Blur Space Iterative De-blurring

RADU CIPRIAN BILCU[1], MEJDI TRIMECHE[2], SAKARI ALENIUS[3], MARKKU VEHVILAINEN[4]
[1,2,3,4]Multimedia Technologies Laboratory,
Nokia Research Center
Visiokatu 1, FIN-33720, Tampere
FINLAND

Abstract: - The main idea of image restoration in the blur space is first to obtain a sequence of blurred images using a set of known point spread functions. Extrapolation of this sequence of images with respect to the blur parameter then gives the restored image. Usually, blur space restoration is done in a non-iterative manner and the amount of de-blurring is a parameter of the algorithm. In this paper, an iterative blur space restoration algorithm is proposed. Because of a simple stopping rule, the de-blurring parameter does not need to be predefined. Moreover, the proposed method contain a regularization procedure at pixel level that prevents edge overshooting. Results showing the improved performance of the proposed method, as opposed to the global methods, are presented.

Key-Words: - Image Restoration, Image De-blurring, De-convolution, Iterative Restoration, Regularization.

1 Introduction

In all image acquisition systems, digital images are corrupted by blur and noise. Blur can be introduced by the relative motion between the camera and the scene, or by the optical system that is out of focus. In the case of aerial images the atmospheric turbulence can also introduce blur in the recorded images. Noise is another important source of degradation and it can be introduced by the recording medium (film, digital sensor), the transmission medium, measurement, and quantization errors. As a consequence, de-blurring and de-noising are very important topics in image processing and restoration. Many solutions have been introduced in the open literature (see [1], [2], [3], [4], [5], [10], [8] and the references therein).

When de-blurring is addressed, various proposed algorithms are compared also from the noise amplification point of view. Some de-blurring algorithms explicitly include a noise reduction method [1], [5], whereas others do not address this problem [4].

Although one can make other classification of the de-blurring methods, they are classified here as iterative and non-iterative approaches. In non-iterative methods, the result is obtained through a one pass processing algorithm, e.g. Laplacian high pass filtering [8], unsharp masking [10], blur domain de-blur [1], frequency techniques [5] to mention a few. In iterative methods, the result is refined during several processing passes. The de-blurring process is controlled by a cost function that sets the criteria for the refining process, e.g. Least Squares method [8] or adaptive Landweber algorithm [4].

In the above mentioned iterative methods, at each iteration an estimate of the clean image is computed and the accuracy of the estimation increases with the number of iterations. However, typically after few iterations, there is not much improvement between adjacent steps. Moreover, the continuation of the de-blurring beyond a certain point might introduce annoying artifacts in the restored image (overshooting of the edges due to over-emphasis of the details or even false colors). As a consequence, algorithms for convergence detection were also proposed in the open literature (see [6] and the references therein). These algorithms try to stop the iterations at an optimum point (e.g. when the improvement in the restoration is very small or when the restored image contains overshoots). Another iterative approach to solve the de-blurring problem is to apply repeatedly a one step de-blurring method with varying parameters and the best result is kept (blind de-convolution). The main problem in these methods is to find a criterion to define the best restoration result.

Our new approach belongs to the class of iterative restoration algorithms in which a one step de-blurring method is repeatedly applied to the observed image. In the new method, a simple rule to measure the quality of the restoration at each iteration is defined. Based on this rule the iterative restoration process is stopped near an optimum point. Moreover, edge overshooting is reduced by a local regularization technique. The one step de-blurring method implemented in the iterative process is the blur domain de-blurring from [1][1].

2 Existing Approach

Let us consider the following model of a degraded image in the presence of additive noise:

$$I = h * f + \eta \tag{1}$$

where I is the observed distorted image, f is the ideal image, h is the point spread function (PSF), η is the additive noise and $*$ denote the convolution operator.

When the PSF h is Gaussian with variance $\sigma^2 = v$ the above equation can be written as follows:

$$I(v) = h * I(0) + \eta \tag{2}$$

where $I(v)$ denotes the fact that the observed image is blurred with the parameter v and $I(0)$ is the original image (it can be considered to be blurred with parameter $v = 0$).

The goal is to find $I(0)$ from the observed image $I(v)$. For doing this, an interesting approach was proposed in [1] where extrapolation in the blur space is used. The blur space restoration technique is based on the compositional properties of the images blurred by a Gaussian PSF (see [1] for more details).

The following non-iterative algorithm was detailed in [1] for the 1-Dimensional case and the 2-Dimensional implementation was also discussed (the observed image is denoted as $I(x, v_0)$ with v_0 being the original blur and x the pixel coordinate):

1. Blur the observed image using N Gaussian PSF's with variances $b_1, b_2, \ldots, b_N$. The blurred images are denoted as $I_i(x) = I(x, v_0 + b_i)$ for $i = 1, \ldots, N$.

2. Compute the derivatives of the images $I_i(x)$ as follows:

$$I_i'(x) = \frac{\partial I\left(x, v_0 + b_i\right)}{\partial\left(v_0 + b_i\right)} \approx \frac{1}{2}\frac{\partial^2 I_i(x)}{\partial x^2} \tag{3}$$

3. Choose weights $w_{0,i}$ for each of $I_i(x)$ and $w_{1,i}$ for I_x'.

4. Choose the order M of the polynomial that is used to model the pixels in the blur space:

$$P(x; d_i) = c_0(x) + c_1(x)d_i + \ldots + c_M(x)d_i^M. \tag{4}$$

[1]Other non-iterative restoration methods can be accommodated as well.

where x denote the fact that different coefficients of the polynomial are defined for different pixel positions x and $d_i = v_0 + b_i$ is the blur applied to the i^{th} image.

5. Solve the following system of $2N$ equations, by the method of least squares, to obtain the coefficients $c_i(x)$.

$$\begin{aligned} w_{0,i}\left(c_0(x) + c_1(x)d_i + \ldots + c_M(x)d_i^M\right) &= \\ &= w_{0,i}I_i(x) \\ w_{1,i}\left(c_1(x) + c_2(x)d_i + \ldots + c_M(x)d_i^{M-1}\right) &= \\ &= w_{1,i}I_i'(x) \end{aligned} \tag{5}$$

where $i = 1, \ldots, N$.

6. The pixel at position x in the restored image is equal to the value of the coefficient $c_0(x)$. This is the value of the polynomial at the origin of the blur space where blurring is zero.

In the above method, the original blur v_0 in the observed image has to be known in order to make the exact restoration. In practice, this parameter is not a-priori known and the image is de-blurred by a predefined amount. Due to this fact it is interesting from practical point of view to modify this method so that the de-blurring parameter v_0 can be approximated.

The above steps, detailed for the 1-Dimensional case, can be extended for 2-Dimensional images using the separability of a multivariate Gaussian [1]. Specifically the above 1-dimensional method can be applied first along the vertical dimension of the image and second along the horizontal dimension of the image [1]. In a simple case of symmetrical PSF, the horizontal and vertical blurring parameters are the same and the formalism of (3), (4) and (5) remains.

The coefficients $w_{0,i}$ and $w_{1,i}$ must be initialized at the beginning of the algorithm. Based on their values, different approximations can be accommodated. The technique of unsharp masking can be viewed as a special case of the above method (when $w_{1,i} = 0$ for all $i = 1, \ldots, N$). Different numbers of blurred images N and various parameters generate the whole class of unsharp mask filters [1].

3 The Proposed Iterative Method

In this section, an iterative approach that is an extension of the method in [1] is introduced. The proposed method is equipped with a stopping rule that detects the optimum number of iterations and with a regularization mechanism at pixel level, that prevents overshooting of the image details. The main idea of

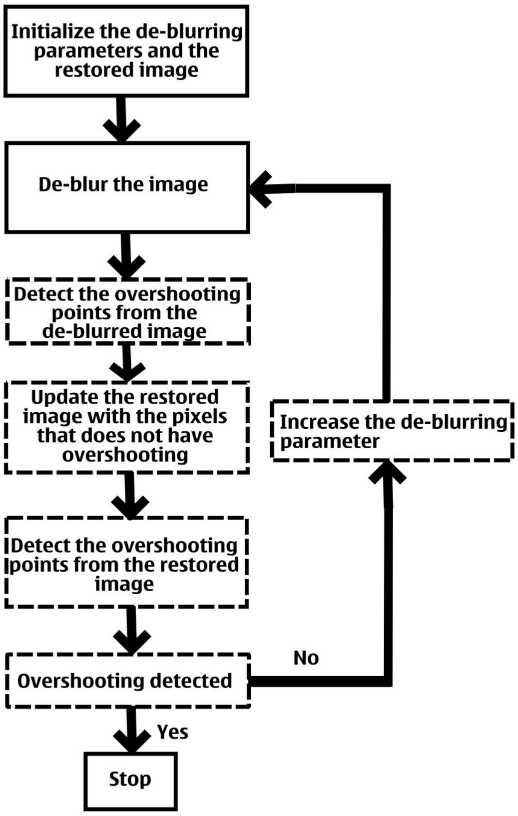

Figure 1: The block diagram of our proposed approach.

the regularization mechanism is to preserve the local monotonicity of the input image. Specifically, pixels where the sign of the local derivative changes, during iterations, represents distorted edges. If such distortions are larger that a predefined level the restoration for those pixels is stopped. A similar mechanism is also implemented for the stopping rule and both methods are detailed in the sequel.

The block diagram of our method is depicted in Fig. 1 where the dashed line blocks refers to our novel approach. Following this block diagram, the iterative restoration method can be described by the following steps:

1. Initialization of the parameters for de-blurring algorithm.

In our approach we have used a simplification of the blur space de-blurring from [1]. Specifically we have used the following setup: $w_{0,i} = 1$ and $w_{1,i} = 0$ for all $i = 1, \ldots, N$. A small de-blurring parameter p was initialized at this step.

2. De-blurring of the observed image using the non-iterative blur space restoration method with the setup from the above step.

The amount of de-blurring is p and the de-blurred image is denoted as $I_{db}(x, y) = I(x, y, v_0 - p)$ (x and y are the horizontal and vertical pixel coordinates).

3. Overshooting detection in de-blurred image. The observed image I and the de-blurred image I_{db} are scanned and the horizontal and vertical differences between adjacent pixels are computed as follows:

$$\begin{aligned}
\epsilon_1(x,y) &= I(x,y) - I(x,y-1),\\
\epsilon_2(x,y) &= I(x,y) - I(x,y+1),\\
\epsilon_3(x,y) &= I(x,y) - I(x-1,y),\\
\epsilon_4(x,y) &= I(x,y) - I(x+1,y),\\
\epsilon_5(x,y) &= I_{db}(x,y) - I_{db}(x,y-1),\\
\epsilon_6(x,y) &= I_{db}(x,y) - I_{db}(x,y+1),\\
\epsilon_7(x,y) &= I_{db}(x,y) - I_{db}(x-1,y),\\
\epsilon_8(x,y) &= I_{db}(x,y) - I_{db}(x+1,y)
\end{aligned} \tag{6}$$

After that, the signs of the corresponding differences computed on I and I_{db} are compared. A pixel for which the signs are not equal contains overshooting. If the amount of overshooting is above a certain threshold the pixel is marked as done for the rest of the iterations.

This step is implemented as follows:

if $sgn\,(\epsilon_1(x,y)) \neq sgn\,(\epsilon_5(x,y))$ **or**
  $sgn\,(\epsilon_2(x,y)) \neq sgn\,(\epsilon_6(x,y))$
  if $abs\,(\epsilon_5(x,y)) \geq T_1$ **or** $abs\,(\epsilon_6(x,y)) \geq T_1$
    $mask(x,y) = 0$
  endif
endif

(7)

where T_1 is a threshold and $abs(x)$ the absolute value of x.

The same procedure as (7) is applied also to the horizontal differences $\epsilon_3(x,y)$, $\epsilon_4(x,y)$, $\epsilon_7(x,y)$, and $\epsilon_8(x,y)$. The overshooting pixels are marked accordingly.

4. Update the restored image. The pixels in the final restored image I_r are updated with the corresponding pixels from the image I_{db}. Here, just the pixels that were not marked as done are updated (pixels with $mask(x,y) \neq 0$).

5. Overshooting detection in the restored image. After updating the restored image, the vertical and horizontal differences between pixels are computed similar to (6).

The signs of the corresponding differences computed on I and on the restored image I_r are compared (for both horizontal and vertical directions) as follows:

if $sgn\,(\epsilon_1(x,y)) \neq sgn\,(\epsilon_9(x,y))$ **or**
$sgn\,(\epsilon_2(x,y)) \neq sgn\,(\epsilon_{10}(x,y))$
$H(x,y) = min\,\{abs(\epsilon_9(x,y)), abs(\epsilon_{10}(x,y))\}$
endif
if $sgn\,(\epsilon_3(x,y)) \neq sgn\,(\epsilon_{11}(x,y))$ **or**
$sgn\,(\epsilon_4(x,y)) \neq sgn\,(\epsilon_{12}(x,y))$
$K(x,y) = min\,\{abs(\epsilon_{11}(x,y)), abs(\epsilon_{12}(x,y))\}$
endif
(8)

where $\epsilon_9(x,y)$, $\epsilon_{10}(x,y)$, $\epsilon_{11}(x,y)$, and $\epsilon_{12}(x,y)$ are computed on the restored image similar to $\epsilon_5(x,y)$, $\epsilon_6(x,y)$, $\epsilon_7(x,y)$, and $\epsilon_8(x,y)$ in (6).

6. Stopping rule. If the maximum value between $H(x,y)$ and $K(x,y)$ is larger than a threshold T_2 the iterative process is stopped. Otherwise, the de-blurring parameter p is increased and the iterative process continue from step 2.

For the two thresholds T_1 and T_2 we have used:

$$T_1 = m_1 \times MAX \quad \text{and} \quad T_2 = m_2 \times MAX \quad (9)$$

where MAX is the maximum pixel value in the observed image and $m_1, m_2 \in [0,1]$.

By (9), we ensure adaptivity of the two thresholds to the dynamic range of the input image. The parameters m_1 and m_2 are set up by the user and define the amount of over emphasis of the details allowed in the restored image (typically values in the interval $[0.1, 0.3]$ shown good restoration performances in all our experiments).

4 Simulation Results

The performance of our approach is compared with the non-iterative approach detailed in [1] for the noisy and noise free input.

For the noise free case, the observed image was obtained by blurring a clean image with a 11×11 Gaussian PSF with variance $v_0 = 2$. The blurred image and the restored images, using our proposed approach and the method in [1], are shown in Fig. 2. For the algorithm introduced in [1], two situations are considered: $p = 2$ (the original blur $v_0 = 2$ was assumed to be known) and $p = 3.5$ (the original blur is erroneous estimated). We notice the overshooting of the details for the non-iterative method with $p = 3.5$ whereas our proposed method provide better restoration result.

In the noisy case, the blurred image was further degraded by adding a zero mean Gaussian distributed noise with variance $\sigma_n^2 = 5$. The observed image and the results of both restoration methods are shown in Fig. 3. The method from [1] was implemented again with $p = 2$ and $p = 3.5$. No noise reduction algorithm was included in the restoration process therefore, in Fig. 3, the effect of noise amplification is evident. For our proposed iterative algorithm the de-blurring step was initialized at $p = 0.1$ and was increased with 0.1 at each iteration. The other parameters of the compared methods were $N = M = 3$, $m_1 = m_2 = 0.2$.

The ISNR values shown in Table 1 also suggest that, for the noisy case, the iterative method provides more robust restoration. For the noise free case the new method has the same performance as the one in [1] with exact knowledge of the blur parameter. From the results shown here, it can be concluded that our iterative method stops near an optimum point such that there is no need to know the amount of blur present in the observed image. Moreover, the regularization mechanism prevents edge overshooting as it can be clearly seen in Fig. 2 and Fig. 3.

Table 1: ISNR for the compared implementations.

Noise	Proposed	Algorithm from [1] (p=2)	Algorithm from [1] (p=3.5)
$\sigma_n^2 = 0$	1.86	1.86	-6.19
$\sigma_n^2 = 5$	-0.16	-8.40	-17.15

5 Conclusions

An iterative algorithm for image restoration in the blur domain is introduced. It does not necessitate knowledge of the blur parameter in the observed image. The proposed algorithm uses a simple stopping rule that finds the optimum number of iterations and a regularization mechanism that controls edge overshooting. Simulations performed on artificially degraded images shown good performance and robustness of the proposed method. The proposed stoping rule and regularization mechanism can be accommodated in several other iterative and non-iterative restoration algorithms.

References:

[1] J. Immerkaer, Use of Blur-Space for Debluring and Edge-Preserving Noise Smoothing, in *IEEE Transactions on Image Processing*, Vol. 10, No. 6, June, 2001, pp: 837-840.

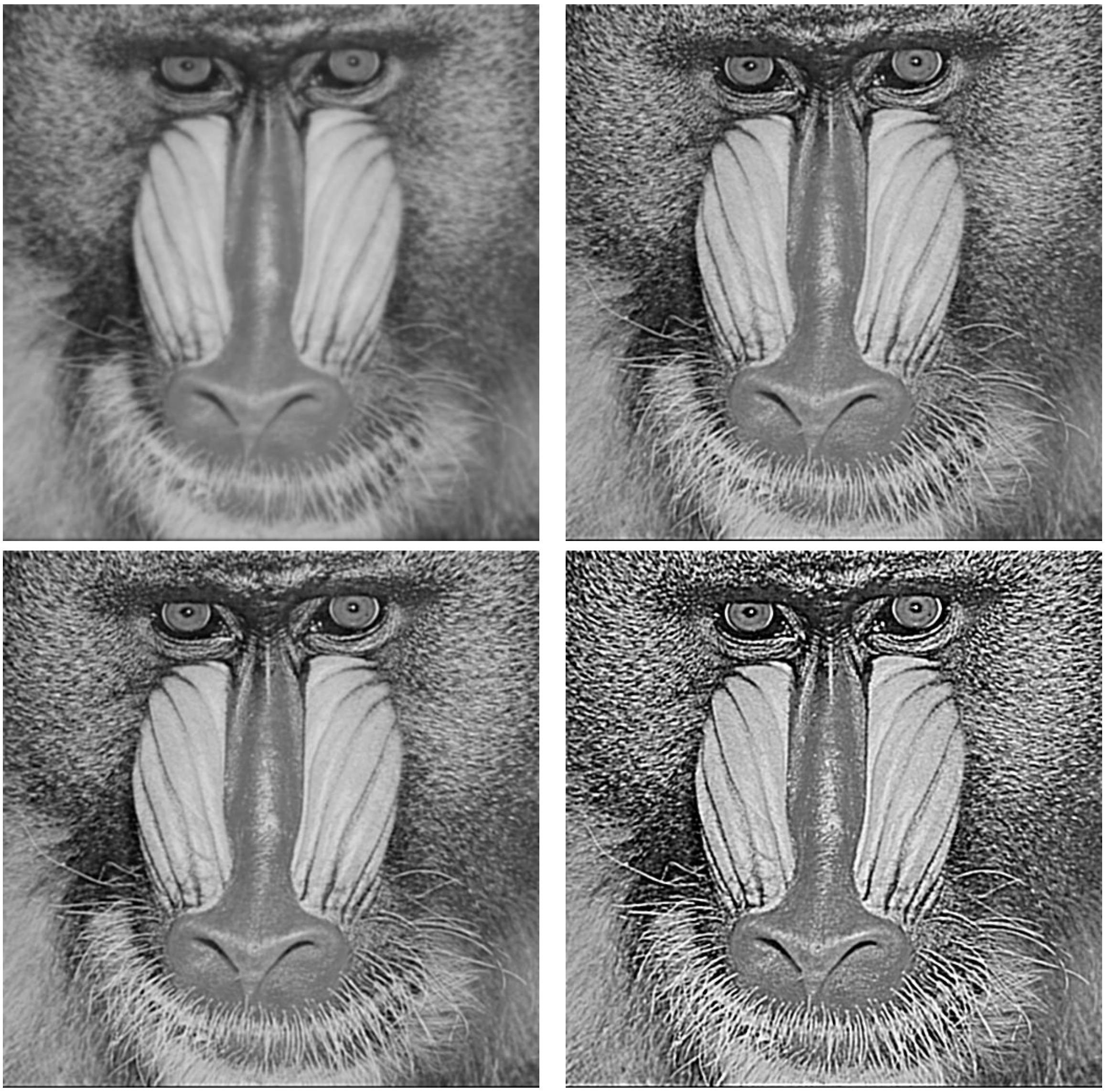

Figure 2: The input blurred image (top left), the result of our approach (top right), the results of the approach from [1] with $p = 2$ (bottom left) and with $p = 3.5$ (bottom right).

[2] A. Bosco, K. Findlater, S. Battiato, A. Castorina, A Noise Reduction Filter for Full Frame Data Imaging Devices, in *IEEE Transactions on Consumer Electronics*, Vol. 49, No. 3, August, 2003, pp: 676-682.

[3] V. Katkovnik, A Multiresolution Nonparametric Regression and Image De-noising, in *Proceedings of IEEE International Conference on Image Processing ICIP 2003*, Barcelona, Spain, September, 2003, pp: 883-887.

[4] L. Liang, Y. Xu, Adaptive Landweber Method to Deblur Images, in *IEEE Signal Processing Letters*, Vol. 10, No. 5, May, 2003, pp: 129-132.

[5] R. Neelamani, H. Choi, R. Baraniuk, ForWaRD: Fourier-wavelet Regularized Deconvolution for Ill-Conditioned Systems, in *IEEE Transactions on Signal Processing*, Vol. 52, No. 2, February, 2004, pp: 418-433.

[6] K. M. Perry, S. J. Reeves, A Practical Stopping Rule for Iterative Signal Restoration, in *IEEE Transactions on Signal Processing*, Vol. 42, No. 7, July, 1994, pp: 1829-1833.

[7] H. J. Trussell, Convergence Criteria for Iterative Restoration Methods, in *IEEE Transaction on Acoustics, Speech and Signal Processing*, Vol. 31, No. 1, February, 1983, pp: 129-136.

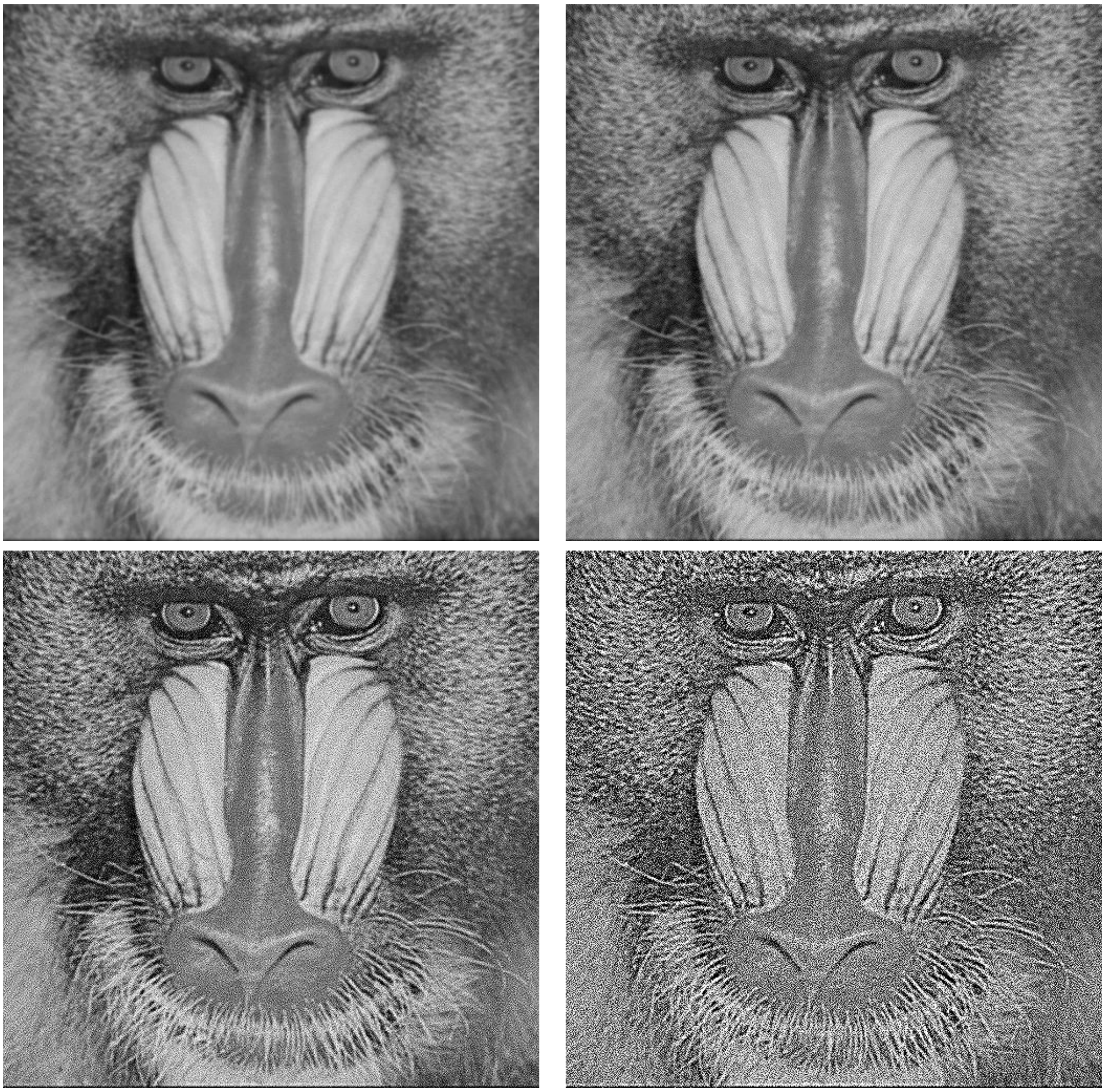

Figure 3: The input blurred noisy image (top left), the result of our approach (top right), the result of the approach in [1] with $p = 2$ (bottom left) and with $p = 3.5$ (bottom right).

[8] A. Rosenfeld, A. C. Kak, *Digital Picture Processing -second edition*, Academic Press, USA, 1982.

[9] W. F. Schreiber, Wire Quality Improvement by Unsharp Masking, in *Pattern Recognition*, Vol. 2, 1970, pp: 117-121.

[10] L. Levi, Unsharp Masking and Related Image Enhancement Techniques, in *Computer Graphics and Image Processing*, Vol. 3, 1974, pp: 163-177.