Filomat 26:3 (2012), 453–465
DOI 10.2298/FIL1203453M

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

Application of the pseudoinverse computation in reconstruction of blurred images

Sladjana Miljković[a], Marko Miladinović[a], Predrag Stanimirović[a], Igor Stojanović[b]

[a] *University of Niš, Department of Mathematics and Informatics, Faculty of Science, Višegradska 33, 18000 Niš, Serbia*
[b] *University of Goce Delcev, Faculty of Computer Science, Macedonia*

Abstract. We present a direct method for removing uniform linear motion blur from images. The method is based on a straightforward construction of the Moore-Penrose inverse of the blurring matrix for a given mathematical model. The computational load of the method is decreased significantly with respect to other competitive methods, while the resolution of the restored images remains at a very high level. The method is implemented in the programming package MATLAB and respective numerical examples are presented.

1. Introduction

Recording images is a very frequent event in everyday human life. However, the recorded images are degraded with respect to the original images. The question of removing these imperfections, in many scientific areas such as satellite imaging, medical imaging, astronomical imaging etc., is of crucial importance. Some of the reasons for having different types of degradations in images are the phenomena described as: blur, noise, geometrical degradations, illumination or color imperfections. A number of outstanding overview articles, journal papers and textbooks deal with the problem of image restoration [1, 3, 13, 16, 17].

In the present article, we investigate the problem of removing the blur from images, caused by a uniform linear motion. Our assumptions are that the linear motion corresponds to an integral number of pixels, and it is aligned with the horizontal (or vertical) sampling. We are concentrated on the usage of the Moore-Penrose inverse solution of a given matrix equation which represents a mathematical model of the uniform linear motion blur.

The methods of image restoration, based on the usage of the Moore-Penrose inverse, have been exploited in many recent papers [6–8]. Several methods for computing the Moore-Penrose inverse have been introduced in [2]. One of the most commonly used methods, is the method of Singular Value Decomposition (SVD). This method is very accurate but also time-intensive since it requires a large amount of computational resources, especially in case of large matrices. An algorithm for fast computation of the Moore-Penrose inverse is also presented in the recent work of P. Courrieu [9]. Courrieu's algorithm is

2010 *Mathematics Subject Classification*. Primary 68U10; Secondary 94A08
Keywords. Image restoration, X-ray, motion blur, matrix equation, Moore-Penrose inverse
Received: 29 September 2011; Accepted: 01 November 2011
Communicated by Dragan S. Djordjević
Research supported by the Serbian Ministry of Science, research project No. 174013
Email addresses: slagana256@yahoo.com (Sladjana Miljković), markomiladinovic@gmail.com (Marko Miladinović), pecko@pmf.ni.ac.rs (Predrag Stanimirović), igor.stojanovik@ugd.edu.mk (Igor Stojanović)

based on the reverse order law for matrix pseudoinverse (eq. 3.2 from [19]), and on the full-rank Cholesky factorization of possibly singular symmetric positive matrices. Another very fast and reliable method for estimation of the Moore-Penrose inverse of full rank rectangular matrices is given by V. Katsikis and D. Pappas [14]. The method uses a special type of tensor product of two vectors.

All methods for computing the Moore-Penrose inverse, mentioned above, are either iterative or use some kind of matrix factorization. The method we propose, explore the structure of the degradation matrix of the model and generates the Moore-Penrose inverse analytically, by means of a set of rules. The motivation behind, is the very proper structure of the matrix which participates as a degradation system in the image formation process. The introduced method is very fast, which is its main advantage. On the other hand, the main disadvantage of the proposed method is its limitation to uniform linear motion blur degradations. The presented numerical results claim the expected decrease in CPU time.

The paper is organized as follows. In the second section we present the ideas behind the processes of image restoration which are based on the usage of the Moore-Penrose inverse. In the third section we present a new method for reconstruction of blurred images, by generating the Moore-Penrose inverse of the model matrix analytically. Finally in the last section, by reporting numerical results, we observe certain enhancement in the computational time and the Improvement in Signal to Noise Ration (*ISNR*), compared to other standard methods for image restoration.

2. Preliminaries

The matrix $A^\dagger$ which satisfies the following properties

$$AA^\dagger A = A, \quad A^\dagger AA^\dagger = A^\dagger, \quad (AA^\dagger)^T = AA^\dagger, \quad (A^\dagger A)^T = A^\dagger A,$$

is called the Moore-Penrose inverse of the matrix A.

This section refers to the formation of the mathematical model that reflects the process of removing the blur in images, which is caused by a uniform linear motion as well as the matrix pseudoinverse solution of the problem.

Suppose that the matrix $F \in \mathbb{R}^{r\times m}$ corresponds to the original image with picture elements $f_{i,j}$, $i = 1, \ldots, r$, $j = 1, \ldots, m$ and $G \in \mathbb{R}^{r\times m}$ with pixels $g_{i,j}$, $i = 1, \ldots, r$, $j = 1, \ldots, m$, is the matrix corresponding to the degraded image. Let l be an integer indicating the length of the linear motion blur in pixels and $n = m + l - 1$. In practice the degradation (index l) is rarely known exactly, so that it must be identified from the blurred image itself. To estimate the index l, two different cepstral methods can be used: one dimensional or two dimensional cepstral method [15]. To avoid the problem when the information from the exact image spills over the edges of the recorded image, we supplement the original image with boundary pixels that best reflect the original scene. Without any confusion we are using the same symbol F for the enlarged original image (matrix) with remark that F now becomes a matrix of dimensions $r \times n$. First, we suppose that the blurring is a horizontal phenomenon. Let us denote the degradation matrix by $H \in \mathbb{R}^{m\times n}$. For each row f_i of the matrix F and the respective row g_i of the matrix G we consider an equation of the form

$$g_i^T = Hf_i^T, \ \ g_i^T \in \mathbb{R}^m, \ f_i^T \in \mathbb{R}^n, \ H \in \mathbb{R}^{m\times n}. \tag{1}$$

The objective is to estimate the original image F, row by row, using the corresponding rows of the known blurred image G and a priori knowledge of the degradation phenomenon H.

Equation (1) can be written in the matrix form as

$$G = \left(HF^T\right)^T = FH^T, \ \ G \in \mathbb{R}^{r\times m}, \ H \in \mathbb{R}^{m\times n}, \ F \in \mathbb{R}^{r\times n}. \tag{2}$$

There is an infinite number of exact solutions for f that satisfy the equation (1). But, only the Moore-Penrose inverse solution solves uniquely the next minimization problem (see, for example [2]):

$$\min \|f\|_2, \text{ subject to } \min \|Hf - g\|_2. \tag{3}$$

The unique vector $\tilde{f}$ satisfying (3) represents a row of the restored image [5–8], and it is defined by

$$\tilde{f} = H^{\dagger} g. \tag{4}$$

The matrix form of the equation (4) i.e., the restored image $\tilde{F}$ is given by

$$\tilde{F} = G(H^{\dagger})^{T}. \tag{5}$$

The matrix $\tilde{F}$ defined in (5) is the minimum-norm least-squares solution of the matrix equation (2).

The matrix equation which characterizes the vertical motion blurring process is given by

$$G = HF, \;\; G \in \mathbb{R}^{r\times m}, \; H \in \mathbb{R}^{r\times n}, \; F \in \mathbb{R}^{n\times m}, \; n = r + l - 1. \tag{6}$$

The corresponding restored image can be computed using the Moore-Penrose inverse by the following formula

$$\tilde{F} = H^{\dagger} G. \tag{7}$$

We assume that the blurring is a local phenomenon, spatially invariant as well that the imaging process captures all light and no additional noise is included. Taking into consideration the given requirements, the degradation matrix of the blurring process reduces to a matrix $H = \text{toeplitz}(h^1, h_1)$. The matrix H is non-symmetric Toepltiz matrix consisting of m rows and $n = m + l - 1$ columns, determined by its first column $h^1 = (h_{i,1})_{i=1}^{m}$ and its first row $h_1 = (h_{1,j})_{j=1}^{n}$ as follows:

$$h_{i,1} = \begin{cases} 1/l, & i = 1, \\ 0, & i = 2, \ldots, m, \end{cases} \quad \text{and} \quad h_{1,j} = \begin{cases} 1/l, & j = 1, \ldots, l, \\ 0, & j = l+1, \ldots, n. \end{cases} \tag{8}$$

An arbitrary ith row of the blurred image can be expressed using the ith row of the original image extended with the boundary pixels as

$$\begin{bmatrix} g_{i,1} \\ g_{i,2} \\ g_{i,3} \\ \vdots \\ g_{i,m} \end{bmatrix} = \begin{bmatrix} \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} & 0 & 0 & 0\ldots & 0 \\ 0 & \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} & 0 & 0\ldots & 0 \\ 0 & 0 & \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} & 0\ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} \end{bmatrix} \begin{bmatrix} f_{i,1} \\ f_{i,2} \\ f_{i,3} \\ \vdots \\ f_{i,n} \end{bmatrix}, \tag{9}$$

where $l - 1$ elements of the vector f_i, are not actually the pixels from the original scene; rather they are *boundary pixels*. How many boundary pixels will be added above the vector f depends of the nature and direction of the movement. However, the rest of them, i.e., $l - 1$ minus the number of pixels added above the vector f, would present the boundary pixels right of the horizontal line, and are added below the vector f [11].

The process of non–uniform blurring assumes that the blurring of the columns in the image is independent with respect to the blurring of the rows. In this case two matrices participate in the formation of the process and the relation between the original and the blurred image can be displayed with the following relation

$$G = H_c F H_r^{T}, \;\; G \in \mathbb{R}^{m_1\times m_2}, \; H_c \in \mathbb{R}^{m_1\times r}, \; F \in \mathbb{R}^{r\times n}, \; H_r \in \mathbb{R}^{m_2\times n}, \tag{10}$$

where $n = m_2 + l_r - 1$, $r = m_1 + l_c - 1$, l_r is the length of the horizontal blurring in pixels and l_c is the length of the vertical blurring in pixels. In this case, the Moore-Penrose solution of the system (10) is given by

$$\tilde{F} = H_c^{\dagger} G (H_r^{T})^{\dagger}. \tag{11}$$

3. New image restoration method

First, we define a method of image restoration in the case when the number of columns of the image, enlarged by boundary pixels, can be divided by an appropriate number of blurring pixels, i.e., when the equality $n = l \cdot p$ holds. We show that in this case the Moore-Penrose inverse $H^{\dagger}$ can be generated analytically, without any iterations. Later, we generalize the method to the case when the dimension n is arbitrary.

Let us suppose that

$$n = m + l - 1 = l \cdot p,$$

where the number of blurring pixels l is a positive integer. In the rest of the section, we construct the matrix $\widetilde{H} = [\widetilde{h}_{ij}], i = 1, \ldots, n, j = 1, \ldots, m$ and show that it is actually the Moore-Penrose inverse of the degradation matrix H.

All elements of the matrix $\widetilde{H}$, excluding zero elements, can be represented by the following two sequences:

$$x_k = -\frac{l}{n}(m - l(k-1) - 1) = -\frac{m - l(k-1) - 1}{p}, \quad k = 1, 2, \ldots, p-1,$$
$$y_k = \frac{l}{n}(m - l(k-1)) = \frac{m - l(k-1)}{p}, \qquad k = 1, 2, \ldots, p.$$

Additionally, we put $z = y_p = \frac{1}{p}$.

The general layout of the matrix $\widetilde{H} \in \mathbb{C}^{n \times n}$ in the case $n = m + l - 1 = l \cdot p$ is given in Figure 3.1.

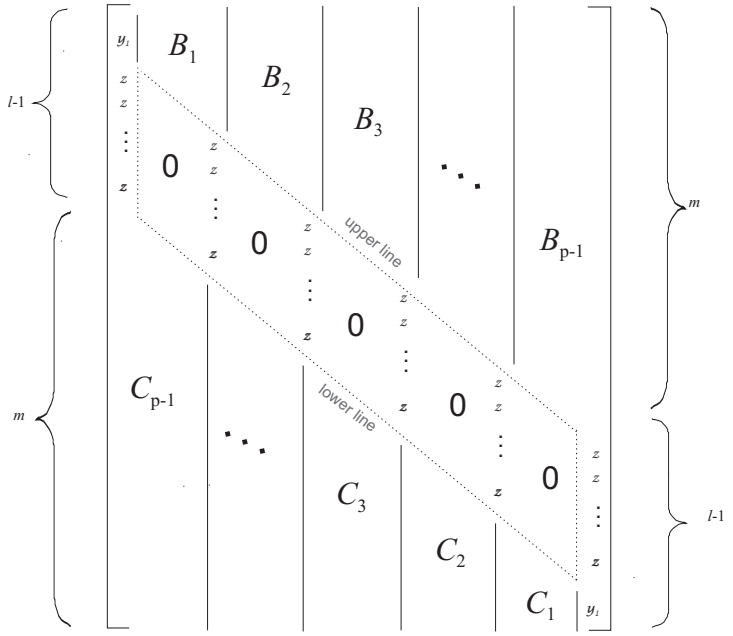

Figure 3.1. General layout of the matrix $H^{\dagger}$ in the case $n = m + l - 1 = l \cdot p$.

The parallelogram between the blocks B_k and C_k is called *zero layer*. The line denoted by *upper line* refers to the elements above the zero layer of the matrix $\widetilde{H}$ which actually constitute the diagonal of the square $m \times m$ matrix formed from the first m rows of the matrix $\widetilde{H}$. Similarly, *lower line* refers to the elements below the zero layer of the matrix $\widetilde{H}$ which constitute the diagonal of the square $m \times m$ matrix formed from the last m rows of the matrix $\widetilde{H}$. The upper line, the lower line, the first l elements of the first row of $\widetilde{H}$ and the last l elements of the last column of $\widetilde{H}$ will be denominated as *sides of the zero layer*.

Further, we preview the structure of the blocks B_k, $k = 1, \ldots, p-1$. Each block B_k can be represented via appropriate block P_k of the following form:

$$P_k = \begin{vmatrix} 0 & & & & & & \\ 0 & 0 & & & & & \\ 0 & 0 & 0 & & & & \\ \vdots & \ddots & \ddots & \ddots & & & \\ 0 & 0 & \ldots & \ldots & 0 & & \\ x_k & 0 & 0 & \ldots & \ldots & 0 & \\ -x_k & x_k & 0 & 0 & \ldots & 0 & z \\ & -x_k & x_k & 0 & 0 & \vdots & z \\ & & \ddots & \ddots & \ddots & & \vdots \\ & & & \ddots & \ddots & 0 & \vdots \\ & & & & -x_k & x_k & z \\ & & & & & -x_k & z + x_k \\ & & & & & & y_{k+1} \end{vmatrix}$$

Zero parallelogram blocks are of dimensions $(l-2) \times (l-1)$. The block B_k is obtained by pasting up k times the block P_k, from the bottom upwards, and then from the resulting block we cut by horizontal line the most upper triangle which has a vertical side containing the upper $l-2$ elements of the block P_k. The missing element of the resulting block B_k is filled by the value of y_{k+1}. The three steps in the formation process of the blocks B_k are presented in Figure 3.2.

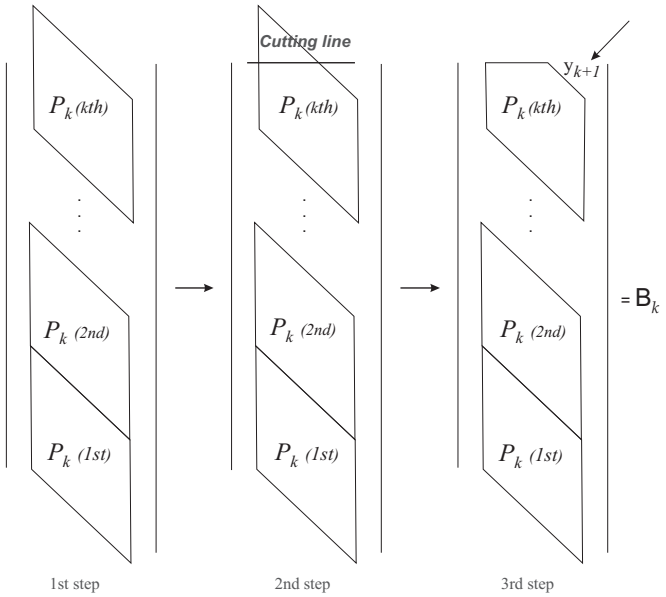

Figure 3.2. Formation of the block B_i from the block P_i.

In order to write the analytical form of the matrix $\widetilde{H}$ we will use the following notation. Let q_i, q_j, r_i and r_j be integers such that for a given ith row and jth column hold $i = q_i l + r_i$ and $j = q_j l + r_j$. From the previous considerations it is clear that: $i \leq j$ if and only if $\widetilde{h}_{ij} \in B_k$, $k = 1, 2, \ldots, p-1$. Similarly, $i \geq l$ and $i > j$ if and only if $\widetilde{h}_{ij} \in C_k$, $k = 1, 2, \ldots, p-1$. Taking this into account, we present the analytical form of the matrix $\widetilde{H}$ as follows

$$\widetilde{h}_{ij} = \begin{cases} y_{q_j+1}, & i \le j,\ r_j = 1,\ r_i = 1, \\ z + x_{q_j}, & i \le j,\ r_j = 1,\ r_i = 0, \\ (-1)^{d+1} x_{q_{j-1}+1}, & i \le j,\ r_j \ne 1,\ q_j \ge q_i,\ (r_{i-j} = 0 \text{ or } r_{i-j} = l-1) \\ \\ z + x_{p-q_j-1}, & i \ge l,\ i > j,\ r_j = 1,\ r_i = 1, \\ y_{p-q_j}, & i \ge l,\ i > j,\ r_j = 1,\ r_i = 0, \\ (-1)^{d} x_{p-q_{j-1}-1}, & i \ge l,\ i > j,\ r_j \ne 1,\ q_j \le q_i,\ (r_{i-j} = 0 \text{ or } r_{i-j} = l-1) \\ \\ z, & r_j = 1,\ r_i \ne 0,\ r_i \ne 1 \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

where d is 0 if $r_{i-j} = 0$ and $d = 1$ otherwise. The first case in (12) gives the elements that are not equal to zero or z of the blocks B_k, $k = 0, \dots, p-1$ plus y_1 from the first column. The second case produces the elements that are not equal to zero or z of the blocks C_k, $k = 0, \dots, p-1$ plus y_1 from the last column. In order to store the whole matrix we need only to store $2p-1$ elements which is lower than n in the case $l > 2$.

To illustrate our description we give the full form of the matrix $\widetilde{H}$ observing the case $p = 4$.

$$\left[\begin{array}{cccccc|cccccc|cccccc}
y_1 & x_1 & & & & y_2 & x_2 & & & & y_3 & x_3 & & & & y_4 \\
z & -x_1 & x_1 & & & z & -x_2 & x_2 & & & z & -x_3 & x_3 & & & z \\
z & & -x_1 & x_1 & & z & & -x_2 & x_2 & & z & & -x_3 & x_3 & & z \\
\vdots & & & \ddots & \ddots & \vdots & & & \ddots & \ddots & \vdots & & & \ddots & \ddots & \vdots \\
z & & & & -x_1 & z & & & & -x_2 & z & & & & -x_3 & z \\
y_4 & & & & -x_1 & z + x_1 & & & & -x_2 & z + x_2 & & & & -x_3 & z + x_3 \\
\hline
z + x_3 & -x_3 & & & & y_2 & x_2 & & & & y_3 & x_3 & & & & y_4 \\
z & x_3 & -x_3 & & & z & -x_2 & x_2 & & & z & -x_3 & x_3 & & & z \\
\vdots & & \ddots & \ddots & & \vdots & & -x_2 & x_2 & & z & & -x_3 & x_3 & & z \\
z & & & x_3 & -x_3 & z & & & \ddots & \ddots & \vdots & & & \ddots & \ddots & \vdots \\
z & & & & x_3 & z & & & & -x_2 & z & & & & -x_3 & z \\
y_4 & & & & x_3 & y_3 & & & & -x_2 & z + x_2 & & & & -x_3 & z + x_3 \\
\hline
z + x_3 & -x_3 & & & & z + x_2 & -x_2 & & & & y_3 & x_3 & & & & y_4 \\
z & x_3 & -x_3 & & & z & x_2 & -x_2 & & & z & -x_3 & x_3 & & & z \\
\vdots & & \ddots & \ddots & & \vdots & & \ddots & \ddots & & \vdots & & -x_3 & x_3 & & z \\
z & & & x_3 & -x_3 & z & & & x_2 & -x_2 & z & & & \ddots & \ddots & \vdots \\
z & & & & x_3 & z & & & & x_2 & z & & & & -x_3 & z \\
y_4 & & & & x_3 & y_3 & & & & x_2 & y_2 & & & & -x_3 & z + x_3 \\
\hline
z + x_3 & -x_3 & & & & z + x_2 & -x_2 & & & & z + x_1 & -x_1 & & & & y_4 \\
z & x_3 & -x_3 & & & z & x_2 & -x_2 & & & z & x_1 & -x_1 & & & z \\
\vdots & & \ddots & \ddots & & \vdots & & \ddots & \ddots & & \vdots & & \ddots & \ddots & & \vdots \\
z & & & x_3 & -x_3 & z & & & x_2 & -x_2 & z & & & x_1 & -x_1 & z \\
z & & & & x_3 & z & & & & x_2 & z & & & & x_1 & z \\
y_4 & & & & x_3 & y_3 & & & & x_2 & y_2 & & & & x_1 & y_1
\end{array}\right].$$

In order to verify that the matrix $\widetilde{H}$ is actually the Moore-Penrose inverse of the matrix H, i.e., that $\widetilde{H} = H^{\dagger}$ we will use the following two lemmas:

Lemma 3.1. *The equality $H\widetilde{H} = I$ holds for the matrix $\widetilde{H}$ given by* (12).

Proof. Each row of the matrix H contains l non zero constant elements equal to $1/l$, so that it is obvious that the elements of the matrix $H\widetilde{H}$ are

$$(H\widetilde{H})_{ij} = \frac{1}{l}\sum_{s=i}^{i+l-1}\widetilde{h}_{sj}, \;\; i = 1,\dots,m \;\text{ and }\; j = 1,\dots,m.$$

Therefore, we need to explore the properties of jth column of the matrix $\widetilde{H}$, $j = 1,\dots,m$, i.e., the sums of its l consecutive elements. For this reason, from the representation of the matrix $\widetilde{H}$ given by (12) we distinguish two different cases:

1 case : $r_j = 1$.

Each set of l consecutive elements contains only two elements different from z. If both of them are above the zero layer their sum is $y_{q_j+1} + (l-1)z + x_{q_j} = 0$. If both of them are below the zero layer their sum is $x_{p-q_j-1} + (l-1)z + y_{p-q_j} = 0$. Otherwise, if one of them is above the zero layer and the other one is below the zero layer, then their sum is $y_{q_j+1} + y_{p-q_j} = l$. Unfortunately, as we mentioned before, the last is the case only when $i = j$.

2 case : $r_j \neq 1$.

Each set of l consecutive elements contains only two non zero elements. If those two elements are above the zero layer or both of them are below the zero layer then it is obvious that their sum is 0. If one of them is above the zero layer and the other one is below the zero layer, thus $i = j$, their sum is $-x_{q_{j-1}+1} - x_{p-q_{j-1}-1}$, which by easy calculations can be shown that equals l.

So finally for the both cases we have

$$(H\widetilde{H})_{ij} = \frac{1}{l}\sum_{s=i}^{i+l-1}\widetilde{h}_{sj} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}, \;\; i = 1,\dots m, \;\; j = 1,\dots m.$$

i.e. $H\widetilde{H} = I$. □

Lemma 3.2. *The equality $(\widetilde{H}H)^T = \widetilde{H}H$ holds for the matrix $\widetilde{H}$ given by* (12).

Proof. For a given $i = 1,\dots,n$ and $j = 1,\dots,n$, we should show that $(\widetilde{H}H)_{ij} = (\widetilde{H}H)_{ji}$. The elements of the matrix $\widetilde{H}H$ can be presented as

$$(\widetilde{H}H)_{ij} = \frac{1}{l}\sum_{s=\max\{1,j-l+1\}}^{\min\{j,m\}}\widetilde{h}_{is}, \;\; i = 1,\dots,n \;\text{ and }\; j = 1,\dots,n. \tag{13}$$

Let us denote by $s = \min\{j,m\} - \max\{1, j-l+1\}$.

Consequently, we should show that the sum of s consecutive elements in the *ith* row of the matrix $\widetilde{H}H$ where the last element is in the jth column; equals the sum s consecutive elements in the *jth* row of the matrix $\widetilde{H}H$, where the last element is in the ith column. So the case when $i = j$ is clear, actually, for a given i these elements actually present the sum of the s consecutive elements that belong to the zero layer as well as his sides. We continue with the opposite case when $i \neq j$.

Let us explore the properties of each row i of the matrix $\widetilde{H}$, $i = 1,\dots,n$. First we recall that if $i \leq j$ that means that $\widetilde{h}_{ij}$ is either y_1 or belongs in a block B_k, $k = 1,\dots,p-1$. And, if $i > l$ and $i > j$ that means that $\widetilde{h}_{ij}$ is either y_1 or belongs in a block C_k.

1 case : $i < j,\ r_i = 1$ and $r_j \neq 1$ $(i = 1, \ldots, n-1$ and $j = 1, \ldots, n)$.

From (13) and definition of the matrix $\widetilde{H}$ we have

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(y_{q_j+1} + x_{q_j+1}) = \frac{z}{l}.$$

That means that the sum of each s consecutive elements equals z, if the elements are in the $(kl+1)st$ row, $k = 0, \ldots, p-1$, and the last element is not in the $(kl+1)st$ column, $k = 0, \ldots, p-1$. Also since $i < j$ the last element is above the diagonal of the square matrix constituted of the first m rows of $\widetilde{H}$.

We need to compare these values with the values of $(\widetilde{H}H)_{ji}$. For this situation we analyze the opposite case i.e. we interchange the conditions for i and j. Suppose,

$$i > j,\ r_i \neq 1 \text{ and } r_j = 1 \ \ (i = 1, \ldots, n \text{ and } j = 1, \ldots, n-1).$$

From here we continue with two different possibilities, denoted by $a1$ and $a2$.

a1: if $r_i = 0$ then

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(-x_{q_j} + z + x_{q_j}) = \frac{z}{l}.$$

a2: if $r_i \neq 0$ then

$$(\widetilde{H}H)_{ij} = \frac{1}{l}\begin{cases} z, & j = 1 \\ z + x_{p-q_j} - x_{p-q_j} = z, & j > 1. \end{cases}$$

Thus the first case is completed i.e. if $i < j,\ r_i = 1$ and $r_j \neq 1$ $(i = 1, \ldots, n-1$ and $j = 1, \ldots, n)$ then $(\widetilde{H}H)_{ij} = (\widetilde{H}H)_{ji}$.

2 case : $i < j,\ r_i = 1$ and $r_j = 1$ $(i = 1, \ldots, m-1$ and $j = 1, \ldots, m)$ then

$$(\widetilde{H}H)_{ij} = y_{q_j+1} + x_{q_j} = z(1-l).$$

Note: Since $m = l(p-1)+1$ and $n = m + l - 1$, if $j > m$ it follows that $r_j \neq 1$.

If the conditions $i > j,\ r_i = 1$ and $r_j = 1$ $(i = 1, \ldots, m$ and $j = 1, \ldots, m-1)$ are satisfied we obtain

$$(\widetilde{H}H)_{ij} = \begin{cases} z + x_{p-1} = z(1-l), & j = 1 \\ z + x_{p-q_j-1} - x_{p-q_j} = z(1-l), & j \neq 1, \end{cases}$$

which completes the proof in the case 2.

3 case : $i < j,\ r_i \neq 1$ and $r_j = 1$ $(i = 1, \ldots, m-1$ and $j = 1, \ldots, m)$ then

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(-x_{q_j} + z + x_{q_j}) = \frac{z}{l}.$$

Under the assumptions $i > j,\ r_i = 1$ and $r_j \neq 1$ $(i = 1, \ldots, m$ and $j = 1, \ldots, m-1)$ one can verify

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(z + x_{p-q_j-1} - x_{p-q_j-1}) = \frac{z}{l},$$

so that the verification of the statement in case 3 is completed.

4 case : $i < j,\ r_i \neq 1$ and $r_j \neq 1$ $(i = 1, \ldots, n-1$ and $j = 1, \ldots, n)$. Similarly as in the previous cases, after considering several possibilities, one can derive the following

$$(\widetilde{H}H)_{ij} = \frac{1}{l}\begin{cases} z(l-1), & r_i = r_j,\ i \leq m \\ z, & \text{otherwise.} \end{cases}$$

Under the opposite assumptions $i > j,\ r_i \neq 1$ and $r_j \neq 1$ $(i = 1, \ldots, n$ and $j = 1, \ldots, n-1)$ we get

$$(\widetilde{H}H)_{ij} = \frac{1}{l}\begin{cases} z(l-1), & r_i = r_j,\ j \leq m \\ z, & \text{otherwise,} \end{cases}$$

and the proof is completed. □

Theorem 3.1. *The matrix $\widetilde{H}$ given by* (12) *is the Moore-Penrose inverse of the matrix H.*

Proof. Since the matrix H is full row rank matrix its Moore-Penrose inverse is its right inverse. From this fact and from the previous two lemmas follows the proof of the theorem. □

Algorithm 3.1 Image deblurring method (MP method).

Require: The blurred image G of dimensions $r \times m$ defined in the blurring process (9).

1: If $m + l - 1 \mod l \neq 0$ then add $l * quotient(m + l - 1, l) + l - m$ boundary pixels, else add $l - 1$ boundary pixels.
2: Compute the matrix $H^{\dagger}$ according to the formula (12)
3: Apply formula (5)
4: Return $\widetilde{F}$.

4. Experimental results

In this section we present numerical results which are obtained by testing the method proposed in Algorithm 3.1 (MP method) on X-ray images. In order to confirm the efficiency, we compared our method with three recently announced methods for computing the Moore-Penrose inverse of the matrix H. Summarizing, the following four methods are compared:

1. The *MP* method,
2. *Pappas*1 method, defined by the MATLAB function `ginv.m` from [14],
3. *Pappas*2 method, defined by the MATLAB function `qrginv.m` from [8],
4. *Courrieu* method from [9].

The experiments are done using *Matlab* programming language [12] on an Intel(R) CPU T2130 @ 1.86 GHz 1.87 GHz 32-bit system with 2 GB of RAM memory. Tests are made for several images of dimensions $r \times m$. The index l that takes values between 10 and 100 is the varying parameter for a given image.

Also we compared the efficiency of four different strategies of image restoration: the approach based on the Moore-Penrose inverse, the Wiener filter (*WF*), the constrained least-squares (*CLS*) filter, and the Lucy-Richardson (*LR*) algorithm. For the implementation of the Wiener filter, the constrained least-squares filter, and Lucy-Richardson algorithm we used built–in functions from the *Matlab* package [10].

In image restoration the quality enhancement of the restored image over the recorded blurred one is evaluated by the signal-to-noise ratio improvement (*ISNR*). The *ISNR* of the recorded (blurred) image is defined in decibels as follows:

$$ISNR = 10 \log_{10} \left(\frac{\sum_{n1,n2} (G(n_1, n_2) - F(n_1, n_2))^2}{\sum_{n1,n2} (\tilde{F}(n_1, n_2) - F(n_1, n_2))^2} \right). \tag{14}$$

Figure 4.1 presents one practical example for restoring blurred X-ray image. The image is taken from the results obtained from Google Image search with the keywords "X-ray image". The picture in Figure 4.1 called *Original image* shows the original X-ray image. The image is divided into $r = 843$ rows and $m = 1050$ columns. To prevent loosing information from the boundaries of the image, we assumed zero boundary conditions, which implies that values of the pixels of the original image F outside the domain of consideration are zero (black). This choice is natural for X-ray images since the background of these images is black. The picture named as *Degraded image* presents the degraded X-ray image with a uniform horizontal motion of length $l = 80$. The pictures named as *Moore-Penrose Inverse*, *Wiener Restored Image*, *Constrained LS Restored Image* and *Lucy-Richardson Restored Image* denote the images obtained after the application of the corresponding restoration algorithms. From Figure 4.1, it is clear that the *MP* method produces the best result.

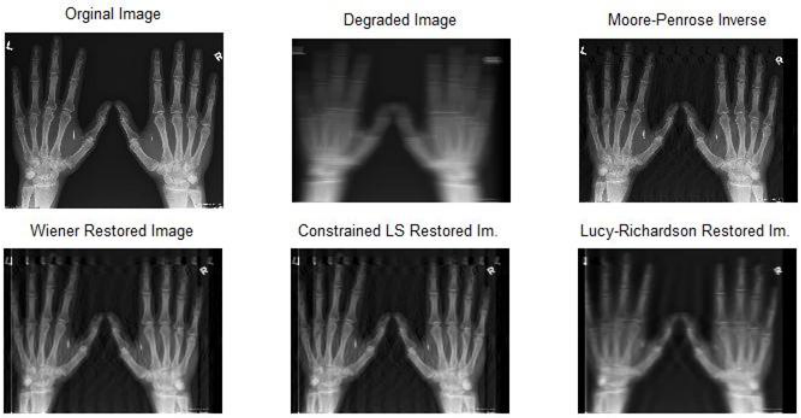

Figure 4.1. Removal of blur, caused by a uniform horizontal motion.

The difference in quality of the restored images among the three methods (*WF*, *CLS* and *LR*) is insignificant, and can hardly be seen by human eye. For this reason, the *ISNR* is applied in order to compare the quality of the restored images. Figure 4.2 (left) shows the corresponding *ISNR* value of the restored images as a function of l for the *MP* method and the other mentioned classical methods. This figure illustrates that the *MP* method for image restoration has the best quality of the restored image with respect to the other methods.

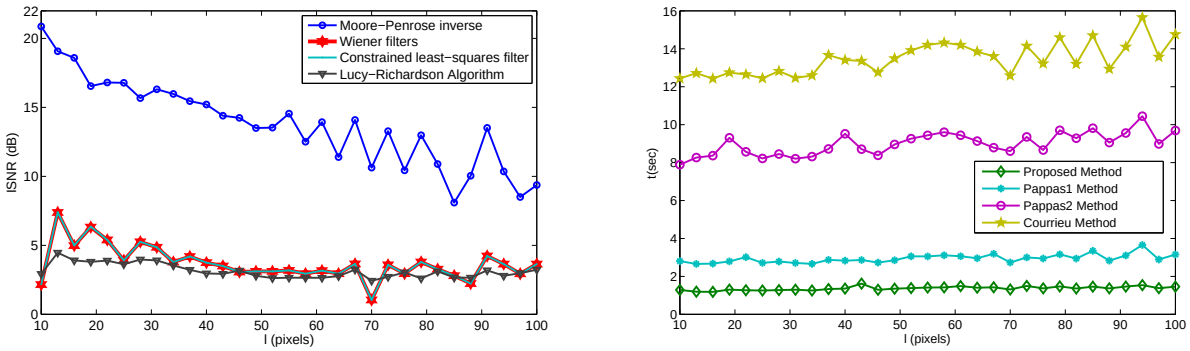

Figure 4.2. (left) Improvement in signal-to-noise-ratio vs. length of the blurring process
(right) Computational time vs. length of the blurring process.

As we have already mentioned, the main advantage of the proposed method for computing the Moore-Penrose inverse, is the time required to obtain the restored image compared to other methods for computing the Moore-Penrose inverse. Figure 4.2 (right) shows the corresponding computational time, denoted by *t(sec)*, as a function of $l < 100$ pixels for the *MP* method and the other three considered methods for computing the Moore–Penrose inverse. Obviously, the minimal computational time is reached by the *MP* method.

Additionally, we compare the method based on the usage of the Moore-Penrose inverse with another image restoration method, which uses a least-squares solution of the following system, arising from (9),

$$g_{i,j} = \frac{1}{l}\sum_{k=0}^{l-1} f_{i,j+k},\ i = 1,\dots,n, j = 1,\dots,m, \tag{15}$$

where $n = m + l - 1$. The corresponding solution is derived by using the standard `Matlab` function `mrdivide()`, and it will be called *LS solution* in the test examples. The function `mrdivide(B,A)` (or equivalently

`B/A`) performs matrix right division (forward slash) [18]. Matrices B and A must have the same number of columns. If A is a square matrix, `B/A` is roughly the same as `B*inv(A)`. If A is an $n \times n$ matrix and B is a row vector with n elements, or a matrix with several such rows, then `X = B/A` is the solution of the equation $XA = B$ computed by using Gaussian elimination with partial pivoting. If B is an $m \times n$ matrix with $m \sim= n$ and A is a column vector with m components, or a matrix with several such columns, then `X = B/A` is a solution in the least-squares sense to the under- or overdetermined system of equations $XA = B$. In other words, X minimizes `norm(A*X - B)`. The rank k of A is determined from the QR decomposition with column pivoting. The computed solution X has at most k nonzero elements per column. If $k < n$, this is usually not the same solution as `X = B*pinv(A)`, which returns a least-squares solution with the smallest norm $\|X\|$. The presented results show that using the pseudo inverse approach leads to better improvements than solving the system directly. The respective comparison is illustrated in the next figure. The left image in Figure 4.3 shows the restoration determined by using the solution of the system (15), while the right image shows the restoration based on the usage of the Moore-Penrose inverse.

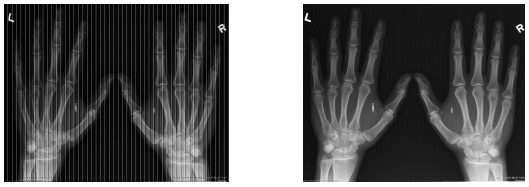

Figure 4.3. a) Restoration arising from the *LS* solution; b) Restoration based on the Moore-Penrose inverse

The next two examples refer to the case when, in the inception, the image is degraded by including an image noise and later it is followed by a uniform linear blur. The corresponding mathematical model generalizes the horizontal blurring process presented with (2) and it becomes

$$G_N = (F + N)H^T = F_N H^T, \tag{16}$$

where N is an additive noise and G_N is the blurred noisy image. To obtain approximation of the original image, we apply two steps:
1. Calculate the restored matrix F_N by using (5), to produce $\tilde{F}_N = G_N(H^\dagger)^T$;
2. Obtain the restored image $\tilde{F}$ by applying filtering process on the image $\tilde{F}_N$ obtained in Step 1.

Similarly, we can formulate a process in the case of having two ways degraded image (with noise and vertical blur). The noisy image, the blurred noisy image and the restored images obtained by using different methods are presented in Figure 4.4.

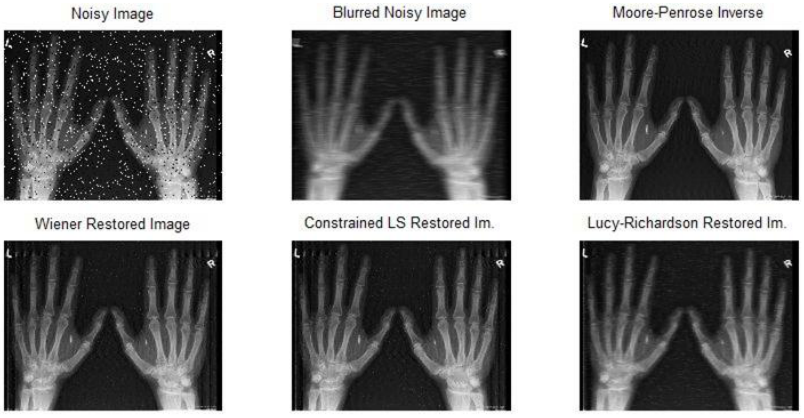

Figure 4.4. Removal of blur, caused by noise and uniform horizontal motion with $r = 843$, $n = 1050$ and $l = 40$.

The results for *ISNR* and the peak signal-to-noise ratio (*PSNR*) [4] for the original image given in Figure 4.1, are presented in Figure 4.5. The original image is degraded in two ways: by "salt and pepper" (white and black) noise with noise density of 0.03 and after that it is blurred by a uniform linear motion with l pixels. The 2-D median filtering is used for the image restoration. The image restoration procedures based on the Moore-Penrose inverse, applied to images degraded by both the motion blur and noise, are more reliable and accurate compared to other image restoration methods.

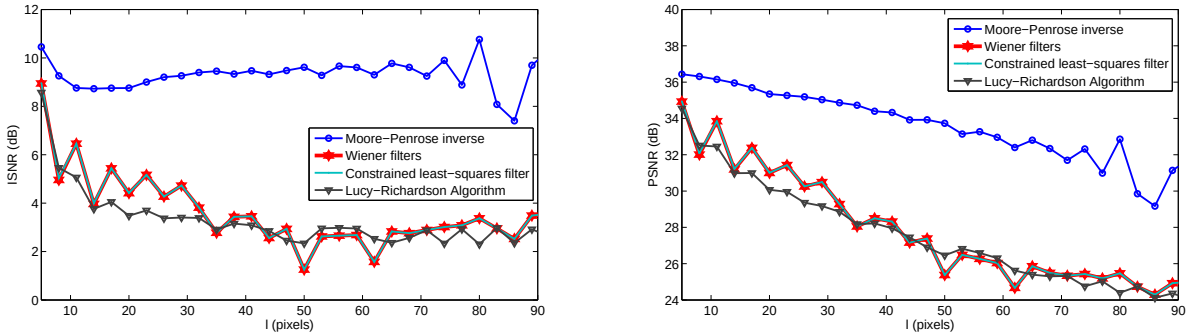

Figure 4.5. (left) Improvement in signal-to-noise-ratio vs. length of the blurring process

(right) Peak signal-to-noise ratio vs. length of the blurring process.

The *ISNR* and *PSNR* values presented in Figure 4.5 indicate that *MP* method is the best among the other methods.

Similar numerical results are generated when the image is blurred by a non-uniform motion detrmined with the relations (10) and (11). The case when the image is blurred by a non-uniform blurring with parameters $l_c = 35$ and $l_r = 25$, is presented in Figure 4.6.

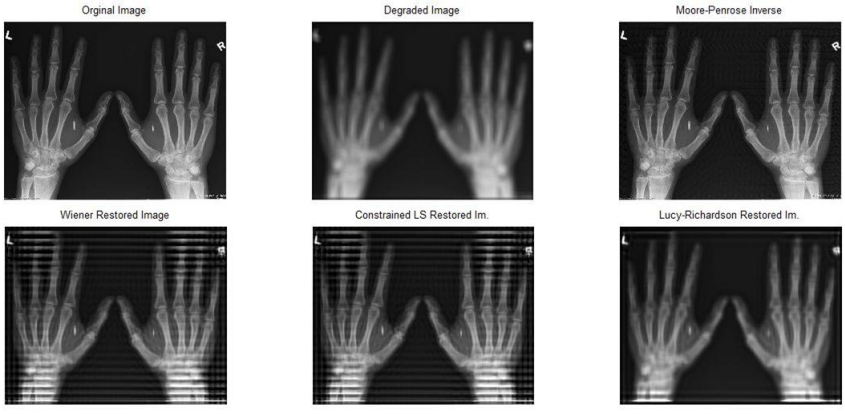

Figure 4.6. Removal of blur, caused by the non-uniform blurring model with $l_c = 35$ and $l_r = 25$.

A confirmation that the proposed *MP* method is faster than the other methods for computing the Moore-Penrose inverse is illustrated in Figure 4.7.

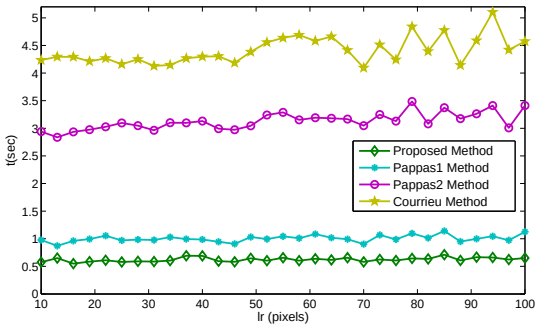

Figure 4.7. Computational time versus variable length lr and $lc = 25$ of the blurring process.

The results presented in Figure 4.7 are made on an Intel(R) Core(TM) i5 CPU M430 @ 2.27 GHz 64/32-bit system with 4 GB RAM memory.

5. Conclusions

We introduce a new method for restoration of images which are blurred by a uniform linear motion. The method is based on the usage of the Moore-Penrose inverse solution of the matrix equation which presents a model of the motion blur. Our method exploits the structure of the blurring matrix and generates its pseudoinverse directly, without any iterations. The main advantage of the method is found in the decrease of the CPU time with respect to other methods for paseudoinverse computation. We illustrate the theoretical findings by comparing the Moore-Penrose inverse method against the Wiener filter, Constrained least-squares filter and Lucy-Richardson algorithm. Also, we present numerical results in which we compare our method (called *MP* method) with well-known *Pappas*1, *Pappas*2 and *Courrieu* methods.

References

[1] M. R. Banham and A. K. Katsaggelos, Digital image restoration, IEEE Signal Processing Magazin 14(2) (1997) 24–41.
[2] A. Ben-Israel and T.N.E. Greville, Generalized inverses: theory and applications, (2nd edition), Springer, 2003.
[3] J. Biemond, R. L. Lagendijk, R. M. Mersereau, Iterative methods for image deblurring, Proceedings of the IEEE 78(5) (1990) 856–883.
[4] A. Bovik, The essential guide to the image processing, Academic Press, 2009.
[5] A. Bovik, Handbook of image and video processing, Academic Press, San Diego, San Francisko, New York, Boston, London, Sydney, Tokyo, 2000.
[6] S. Chountasis, V. N. Katsikis, D. Pappas, Applications of the Moore-Penrose inverse in digital image restoration, Mathematical Problems in Engineering, Volume 2009, Article ID 170724, 12 pages doi:10.1155/2009/170724.
[7] S. Chountasis, V. N. Katsikis, D. Pappas, Digital Image Reconstruction in the Spectral Domain Utilizing the Moore-Penrose Inverse, Mathematical Problems in Engineering, Volume 2010, Article ID 750352, 14 pages doi:10.1155/2010/750352.
[8] S. Chountasis, V. N. Katsikis, D. Pappas, Image restoration via fast computing of the Moore-Penrose inverse matrix, Systems, Signals and Image Processing, (2009), 4 pages.
[9] P. Courrieu, Fast Computation of Moore-Penrose Inverse matrices, Neural Information Processing-Letters and Reviews (2005) 8:25–29.
[10] R. C. Gonzalez, R. E. Woods, S. L. Eddins, Digital Image Processing Using MATLAB, Prentice-Hall, 2003.
[11] P.C. Hansen, J.G. Nagy, D.P. O'Leary, Deblurring images: matrices, spectra, and filtering, SIAM, Philadelphia, 2006.
[12] Image Processing Toolbox User's Guide, The Math Works, Inc., Natick, MA, 2009.
[13] A. K. Katsaggelos, editor, Digital Image Restoration, Springer Verlag, New York, 1991.
[14] V. Katsikis, D. Pappas, Fast computing of the Moore- Penrose Inverse matrix, Electronic Journal of Linear Algebra 17 (2008) 637–650.
[15] F. Krahmer, Y. Lin, B. Mcadoo, K. Ott, J. Wang, D. Widemann, and B.Wohlberg, Blind image deconvolution: Motion blur estimation, In IMA Preprints Series 2133-5 (2006).
[16] D. Kundur and D. Hatzinakos, Blind image deconvolution: an algorithmic approach to practical image restoration, IEEE Signal Processing Magazin 13(3) (1996) 43–64.
[17] R. L. Lagendijk and J. Biemond, Iterative Identification and Restoration of Images, Kluwer Academic Publishers, Boston, MA, 1991.
[18] MATLAB 7 Mathematics, The Math Works, Inc., Natick, MA, 2010.
[19] M. A. Rakha, On the Moore-Penrose generalized inverse matrix, Applied Mathematics and Computation 158 (2004) 185–200.