshi, Richard Szeliski, and David J. Kriegman. Psf estimation using sharp edge prediction. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2008.

[3] Neel Joshi. *Enhancing Photographs using Content-Specific Image Priors*. PhD thesis, Department of Computer Science and Engineering, University of California, San Diego, September 2008.

[4] D. C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866+, 1971.

[5] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[6] D. Kundur and D. Hatzinakos. Blind image deconvolution. *Signal Processing Magazine, IEEE*, 13(3):43–64, 1996.

[7] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[8] Eli Levy, Doron Peles, Michal Opher-Lipson, and Stephen G. Lipson. Modulation transfer function of a lens measured with a random target method. *Appl. Opt.*, 38(4):679–683, 1999.

[9] Arnold Daniels, Glenn D. Boreman, Alfred D. Ducharme, and Eyal Sapir. Random transparency targets for modulation transfer function measurement in the visible and infrared regions. *Optical Engineering*, 34(3):860–868, 1995.

[10] M.R. Banham and A.K. Katsaggelos. Digital image restoration. *Signal Processing Magazine, IEEE*, 14(2):24–41, March 1997.

[11] Said Ladjal. *Quantification et flou dans les images numériques*. PhD thesis, Centre de Mathématiques et de Leurs Applications, Ecole Normale Supérieure de Cachan, March 2005.

[12] M. Cannon. Blind deconvolution of spatially invariant image blurs with phase. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 24(1):58–63, Feb 1976.

[13] Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 70, New York, NY, USA, 2007. ACM.

[14] Craig Kolb, Don Mitchell, and Pat Hanrahan. A realistic camera model for computer graphics. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 317–324, New York, NY, USA, 1995. ACM.

[15] Lionel Moisan. Modeling and image processing. Master recherche Maths appliquées 2ème année, Maths Vision Apprentissage (MVA), Ecole Normale Supérieure de Cachan, November 2005.

[16] Wikipedia. Bayer filter — wikipedia, the free encyclopedia, 2009. [Online; accessed 14-July-2009].

[17] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. Fourth Alvey Vision Conference*, pages 147–151, 1988.

[18] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *Int. J. Comput. Vision*, 60(1):63–86, 2004.

[19] O Cheng, W Guangzhi, Z Quan, K Wei, and D Hui. Evaluating harris method in camera calibration. *Conf Proc IEEE Eng Med Biol Soc*, 6:6383–6, 2005.

[20] Luca Lucchese and Sanjit K. Mitra. Using saddle points for subpixel feature detection in camera calibration targets. In *APCCAS (2)*, pages 191–195. IEEE, 2002.

[21] Dazhi Chen and Guangjun Zhang. A new sub-pixel detector for x-corners in camera calibration targets. In *WSCG (Short Papers)*, pages 97–100, 2005.

[22] Frederic Devernay and Olivier D. Faugeras. Straight lines have to be straight. *Machine Vision and Applications*, 13(1):14–24, 2001.

[23] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(6):567–585, 1989.

[24] R. Sprengel, K. Rohr, and H.S. Stiehl. Thin-plate spline approximation for image registration. In *Engineering in Medicine and Biology Society, 1996. Bridging Disciplines for Biomedicine. Proceedings of the 18th Annual International Conference of the IEEE*, volume 3, pages 1190–1191 vol.3, Oct-3 Nov 1996.

[25] Per Christian Hansen and Dianne Prost O'Leary. The use of the l-curve in the regularization of discrete ill-posed problems. *SIAM J. Sci. Comput.*, 14(6):1487–1503, 1993.

[26] Gene H. Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[27] A. N. Iusem. On the convergence properties of the projected gradient method for convex optimization. *Computational and Applied Mathematics*, 22:37–52, 2003.

[28] JONATHAN BARZILAI and JONATHAN M. BORWEIN. Two-Point Step Size Gradient Methods. *IMA J Numer Anal*, 8(1):141–148, 1988.

[29] Mila Nikolova. Optimisation. application à la restauration d'image. Master recherche Maths appliquées 2ème année, Maths Vision Apprentissage (MVA), Ecole Normale Supérieure de Cachan, December 2008.

[30] Luís F. Portugal, Joaquim J. Júdice, and Luís N. Vicente. A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables. *Math. Comput.*, 63(208):625–643, 1994.

[31] M. Grant and S. Boyd. Cvx: Matlab software for disciplined convex programming (web page and software). 2000. [Online; accessed 19-July-2009].

[32] Donghui Chen and Robert J. Plemmons. Nonnegativity constraints in numerical analysis. 2009.

[33] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[34] Wei Zhang and Wai-Kuen Cham. A single image based blind super-resolution approach. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 329–332, Oct. 2008.

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

[36] L. B. Lucy. An iterative technique for the rectification of observed distributions. *Astron. J.*, 79:745+, June 1974.

[37] G.E. Healey and R. Kondepudy. Radiometric ccd camera calibration and noise estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(3):267–276, Mar 1994.

[38] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley Publishing Company, Inc., Boston, MA, USA, 1992.

[39] N. Wiener. *Extrapolation, Interpolation and Smoothing of Stationary Time Series, with Engineering Applications*. Technology Press and Wiley, New York, 1949. note: Originally issued in Feb 1942 as a classified Nat. Defense Res. Council Rep.

[40] Y. Weiss and W. T. Freeman. What makes a good model of natural images? In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.

[41] S. Roth and M. Black. Fields of experts: A framework for learning image priors. 2005.

[42] A. Björck. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, 1996.

[43] David Capel. *Image Mosaicing and Super-Resolution (Cphc/Bcs Distinguished Dissertations.)*. SpringerVerlag, 2004.

[44] Mehran Ebrahimi and Edward R. Vrscay. Solving the inverse problem of image zooming using "self-examples". In *ICIAR*, pages 117–130, 2007.