Pattern Recognition Letters 5 (1987) 49–61
North-Holland

January 1987

Inversion of picture operators

Haim SHVAYSTER and Shmuel PELEG
Department of Computer Science, The Hebrew University of Jerusalem, 91904 Jerusalem, Israel

Received August 1985

Abstract: Inversion of operators on pictures is shown to be an important part of classical image restoration techniques. A method is presented for nonlinear inversion of operators that enables nonlinear restoration. The problem of out of range values is handled by introducing a new definition for operations on pictures such that the set of all pictures is a vector space. It is empirically shown that the Conjugate Gradients algorithm converges quickly in this vector space and enables inversion of operators on big pictures with relatively little computation. Examples are given for applications of this method to linear and nonlinear operators.

Key words: Image restoration, picture vector space, nonlinear restoration.

1. Introduction

In image restoration, given a blurred picture and a model of the blurring process, the original (pre-blurred) picture is sought. Since digital images are represented by matrices, the blurring process can be regarded as an operator on matrices. The de-blurring problem can be formulated as:
Find a picture x such that

$$f(x)+n=y \tag{1}$$

where y is the observed blurred picture, n is stochastic additive noise, and f is the blurring operator.

We assume that the signal x and the noise n are independent.

The error criterion which is usually applied for solving (1) is

$$e1=E\{\|x-\hat{x}\|^2\} \tag{2}$$

where $\hat{x}$ is the estimated solution, x the exact solution, and E the expectation operator over all x, n, y of equation (1).

This research has been supported by a grant from the Israel Academy of Sciences.

The exact solution that appears in the definition of $e1$ is the picture x which satisfies equation (1) when $n\equiv 0$, i.e. a solution of an equation of the type

$$f(x)=y. \tag{3}$$

We will show that when f is a blurring operator, and the solution x is required to be a picture, the solution to equation (3) is nonunique. This means that the 'exact' solution x in the definition of $e1$ is ambiguous, and therefore, the error $e1$ is not well defined. Because of the non-uniqueness, the error criterion which corresponds to $e1$ is

$$e1'=\inf_x E\{\|x-\hat{x}\|^2\} \tag{4}$$

where $\hat{x}$ is the estimated solution, E the expectation operator, and the infimum is over all x which are solutions to (1) with $n\equiv 0$.

The problem with the above definition is that a practical method for minimizing $e1'$ is hard to find.

We suggest another error criterion:

$$e2=E\{\|f(x)-\hat{f}\|^2\} \tag{5}$$

where E, f, x, are as in equations (1) and (2), and $\hat{f}$ is the estimate for the blurred picture in the case $n\equiv 0$.

Solving equation (1) using $e2$ as the error criterion has two main steps, to which we refer as Algorithm 1.

Algorithm 1.

(1) Minimize $e2$ to get an estimate $\hat{f}$ for $f(x)$ using statistical information about the noise and $f(x)$. (Noise cleaning.)

(2) Invert the operator f by solving

$f(x)=y$ where $y=\hat{f}$.

Both definitions of $e1$ and $e2$ are very intuitive, but they are not equivalent in general. We will show however, that under certain assumptions on f which are used in classical linear restoration techniques, both definitions *are* equivalent. The equivalence is in the sense that a solution which minimizes one of them minimizes the other. Establishing the equivalence conditions between $e1$ and $e2$ will be done in Section 2. In Section 3 we show that even for simple linear blurring operators, the solution of equation (1) is nonunique. This means that $e1$ cannot be used as an error criterion; however, $e2$ is still applicable. Using $e2$ as an error criterion (i.e. using Algorithm 1) separates the noise cleaning phase from the operator inversion phase. The topic of noise cleaning will not be considered here, and the rest of the paper will deal with the problem of nonlinear operator inversion.

2. Equivalence conditions between $e1$ and $e2$

In this section we show that when f is *linear* and *invertible*, (i.e. has a unique inverse) the best solution for equation (1) using $e1$ as error criterion is identical to the best solution using $e2$ as error criterion. In practice, one cannot usually find the best solution, and it is common to use the best linear solution (Wiener filter). It is shown that under the above conditions, equivalence between $e1$ and $e2$ also holds for the linear case.

2.1. The nonlinear case

Lemma 1. *The best (nonlinear) approximation for x in the equation*

$$f(x)+n=y$$

where n is a stochastic noise and x, y matrices, under the error criterion $e1$ is

$$\hat{x}=E\{x\,|\,y\}.$$

Lemma 2. *The best (nonlinear) approximation for f in the equation*

$$f+n=y$$

where n is a stochastic noise and f, y matrices, under the error criterion $e2$ is

$$\hat{f}=E\{f\,|\,y\}.$$

Both lemmas are results of a general theorem in nonlinear approximation theory [1]. We give here a direct proof for Lemma 1, to show its independence of any characteristics of f.

Proof of Lemma 1. Denote the approximate solution by $\hat{x}$. It is a function of y, i.e.:

$$\hat{x}=g(y);$$

$\hat{x}$ minimizes $e1$:

$$\begin{aligned} e1 &= E\{\|x-\hat{x}\|^2\}=E\{\|x-g(y)\|^2\} \\ &= \int_{xy}\|x-g(y)\|^2 p(x,y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_{xy}\|x-g(y)\|^2 p(x\,|\,y)p(y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_{y} p(y)\left[\int_{x}\|x-g(y)\|^2 p(x\,|\,y)\,\mathrm{d}x\right]\mathrm{d}y. \end{aligned}$$

Minimizing $e1$ is equivalent to minimizing the term in brackets for every instance of y. For a fixed y, $g(y)$ is a constant (matrix) $g(y)=c$.

We proceed to minimize the term in brackets:

$$\begin{aligned} &\int_x \|x-c\|^2 p(x\,|\,y)\,\mathrm{d}x \\ &\quad=\int_x \|x\|^2 p(x\,|\,y)\,\mathrm{d}x \\ &\qquad -2\int_x \sum_{ij} x_{ij}c_{ij}p(x\,|\,y)\,\mathrm{d}x+\|c\|^2\int_x p(x\,|\,y)\,\mathrm{d}x \end{aligned}$$

The minimum point is the point where the derivatives with respect to c_{ij} equal 0 for all ij, i.e.

$$2\int_{x_{ij}} x_{ij}p(x|y)\,dx_{ij} = 2c_{ij}\cdot\int_{x_{ij}} p(x|y)\,dx_{ij}$$

$$c_{ij} = \frac{\int_{x_{ij}} x_{ij}p(x|y)\,dx_{ij}}{\int_{x_{ij}} p(x|y)\,dx_{ij}}$$

and in matrix notation:

$$c = \frac{\int_x xp(x|y)\,dx}{\int_x p(x|y)\,dx} = E\{x|y\}. \qquad \square$$

The proof of Lemma 2 is identical and will be omitted.

Theorem 1. *When f is **linear and invertible**, the best (nonlinear) approximation according to $e1$ and the result of the two steps of Algorithm 1 are the same.*

Proof. The best solution using $e1$ is given by Lemma 1:

$$\begin{aligned}\hat{x} &= E\{x|y\} = E\{f^{-1}(y-n)|y\}\\ &= E\{f^{-1}(y) - f^{-1}(n)|y\}\\ &= E\{f^{-1}(y)\} - E\{f^{-1}(n)|y\}\\ &= f^{-1}(y) - E\{f^{-1}(n)|y\}.\end{aligned}$$

The best solution using $e2$ is given by Lemma 2:

$$\begin{aligned}\hat{f} &= E\{f(x)|y\} = E\{y-n|y\}\\ &= E\{y\} - E\{n|y\}\\ &= y - E\{n|y\},\\ \hat{x} &= f^{-1}(\hat{f}) = f^{-1}(y - E\{n|y\})\\ &= f^{-1}(y) - f^{-1}(E\{n|y\})\\ &= f^{-1}(y) - E\{f^{-1}(n)|y)\}. \qquad \square\end{aligned}$$

Notice that the special properties of f are used only in the second step of Algorithm 1, and the result

$$\hat{f} = y - E\{n|y\}$$

does not depend on any property of f. Therefore, $\hat{x}$ can always be defined as a solution to the equation:

$$f(x) = y - E\{n|y\}$$

2.2. The linear case

It is known that when f is linear and invertible, the best *linear* solution to equation (1) using $e1$ as error criterion is the Wiener filter [2]. This filter is usually implemented in the frequency domain:

$$\hat{X} = Y\frac{1}{F}\frac{|F|^2}{|F|^2 + S_{nn}/S_{xx}}$$

where $\hat{X}, Y, F, S_{nn}, S_{xx}$ are the Fourier transforms of $\hat{x}, y, f, R_{nn}, R_{xx}$. R_{nn} and R_{xx} are the autocorrelations of n and x.

This filter can also be obtained as the result of Algorithm 1. To show this, the following properties from linear estimation theory are used [3]:

(a) The best *linear* solution $\hat{f}$ to the equation

$$f + n = y$$

where f, y matrices and n stochastic noise independent of f, under error criterion $e2$ is:

$$\hat{F} = Y\frac{1}{1 + S_{nn}S_{ff}}$$

where $\hat{F}, Y, S_{nn}, S_{ff}$ are the Fourier transforms of $\hat{f}, y, R_{nn}, R_{ff}$.

(b) If

$$\hat{f} = f * x$$

where * indicates the convolution operator then

$$S_{ff} = S_{xx}|F|^2$$

where S_{ff}, S_{xx}, F are the Fourier transforms of R_{ff}, R_{xx}, f.

Theorem 2. *If f is **linear** and **invertible** then the best **linear** approximation defined by $e1$ and the result of the two steps of Algorithm 1 are the same.*

Proof. It is enough to show that the two steps of Algorithm 1 result in the Wiener filter. Using properties (a) and (b) for the first step of Algorithm 1 we get:

$$\hat{F} = Y\frac{|F|^2}{|F|^2 + S_{nn}/S_{xx}}.$$

In step 2 of Algorithm 1 we have to invert the operator. If $\hat{f} = f * x$ then $\hat{F} = F\cdot X$ and so:

$$F\cdot X = Y\frac{|F|^2}{|F|^2 + S_{nn}/S_{xx}}.$$

The Wiener filter is obtained by dividing both sides by F. □

Notice that the use of f in the first step was only for transferring the statistical information of S_{xx} to S_{ff}.

When f does not have a unique inverse, (the conditions of the theorem do not hold), the definition of $e1$ is of little use. Algorithm 1, however, can still be used. Step 1 of this algorithm can be any noise filtering method. The rest of this paper deals mainly with the second step of the restoration, i.e. inverting operators as in equation (3).

3. Problems involved in inverting picture operators

The operator f in equation (3) may have an inverse operator f^{-1}. In these cases, it seems as though the solution to equation (3) should be $f^{-1}(y)$; however, this solution cannot always be interpreted as a picture. (As will be seen later, f does not usually have a unique inverse.) The difficulties in applying mathematical techniques to solve equation (3) and ge *a picture* as a solution are as follows:

(a) *Nonuniqueness*

If f is a blurring operator, every pixel in the picture y can be considered as a result of a weighted average of several neighboring pixels in x. This is also true for pixels on the boundaries of the picture y; therefore, the size of x must be larger than the size of y. If the boundaries of x are known in advance, y can be exactly restored, but this is usually not the case. This means that f is not 1–1 and has no unique inverse.

(b) *Out of range values. Nonlinear inversion*

Digital pictures are represented in the computer by matrices of numerical values in a specified range $[0, M]$. Solving equation (3) without introducing range constraints on the values of the pixels usually results in solution values that are out of range. Therefore, a projection back into the picture domain (i.e. having all pixels in the range $[0, M]$) must be performed. However, such a projection may destroy the solution. In other words, if $x^* = f^{-1}(y)$ is a solution to equation (3) and x^{**} is a projection of x^* into the picture domain that obeys the range constraints, then usually $f(x^{**}) \neq y$. Solving equation (3) with range constraints requires much more computation. The simpler problem of keeping only positivity constraints has been discussed in [4] with no satisfactory solution. Positivity constraints have a physical meaning, since light illumination cannot be negative. We are forced to introduce an additional constraint due to the way pictures are digitized and stored, having a maximum value as well as a minimum value. By range constraints we therefore mean that all elements in the solution matrix x are in the range $[0, M]$.

If $f^{-1}(y)$ has out of range values, it cannot be accepted as a solution. The problem of finding a solution must be formulated now as a quadratic programming problem:

$$\text{Minimize } \| f(x) - y \|^2 \tag{6}$$

where x satisfies the range constraints.

Various Numerical Analysis algorithms are known that can be applied to this quadratic programming problem [5]. However, they all perform badly for relatively large pictures. In this paper we try to handle the computational complexity of this problem by a new method, which eliminates the need for range constraints.

4. Pictures as elements in a vector space

A solution to the out of range values problem was suggested by the authors in [6]. The idea is to define operations between pictures in such a way that the set of all pictures is a closed vector space. Two basically different approaches were suggested. The first was a definition of equivalence classes among pictures based on a visual model, and the second was a mapping of the set of pictures into $\mathbb{R}^n$ by means of a special mappings. A generalization of the second method follows:

4.1. Mapping of pictures into $\mathbb{R}^n$

Since all finite vector spaces having the same dimension are isomorphic, a definition of opera-

tions between pictures is equivalent to a definition of a *one to one and onto* mapping from the set of all pictures into $\mathbb{R}^n$, where n is the number of pixels. Such mappings exist, because the domain and the range have the same cardinal number; however, they cannot be continuous, since the domain is compact and the range is not. The continuity of the mapping is very important, since otherwise infinitely small errors may cause large errors in grey levels. Continuous mappings can be found if the domain is considered an open set. The domain is compact because the grey levels are from the closed interval $[0, M]$. In practice, grey levels are obtained as results of measurements with finite accuracy, where the minimum grey level value 0, as well as the maximum grey level M, are obtained after rounding. Assuming an error δ in the measurements and the rounding process, it is possible to use ε, $0<\varepsilon<\delta$ for zero, and $M-\varepsilon$, $0<\varepsilon<\delta$ for M. Thus, the domain is an open set, and continuous mappings (homeomorphisms) can be found.

Every homeomorphism between $(0, M)^n$ and $\mathbb{R}^n$ defines operations among pictures. We will characterize a family of mappings which are 'simple', although other mappings may still be useful for special purposes. A great simplification is to consider mappings which operate on each coordinate separately, i.e. a mapping

$$\Psi : (0, M) \to \mathbb{R}.$$

The following properties are applied to Ψ:

(1) Ψ is a homeomorphism.

(2) Property (1) implies that Ψ is monotone. We require Ψ to be monotonically increasing to ensure that whenever one grey level is greater then another, the relation will also hold after the mapping.

(3) Antisymmetry around the middle grey level $M/2$, i.e. $\Psi(M/2-g) = -\Psi(g)$. This simplifies the computation of Ψ, and, as will be shown later, will enable a nice characterization of a negative picture.

From (1), (2), and (3) it follows that

$$\Psi(0) = -\infty, \qquad \Psi(M/2) = 0, \quad \Psi(M) = \infty.$$

We introduce a new variable t related to the grey level g by

$$t = \frac{g - M/2}{M/2}, \tag{7}$$

which satisfies $-1<t<1$. The mapping Ψ written in terms of t is

$$\Psi(g) = \Phi(t) = \Phi\left(\frac{g - M/2}{M/2}\right) \tag{8}$$

and we have

$$\Phi : (-1, 1) \to (-\infty, \infty)$$

Φ is a homeomorphism, monotone increasing, and antisymmetric around 0, satisfying

$$\Phi(-1) = -\infty, \qquad \Phi(0) = 0, \qquad \Phi(1) = \infty.$$

Considering functions with Taylor expansions, the antisymmetry around 0 implies that Ψ is odd:

$$\Phi(t) = \sum_{j=0}^{\infty} a_j t^{2j+1}.$$

Because $\Phi(1) = \infty$, there are infinitely many $a_j \neq 0$. To ensure convergence for $-1<t<1$, it is enough to require that a_j is a rational function of j. To ensure the monotone increasing property, it is enough to require that $a_j \geq 0$ for all j. The typical graph of these functions is shown in Figure 1.

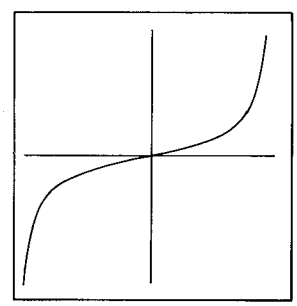

Figure 1. Graph of the Φ functions.

Examples. (a) $a_j = 1$ $\forall j$,

$$\Phi(t) = t \sum_{j=0}^{\infty} (t^2)^j = \frac{t}{1-t^2},$$

$$\Phi^{-1}(s) = \frac{\sqrt{4s^2+1}-1}{2s}.$$

We call this mapping the Log-ratio transformation.

(b) $a_j = (1/2j+1)$ $\forall j$,

$$\Phi(t) = \sum_{j=0}^{\infty} \frac{1}{2j+1} t^{2j+1} = \log\left(\frac{1+t}{1-t}\right) = 2\tanh^{-1}(t),$$

$$\Phi^{-1}(s)=\frac{e^s-1}{e^s+1}=\tanh\left(\frac{1}{2s}\right).$$

We call this mapping the Log-ratio transformation.

(c) The tangent function.

Throughout this paper we use the following convention: If x is a picture, i.e. a matrix (x_{ij}) where $0<x_{ij}<M$, we define $\Psi(x)$ as the mapping Ψ applied to each element of x.

4.2. *A definition of operations on pictures*

Let x, y be pictures, and α a scalar. We define:

$$x+y\equiv\Psi^{-1}(\Psi(x)+\Psi(y))$$

$$\alpha\cdot x\equiv\Psi^{-1}(\alpha\Psi(x)).$$

Examples of these operations for the Ratio transformation are given in Figure 2. The results for the Log-ratio transformation gave very similar results.

The following properties are of interest:

(1) $-x\equiv(-1)\cdot x\equiv\Psi^{-1}(-\Psi(x))$. From (7) and (8) it follows that every pixel g is mapped to $M-g$, which is the classical definition for the negative picture.

(2) The zero picture is the picture with all grey values equal to $M/2$.

(3) The sum $x+y$ preserves visual information from both x and y.

(4) The multiplication, $\alpha\cdot x$, increases contrast for $\alpha>1$, and decreases contrast for $0<\alpha<1$.

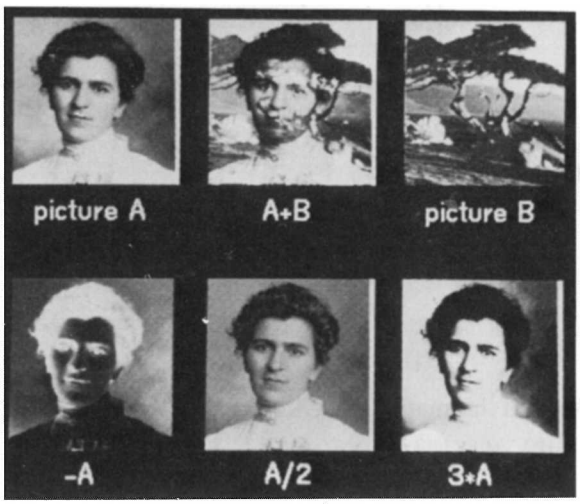

Figure 2. Addition and multiplication in the Ratio space.

When $\alpha\to\infty$, the picture is thresholded at $M/2$.

An apparent disadvantage of the suggested technique is in dealing with linear operators. Many practical operators are linear (e.g. convolution operators). These operators are nonlinear after the transformations; however, we will show that dealing with such operators is simple, and gives very good results in spite of the nonlinearity.

5. Numerical optimization

Using the new definition of operations among pictures, the quadratic programming problem (6) with constraints, can be solved without constraints, but the functional is no more quadratic. All operations, including the norm, are performed in the new spaces. The resulting minimization problem is: Find a minimum to

$$g(X)=\|F(X)-Y\| \quad \text{over all } X, \tag{9}$$

where Y is the blurred picture, and F the blurring operator.

Because of the high dimensionality of both X and Y, and the nonuniqueness of the solution, an iterative algorithm was used to determine a solution. The numerical method used can be any method which can be applied to nonquadratic functionals. The Conjugate Gradients algorithm which has already been tried in [7], proved to be very efficient in our experiments. See Table 1.

Table 1
The Conjugate Gradients method

Minimizing the functional g

(1) Initially choose an arbitrary guess X_0 and set $r_0=-\nabla g(X_0)$, $P_0=r_0$.

(2) Choose α to minimize $\Phi(\alpha)=g(X_k+\alpha P_k)$.

(3) $X_{k+1}=X_k+\alpha P_k$; $r_{k+1}=-\nabla g(X_{k+1})$.

(4) $b=\dfrac{|r_{k+1}|^2-\langle r_k, r_{k+1}\rangle}{|r_k|^2}$; $P_{k+1}=r_{k+1}-bP_k$.

For a quadratic g, the algorithm is known to converge in less than n iterations, where n is the size of X [8]. In our case, g is nonquadratic, and n (the number of pixels), is usually very large, so that n iterations are impractical. However, experimental results that will be described later, show

that convergence up to a reasonable accuracy can be achieved in a very small (constant) number of iterations. Notice that all operations of addition, scalar multiplication, inner product, and differentiation are in the Ratio or Log-ratio spaces. The computation of the gradient and the method for optimization along a line are discussed in the appendix.

6. Experimental results

A series of computer simulation experiments have been performed to investigate and evaluate the suggested method. The points that were tested were:

(1) The number of iterations needed for a reasonable convergence and its dependence on the picture size.

(2) The quality of the results compared to other restoration techniques.

(3) Sensitivity to noise.

(4) The effect of the initial guess.

(5) Sensitivity to various kinds of blurring functions. The point spread functions we tried were:

(a) Uniform motion blur simultated by giving each pixel a directional average of its neighbors.

(b) Gaussian blur:

$$h(x, y, \alpha, \beta) = \exp\left(- \frac{(x-\alpha)^2 + (y-\beta)^2}{B} \right).$$

(c) Sinc blur:

$$h(x, y, \alpha, \beta) = \left[\frac{\sin((x-\alpha)/B)}{(x-\alpha)/B} \right]^2 \cdot \left[\frac{\sin((y-\beta)/B)}{(y-\beta)/B} \right]^2.$$

(d) A single edge detection operator.

All experiments were performed on a Vax 780 computer with a single precision floating point arithmetic. In all cases except when otherwise stated, the initial guess was the zero picture (i.e. a picture with all grey levels equal $M/2$.) The reason for this in connection with maximum entropy restoration techniques will be discussed later. In the noisy pictures, a random noise from a Gaussian distribution was added and the SNR (signal to noise ratio) was computed as the ratio between the variance of the picture and the noise variance.

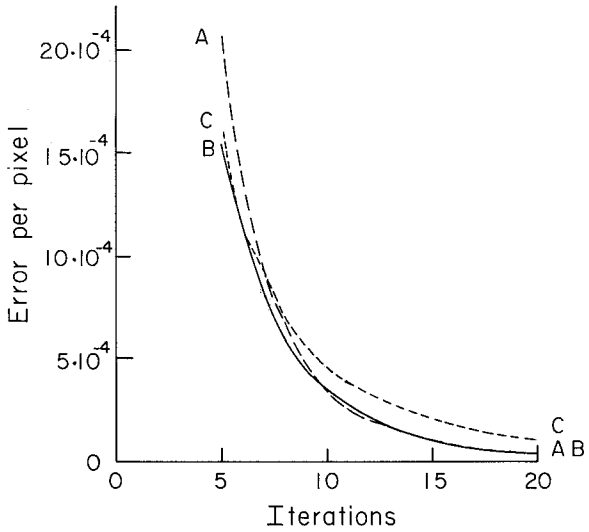

Figure 3. Speed of convergence for several picture sizes. A - Picture 64×64, B - Picture 128×128, C - Picture 400×400.

We tried our restoration technique on pictures of several sizes. The value of the error $g(X)$ as defined by equation (9), divided by the number of pixels, for picture sizes of 64×64, 128×128, and 400×400, is plotted as a function of the iteration number in Figure 3. The decrease of $g(X)$ in the first several iterations was very fast. Therefore, for scaling purposes, the graph shows values starting at the fifth iteration. The blurring in this case was a noise free uniform motion blur of ten pixels for the smaller pictures, and twenty for the 400×400 picture. As can be seen, the size of the picture seems to have a very small effect on the convergence speed, and the error per pixel is sometimes even smaller for the bigger pictures. Some of the iterations of the 128×128 picture are shown in Figure 4. Notice that (as discussed in Section 3) the restored picture is larger than the blurred picture.

Comparing the quality of the restored pictures obtained by our method to the quality of restorations obtained by other techniques involves some difficulties. The difficulties are in finding a good criterion for the comparison. The criterion we used in our experiments was the Euclidean distance between the original and the restored picture:

$$d = \sqrt{\frac{\sum_{ij} (x_{ij} - y_{ij})^2}{n}}$$

where y is the original picture, x, the restored pic-

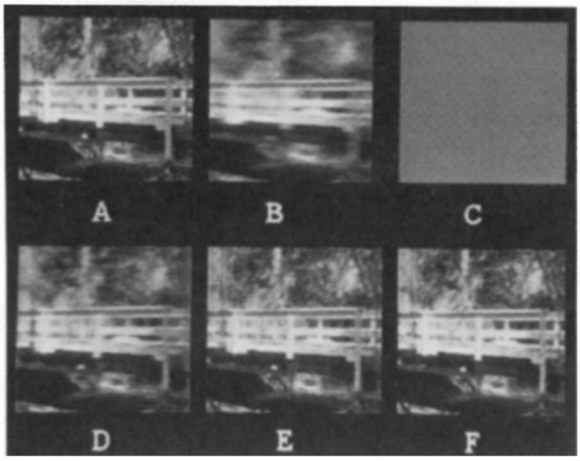

Figure 4. Restoration of a picture degraded by a noise free uniform motion blur. A - Original picture, B - Blurred picture, C - Initial guess, D - After 5 iterations, E - After 10 iterations, F - After 20 iterations.

ture, and n the number of pixels. We refer to this distance as the posteriori error in the restoration. Although very simple to compute, the distance posteriori error proved to be far from ideal. In noisy pictures, the distance posteriori error *increases* after several restoration iterations, although it is obvious (by looking at the resulting pictures) that the iterations improve the pictures.

The two other methods we compared to ours were the Inverse Fourier Filter, where the Fourier transform of the picture is divided by the Fourier transform of the blurring operator, and an algebraic inversion method. We tried two methods of handling the out of range values that occur in these methods: clipping and rescaling. Our experiments show that clipping always gave better results than rescaling. In all experiments, the Fourier Inverse Filter gave the worst results. See for example Figure 5. We focus therefore, on comparing our method with the algebraic method, which was implemented by using the same conjugate gradient algorithm we used for our Ratio and Log-ratio methods, always with the same initial guess. The results for noisy pictures are given in Figure 5 and the corresponding graphs of our posteriori error in Figures 6 and 7. The results show that for almost every iteration, our method gives a smaller posteiori error, and this is more evident for the pictures with smaller values fo SNR.

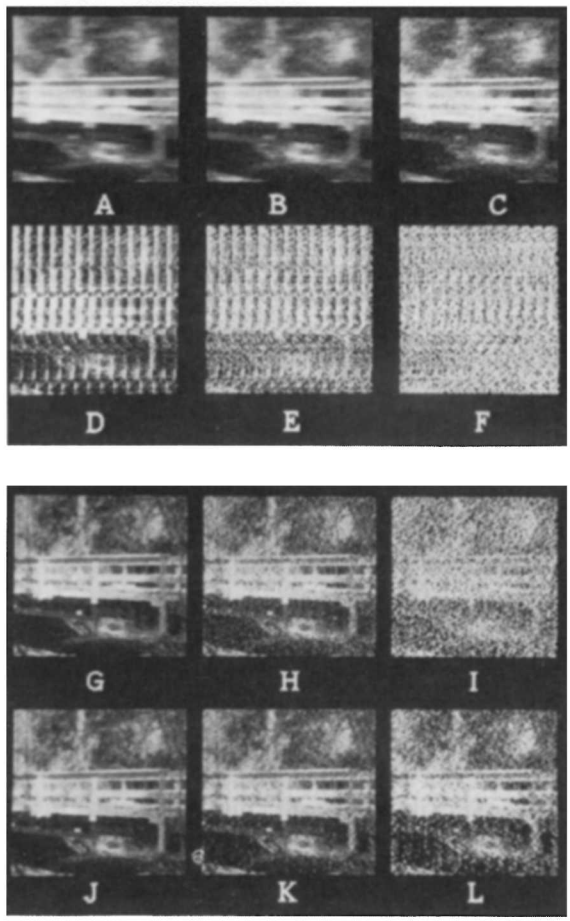

Figure 5. Restoration of noisy blur. A,B,C - Blurred picture, SNR = 1000:1, 100:1, 10:1, D,E,F - Fourier restoration of A,B,C; G,H,I - Algebraic restoration of A,B,C; J,K,L - Log-ratio restoration of A,B,C.

The experiments show that the Log-ratio method performs better, even for the noise free case, for all point spread functions checked. The results of recovering from Gaussian blur and Sinc blur are shown in Figure 8 and the graph comparison to the Algebraic inversion in Figures 9 and 10.

We also tried our algorithm on a simple edge detection operator. The operator gives each pixel the absolute value of the difference between itself and its left neighbor. This operator does not have a unique inverse, and is highly nonlinear. The number of iterations needed in this case was much higher than the number of iterations needed when other operators were used, and therefore, a very small picture of 8×8 pixels was used. The results are shown in Figure 11. In this case, a random picture was used as an initial guess.

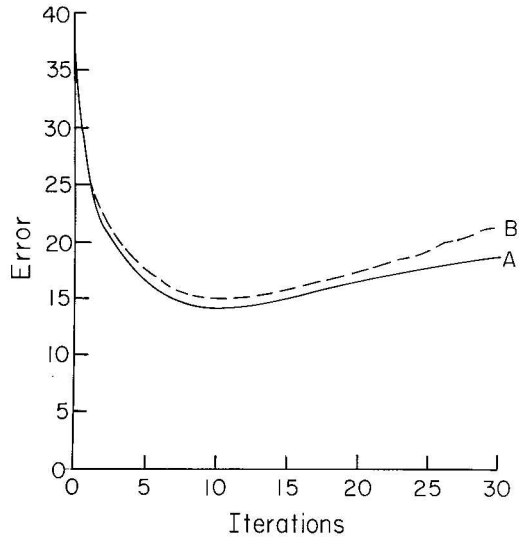

Figure 6. Posteriori errors in several iterations for SNR = 1000:1. A - Log-ratio restoration, B - Algebraic restoration.

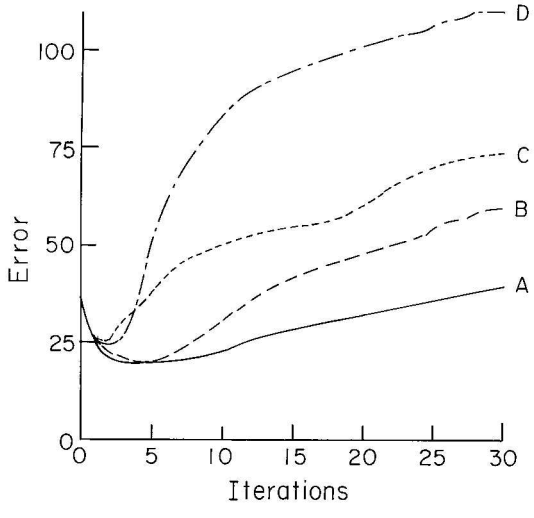

Figure 7. Posteriori errors in several iterations for SNR = 100:1 and 10:1. A - Log-ratio restoration SNR = 100:1, B - Algebraic restoration SNR = 100:1, C - Log-ratio restoration SNR = 10:1, D - Algebraic restoration SNR = 10:1.

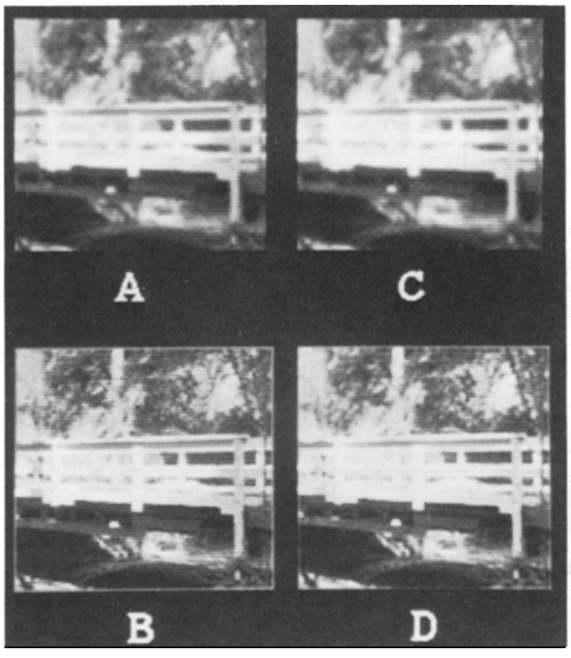

Figure 8. Log-ratio restoration of noise free Gaussian and Sinc blur. A - Gaussian blurred picture. B - A restored after 20 iterations. C - Sinc blurred picture. D - C restored after 20 iterations.

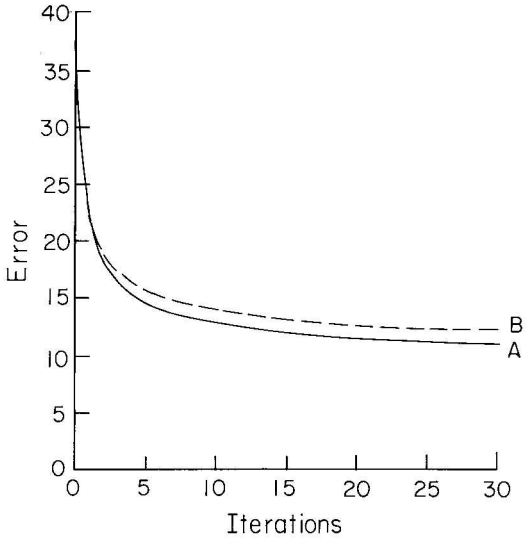

Figure 9. Posteriori errors in several iterations for Gaussian blur. A - Log-ratio restoration, B - Algebraic restoration.

7. Discussion of the results. Comparison to other methods

The main novelty of the technique used in this paper is the way by which our of range values are treated. Although the range constraints include both the positivity constraint and the maximum value constraint, the amount of computation needed is similar to the amount of computation needed by simple algebraic restoration techniques. As reported in [4,9], quadratic optimization techniques with constraints failed to give satisfactory results for pictures of reasonable size. In our

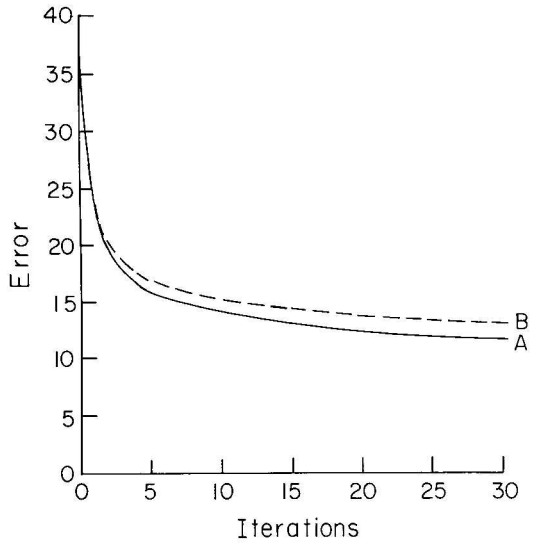

Figure 10. Posteriori errors in several iterations for Sinc blur. A – Log-ratio restoration, B – Algebraic restoration.

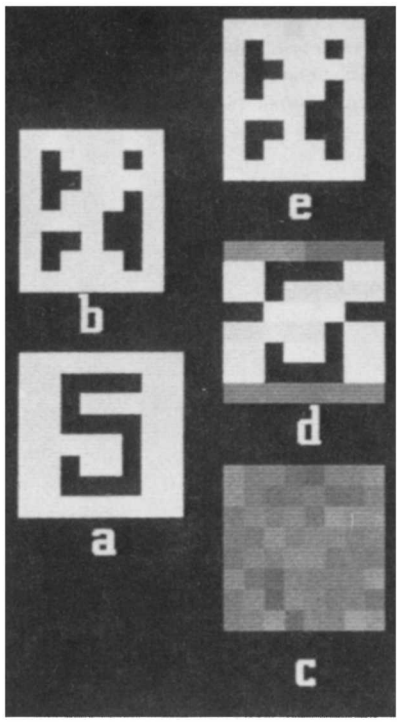

Fig. 11. Recovering a picture from its horizontal edges. A – original 8 × 8 picture, B – After applying the edge operator, C – Initial guess, D – Restored picture, E – Applying the edge operator to D.

method, the constraints are eliminated, but the functional to be minimized is nonquadratic.

The idea of a coordinate transformation was first considered in the work of Oppenheim [10]. His motivation was the model of light reflection from physical bodies. Oppenheim observed that since the reflection is a multiplicative process, taking logarithm enables one to work in a vector space where multiplication is reduced to addition. Multiplicative (nonlinear) operators are thus reduced to linear operators, and linear techniques can be used. Our approach is different. We use coordinate transformation in such a way that linear operators become *nonlinear*. Our approach is not restricted to either linear or multiplicative operators, and can be applied to any kind of operators.

A comparison of the results presented in this paper to other restoration techniques is complicated because many techniques involve noise filtering with inversion of the blurring operator. As was shown in the introduction, these two steps can be considered separately for operators that are linear and invertible. Two of the commonly used techniques for inversion of operators are the inverse Fourier filter and algebraic least squares methods. A short comparison of these techniques to our method is now given.

7.1. The inverse Fourier method

The inverse Fourier filter technique has some problems that do not occur in algebraic restoration methods. Using this method, the operator is assumed to be a cyclic convolution operator. Therefore, the operator is 1-1 and onto with a unique solution which is not necessarily the right solution. It can be shown [4,11] that this assumption about general convolution operators has an undesirable effect at least on the boundaries of the picture, and in some cases, like the uniform motion blur, over the whole picture. The restored picture obtained by this method has the same size as the blurred picture, while in algebraic methods it is possible to get a restored picture which is bigger than the blurred picture.

It is known that Fourier inversion is very sensitive to noise. A relatively small amount of noise causes the noise level in the restored picture to increase drastically. Even after noise filtering, the noise cannot be completely removed, and because of the high sensitivity to noise of the inversion pro-

cess, the results are unsatisfactory. Inverse Fourier restoration results almost always in out of range values. These are usually treated by rescaling or clipping. Our experiments show that the clipping approach is better than rescaling, but still, with SNR of less than 1000:1 the Fourier inversion is unsatisfactory. From the computational aspect, the inverse Fourier can be carried out in $O(n \log n)$ where n is the total number of pixels, when the picture dimensions are powers of 2. In other cases, the computation time may grow considerably. In this case one cannot pad the picture with zeros to get the right size, because this will affect the entire restored picture. Algebraic methods like ours, take time of $O(nm)$ where n is the total number of pixels and m the size of the point spread function which is usually small.

7.2. Algebraic methods

The major difference between the method suggested in this paper and other least square techniques is in the handling of out of range values. Intuitively, one can consider our method as giving different weights to error in the restoration process in such a way that out of range values result in infinite error, and therefore can never happen. The experimental results discussed in the previous section show that our method gave improved results for all point spread functions and for all levels of noise. As can be expected, the superiority of our method becomes more evident as the noise level increases, but even for noise free blur, our restoration gives better results and seems to converge faster. Although we did not try other algorithms like the pseudo inverse method [12], it seems reasonable that their asymptotic behavior is similar to the Conjugate Gradients algorithm used in our experiments (for the unconstrained restoration). It seems to us that the inversion of the edge detection operator described in the previous section is a very good example to the advantage of using range constraints. One cannot hope to solve such a problem without introducing such constraints.

Some restoration techniques assume a partial knowledge about the noise statistics, usually its variance. In these cases, even if f is invertible, there are infinitely many possible solutions. The solution is then chosen by using an additional constraint, like a smoothness criterion [4] or a maximum entropy [13]. Introducing a priori knowledge is also possible in our method, simply by choosing an appropriate initial guess for the iterative algorithm. Assuming the set of all solutions (in our case, the set of all local minima) to be dense, starting from an initial guess will converge to a solution close to that initial guess. In our experiments, the picture used for an initial guess was almost always a picture with all grey levels equal to $M/2$. It is easily seen that this is the maximum entropy picture. Therefore, the local minima obtained are with high entropy.

8. Appendix

In this appendix we show how to implement the Conjugate Gradient algorithm in the picture vector space defined by a mapping Ψ as discussed in Section 4. In the following discussion, the pictures are given as matrices

$$x=(x_{ij}),\quad i=1,\dots,m,\quad j=1,\dots,n,\quad 0<x_{ij}<M,$$

$$y=(y_{ij}),\quad i=1,\dots,p,\quad j=1,\dots,q,\quad 0<y_{ij}<M.$$

Using the mapping Ψ, we define a normalization of the picture matrices:

$$x\to\Psi(x)\equiv X,\qquad y\to\Psi(y)\equiv Y.$$

8.1. Computing the gradient

Since the optimization is performed in one of the new spaces, the operator F in (9) is not f of (3), but the corresponding operator in the new space, i.e.:

$$F(X)=\Psi(f(\Psi^{-1}(X)))=\Psi(f(x))$$

$$F:\mathbb{R}^{mn}\to\mathbb{R}^{pq}.$$

Assuming inner product norm in (9), we have:

$$\begin{aligned} g(X) &= \|F(X)-Y\|^2 \\ &= \|F(X)\|^2-2\langle F(X),Y\rangle+\|Y\|^2 \\ &= \sum_{i,j=(1,1)}^{(p,q)}(F(X))_{ij}^2-2\sum_{i,j=(1,1)}^{(p,q)}(F(X))_{ij}Y_{ij} \\ &\quad+\sum_{i,j=(1,1)}^{(p,q)}Y_{ij}^2. \end{aligned}$$

To compute ∇g, we differentiate with respect to $X_{\alpha\beta}$ for arbitrary $\alpha\beta$.

$$\frac{\partial g}{\partial X_{\alpha\beta}} = 2 \sum_{i,j=(1,1)}^{(p,q)} (F(X))_{ij} \frac{\partial (F(X))_{ij}}{\partial X_{\alpha\beta}} - 2 \sum_{i,j=(1,1)}^{(p,q)} Y_{ij} \frac{\partial (F(X))_{ij}}{\partial X_{\alpha\beta}} = 2 \sum_{i,j=(1,1)}^{(p,q)} ((F(X))_{ij} - Y_{ij}) \frac{\partial (F(X))_{ij}}{\partial X_{\alpha\beta}},$$

$$\frac{\partial (F(X))_{ij}}{\partial X_{\alpha\beta}} = \Psi'((f(x))_{ij}) \frac{\partial (f(x))_{ij}}{\partial X_{\alpha\beta}},$$

$$\frac{\partial (f(x))_{ij}}{\partial X_{\alpha\beta}} = \frac{\partial (f(\Psi^{-1}(X)))_{ij}}{\partial X_{\alpha\beta}} = \frac{\partial (f(x))_{ij}}{\partial x_{\alpha\beta}} (\Psi^{-1})'(X_{\alpha\beta}).$$

From the above equations it follows that

$$\frac{\partial (F(X))_{ij}}{\partial X_{\alpha\beta}} = \frac{1}{\Psi'(x_{\alpha\beta})} \Psi'((f(x))_{ij}) \frac{\partial (f(x))_{ij}}{\partial x_{\alpha\beta}}$$

and then

$$\frac{\partial g}{\partial X_{\alpha\beta}} = \frac{2}{\Psi'(x_{\alpha\beta})} \sum_{i,j=(1,1)}^{(p,q)} ((F(X))_{ij} - Y_{ij}) \Psi'((f(x))_{ij}) \times \frac{\partial (f(x))_{ij}}{\partial x_{\alpha\beta}}.$$

f is usually a local operator, and therefore, the term $\partial f(x)_{ij}/\partial x_{\alpha\beta}$ is identical to 0 except for few i,j. Therefore, ∇g can be computed in time proportional to the number of pixels. When f is a convolution operator, described by the convolution matrix (a_{ij}), $\partial f(x)_{ij}/\partial x_{\alpha\beta}$ is actually the constant a_{ij} of the convolution matrix.

8.2. Minimization along a line

In the Conjugate Gradients algorithm, at step (2), a minimum along a line has to be found to $\Phi(\alpha) = g(X_k + \alpha P_k)$. Any method for line minimization can be used [5], but since each computation of ∇g or g is very expensive, we used the following method:

Notice that step (2), $\Phi(0) = g(X_k)$ is known and $\Phi'(0) = \langle \nabla g(X_k), P_k \rangle$ can be computed by one inner product. Select a small positive number h (small relative to $\|\nabla g\|$), and compute $\Phi(h) = g(X_k + hP_k)$. From the Taylor expansion

$$\Phi(h) \approx \Phi(0) + h\Phi'(0) + \frac{h^2}{2} \Phi''(0).$$

Therefore,

$$\Phi''(0) \approx 2 \frac{(\Phi(h) - \Phi(0))/h - \Phi'(0)}{h},$$

and a first approximation to α^* that minimizes $\Phi(\alpha)$ is

$$\alpha^* \approx -\frac{\Phi'(0)}{\Phi''(0)}.$$

This approximation is actually a one step Newton-Raphson estimate which is exact when g is quadratic, but can be very rough in some cases.

Alternatively, assuming $\Phi(\alpha)$ to be similar to a paraboloid (it is exactly a paraboloid when g is quadratic), we can solve for the paraboloid coefficients using the three values $\Phi(0)$, $\Phi'(0)$, and $\Phi(h)$, and compute the extremum point of the resulting paraboloid. The resulting formula is identical to the previous formula. When a maximum rather than a minimum is found, α^* is negative. When this rare event happens, another point ($\Phi(2h)$ or $\Phi'(h)$) can be computed to find a third degree interpolation.

8.3. Complexity

We consider a picture of size $n \times n$ and a convolution operator f on m neighbors. The Conjugate Gradients algorithm requires 4 $n \times n$ matrices. An additional matrix is required for Y, and usually another two are needed for temporary storage. The total memory needed is therefore $7n^2$.

Regarding time complexity, the functions Ψ and Ψ^{-1} can be computed in two multiplications each, using a table and linear interpolation. The total number of multiplications per iteration is then $(14 + 3m)n^2$. For example, if $n = 512$, $m = 50$, it takes approximately 43 000 000 multiplications per iteration. Assuming 1 μs per multiplication and that other operations take about the same time as multiplication gives running time of 86 seconds per iteration.

References

[1] Doob, J.L. (1953). *Stochastic Processes*. Wiley, New York.

[2] Rosenfeld, A. and A.C. Kak (1982). *Digital Picture Processing*. Second Edition, Academic Press, New York.

[3] Papoulis, A. (1984). *Probability, Random Variables, and Stochastic Processes*. Second Edition, McGraw-Hill, New York.

[4] Andrews, H.C. and B.R. Hunt (1977). *Digital Image Restoration*. Prentice-Hall, Englewood Cliffs, NJ.

[5] Fletcher, R. (1980). *Practical Methods of Optimization*. Wiley, New York.

[6] Shvayster, H. and S. Peleg (1983). Pictures as elements in a vector space. *IEEE Conf. Computer Vision Pattern Recognition*, Arlington, VA, June 1983, 222–230.

[7] Angel, E.S. and A.K. Jain (1978). Restoration of images degraded by spatially varying point spread functions by a conjugate gradient method. *Appl. Opt.* 17, 2186–2190.

[8] Hestenes, M. (1980). *Conjugate Direction Methods in Optimization*. Springer, Berlin.

[9] Mascarenhas, N.D.A. and W.K. Pratt (1975). Digital image restoration under a regression model. *IEEE Trans. Circuits Syst.* 22, 252–266.

[10] Oppenheim, A.V. and R.W. Schafer (1968). *Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, N.J.

[11] Gonzalez, R.C. and P. Wintz (1977). *Digital Image Processing*. Addison Wesley, Reading, MA.

[12] Pratt, W.K. and F. Davarian (1977). Fast computational techniques for pseudoinverse and wiener image restoration. *IEEE Trans. Comput.*, June.

[13] Burch, S.F., S.F. Gull and J. Skilling (1983). Image restoration by a powerful maximum entropy method. *Comput. Vision Graphics Image Processing* 23, 113–128.