論文 94-19-10-16

영상신호의 고속처리를 위한 최적화 알고리즘에 대한 연구

正會員 權 起 弘*

A study of optimal algorithm for high-speed process of image signal

Kee Hong Kwon*

要 約

본 논문에서는 훼손된 영상을 처리하는 방법에 대하여 연구하였다. 기존의 처리방법은 특이점이나 악 조건일 경우 수렴 속도가 늦어진다는 점과 처리시간이 많이 소요되는 단점이 있다.

이러한 단점을 보완하기 위하여 Gauss-Seidel 방법으로 처리하는 방법이 있으나 이러한 경우 영상을 반복해서 처리해야 하므로 처리시간이 많이 소요된다.

이러한 단점(수렴 속도, 전체 처리시간)을 개선하기 위하여 본 논문에서는 기존의 신호처리(Gauss-Seidel)와 제안된 알고리즘을 적용시켜 비교하여 봄으로써 특이점 혹은 악조건일 경우에도 수렴속도를 고속화하여 기존의 Gauss-Seidel 신호 처리 방법보다 처리 시간을 단축할 수 있는 영상 복원 방법을 제시하였다.

제안된 최적화 알고리즘을 영상신호에 적용시켜 가속 상수에 따른 처리 신호의 simulation과 MSE(mean-square error)의 변화를 비교하여 봄으로써 처리정도를 알아보았다.

그리고, 본 알고리즘의 유효성을 입증하기 위하여 모든 가속상수의 변화에 대한 영상복원 결과와 처리시간을 측정하여 보았다.

ABSTRACT

In this paper, the method of processing a blurred noisy image has been researched.

The conventional method of processing signal has faluts which are slow convergence speed and long time-consuming process at the singular point and/or in the ill condition.

There is the process, the Gauss-Seidel's method to remove these faults, but it takes too much time because it processed singnal repeatedly.

For overcoming the faults, this paper shows a image restoration method which takes shorter than the Gauss-Seidel's by comparing the Gauss-Seidel's with proposed alogorithm and accelerating convergence speed at the singular point and/or in the ill condition.

In this paper, the conventional process method(Gauss-Seidel) and proposed optimal algorithm were used to get a standard image(256×56×bits), and then the results are simulated and compared each other in order

* 신일전문대학 전자계산과
Dept. of Computer Science., Shiil Technical College
論文番號 : 94172
接受日字 : 1994年 6月 27日

to examine the variance of MSE(Mean Square Error) by the acceleration parameter in the proposed image restoration.
The result of the signal process and the process time was measured at all change of acceleration parameter in order to verify the effectveness of the proposed algorithm.

I. 서 론

일반적인 신호처리 방법의 공통적인 목적은 원래의 입력신호와 가장 근사한 같은 신호를 재현하는데 중심을 두고 있다.

신호는 시스템이 가지는 특성으로 인해 형성, 감시, 기록, 전송의 단계를 거치는 동안 여러원인에 의해 훼손된다.

이러한 훼손은 다음과 같이 표현된다.[1-7]

$$y = H \cdot x + n \tag{1}$$

훼손된 신호 y는 원래의 입력신호 x가 시스템함수 H에 의해 흐려지고, 잡음 n이 더하여진 형태로 나타난다.

식(1)과 같이 훼손된 신호는 보통 흐림훼손과 가산잡음에 의한 훼손으로 나뉘어져 분리 처리되고 있다.

즉, 흐림에 의한 훼손을 처리하는 역필터, Jacobi방법, Gauss Seidel방법등과, 가산잡음에 의한 훼손을 처리하는 Winear, Lee, CSW, 적응성 필터 등이 있으며, 흐림훼손과 가산잡음에 의한 훼손을 처리하는 Tikhonov Miller방법등이 있다. 기존의 Jacobi방법이나 Gauss Seidel방법은 수학적인 모델이 비교적 간단하고, 신호처리효과도 우수한 실용적인 방법이다.

그러나, 이러한 신호처리방법이 여러면에서 우수함에도 불구하고 그동안 신호처리에 적용될때 발생되는 가장 큰 단점은 신호처리시 항상 수렴하지 않을 뿐아니라 수렴시 반복처리를 필요로 하여 많은 처리시간의 소요와 식(1)과 같이 잡음이 포함되어 있는 훼손의 경우에 이 방법을 적용하였을 경우 잡음의 증폭이 발생한다.[8]

본 논문에서는 신호처리의 장점과 Tikhonov Miller 방법의 장점을 이용한 방법에 가속 상수를 도입, 기존의 여러가지 단점을 보완하고, 신호처리성능을 증가시킬 뿐만 아니라, 기존의 방법에 있어서 가장 문제시되는 처리시간을 현저히 줄일 수 있는 새로운 알고리즘을 제안하였다.

본 논문에서 제안하게 된 알고리즘은 수학적 해석과 신호처리기술에 바탕을 두며, 이로부터 유도해석된 최적와 상수를 이용하여 신호를 고속처리 가능케 한 것이다.

II. 기존 신호 처리 방법의 분석

1. Jacobi/Gauss-Seidel 처리 방법

신호가 식(1)과 같이 훼손되었을 경우, 신호처리시 역행렬이 존재하지 않으면 해를 구할 수 없다.

즉, H^{-1}인 경우, H가 특이점을 가지거나 악조건인 경우 역필터 방법으로 처리가 불가능하다. 이러한 단점을 해결하기 위하여 Jacobi 반복 처리 방법이 있는데 이 방법은 0이 아닌 대각선 요소를 갖는 행렬에 적용한다.

예를 들어 AX=B라는 식 (2)에서 Jacobi방법을 적용시킬 경우 다음과 같다.[10]

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} \begin{vmatrix} x_1 \\ x_2 \\ x_3 \end{vmatrix} = \begin{vmatrix} b_1 \\ b_2 \\ b_3 \end{vmatrix} \tag{2}$$

첫번째 반복시에는

$$\begin{aligned} x_1 &= (b_1 - a_{12}, x_2 - a_{13}, x_3)/a_{11} \\ x_2 &= (b_2 - a_{21}, x_1 - a_{23}, x_3)/a_{22} \\ x_3 &= (b_3 - a_{31}, x_1 - a_{32}, x_2)/a_{33} \end{aligned} \tag{3}$$

과 같은 식이 된다.

일반적으로 Jacobi방법의 알고리즘은

$$x_i^{(k+1)} = \frac{(b_i - \sum_{j=1}^{i-1} a_{ij} * x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} * x_j^{(k)})}{a_{ii}} \quad (4)$$

이다.[11]

그런데 Jacobi방법은 $x_2^{(k+1)}$의 해를 구할 경우 최근에 구하여진 해 $x_1^{(k+1)}$을 사용하지 않는다.

이러한 단점을 보완한 것이 Gauss Seidel방법인데 이를 수식적으로 표현하면

$$For\ i = 1, 2, 3, \cdots\cdots, n$$

$$x_i^{(k+1)} = \frac{(b_i - \sum_{j=1}^{i-1} a_{ij} * x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} * x_j^{(k)})}{a_{ii}}$$

(5)

여기에서 $x_1^{(k+1)}$가 계산과정에 포함되어 조금 더 근사한 값이 계산과정에 포함됨으로 처리 속도가 빠름을 알 수가 있다.

위의 두가지 방법에 메모리 절약과 시간 단축을 위하여 L,D,U행렬을 사용한다.

$$L = \begin{vmatrix} 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ a_{21} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & \cdot & & & \\ & & & \cdot & & & \\ & & & \cdot & & & \\ & & & \cdot & & & \\ a_{n1} & \cdot & \cdot & \cdot & \cdot & a_{nm-1} & 0 \end{vmatrix} \quad (6)$$

$$D = diag(a_{11}, \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot a_{nn}) \quad (7)$$

$$U = \begin{vmatrix} a_{12} & \cdot & \cdot & \cdot & \cdot & \cdot & a_{1n} \\ & & & \cdot & & & a_{2n} \\ & & & \cdot & & & \\ & & & \cdot & & & \\ & & & \cdot & & & a_{1-nn} \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \end{vmatrix} \quad (8)$$

L, D, U행렬을 jacobi방법에 도입하면

$$\begin{array}{ll} M_j \cdot X^{(k+1)} = & N_j \cdot X^{(k)} + b \\ M_j = D & N_j = (L + D) \end{array} \quad (9)$$

Gauss Seidel 방법은

$$M_j \cdot X^{(k+1)} = N_j \cdot X^{(k+1)} + b \quad (10)$$

여기서 $M_g = (D+L)$ $N_g = -U$가 된다.

윗 식에서 해가 얼마나 빨리 $X = A^{-1}$ B에 수렴할 수 있느냐가 중요하다. Gauss-Seidel에 의한 방법이 Jacobi의 방법보다 수렴 속도가 약간 빠를 뿐만 아니라 계산기에 적용할 경우 간단하다.[10]

2. 정칙화된 Tikhonov-Miller 방법의 분석

Gauss Seidel방법으로 신호처리시 가산잡음이 포함되어 있는 훼손신호에 사용하였을 경우 잡음의 증폭이 발생한다.

이러한 단점을 보완한것이 Tikhonov Miller 방법이다.[3]

신호처리시 대개 훼손된 신호의 잡음항 norm의 가능한 근사치를 사용하게 된다.[8]

가능한 해의 집합은 이 잡음 norm에 대한 개선도 $(y - H\hat{x})$의 norm을 제한함으로써 정의할 수 있다.

정칙화된 Tikhonov Miller방법은 식(1)에서 개선도의 norm을 구하면

$$\| y - H\hat{x} \|_R = [\ (y - H\hat{x})^t R (y - H\hat{x})]^{1/2} \le E \quad (11)$$

여기서 범위 ε은 훼손된 신호 y의 잡음항의 합과 관련된 변수이다.

필터를 통과한 신호 $L\hat{x}$의 norm은

$$\| L\hat{x} \|_S = [\ (L\hat{x})^t S (L\hat{x})]^{1/2} \le E \quad (12)$$

이고, 여기서 L은 선형 정칙화 연산자이다.

Miller의 정칙화 연구에서는 식(11)와 (12)를 결합하여

$$\Phi(\hat{x}) = \| y - H\hat{x} \|_R^2 + \alpha \| L\hat{x} \|_S^2 \le 2\varepsilon^2 \quad (13)$$

되고, 여기서 α는 정칙화변수로 고정된 값 $\alpha=(\frac{\varepsilon}{E})^2$이다.

만약 해 $\hat{x}$가 식(11)와 (12)의 범위를 만족한다면 식(13)을 만족한다.

역으로, 해 $\hat{x}$가 식(13)을 만족한다면 식(11)와 (12)도 또한 만족한다. 식 (13)을 만족하는 해중에서 적절한 해는 $\hat{x}_m$으로써 Miller의 정칙화해라 부르며, $\Phi(\hat{x})$를 최소화한다.

이 최소화 해는 다음과 같이 주어진다.

$$(H^tRH+{}_{\alpha}L^t\ SL)\ \hat{x}_m = {}_{H^t}Ry \tag{14}$$

$$\hat{x}_m = \frac{H^tRy}{H^tRH+\alpha L^t} \tag{15}$$

로 구해진다.

III. 제안된 알고리즘

Tikhonov-Miller방법은 Gauss-Seidel방법을 수정 보완한것으로 반복신호처리에 따른 처리시간이 많이 소요된다.

즉, 신호처리를 기존의 방법으로 할 경우 반복처리에 따른 처리시간이 문제가 된다. 이러한 단점을 보완하기 위하여 어떠한 경우에도 수렴속도가 늦어지지 않으며, 가산잡음이 포함된 훼손신호에 적용함에 있어서도 전체 처리시간의 현저한 감소와 신호처리성능을 대폭 개선할 수 있는 새로운 알고리즘을 제안한다.

즉, Jacobi방법은

$$(X_i)^{(k+1)}=\frac{-1}{a_{ii}}[\ b_i-\sum_{j=1,j\neq i}^{n}a_{ij}\cdot X_i^{(k)}] \tag{16}$$

$For\ i=1,2,3,\cdots\cdots,n$이다.

Gauss-seidel방법은 위의 방법을 보완한 것으로

$$a_{ii}\cdot X_i^{(k+1)}$$
$$=-\sum_{j=1}^{i-1}a_{ij}\cdot X_j^{(k+1)}-\sum_{j=i+1}^{n}a_{ij}\cdot X_j^{(k)}+b_i \tag{17}$$

$For i=1,2,3,\cdots\cdots,n$

즉, Jacobi방법의 계산속도를 빠르게 한 것이나 Gauss-Seidel방법 역시 특이점(singular point)나 악 조건(ill-condition)인 경우 수렴 속도가 매우 느려진다는 단점이 있다.

이것은 α_{ii}항인 D가 0인 (특이점)인 경우 식 (10)의 M_{ii}항이 적어져 수렴속도가 느려진다는 것이다.

이것은 식(10)에서

$$D\cdot X^{x+1}=-L\cdot X^{k+1}-U\cdot X^k+b \tag{18}$$

와 같이 된다. 여기서

D : 대각선 행렬
L : 하부 삼각 행렬
U : 상부 삼각 행렬

식 (18)를 정리하면

$$X_i^{(k+1)}=\frac{1}{a_{ii}}[\ b_i-\sum_{j=1}^{i-1}d_{ij}\cdot X_j^{(k+1)}$$
$$-\sum_{j=i+1}^{n}a_{ij}\cdot X_j^{(k)}] \tag{19}$$

$For\ i=1,2,3,\cdots\cdots,n$

으로 쓸 수 있는데 (K+1)번째의 근사해 $X^{(k+1)}$에 대하여

$$A\cdot X^{(k+1)}\ =B^{(k+1)}\text{이라하면}$$
$$R^{(k+1)}\ =B^{(k+1)}-A\cdot X^{(k)} \tag{20}$$

여기서 $R^{(k+1)}$은

$$R^{(k+1)}=[\ \ (r_1)^{k+1},(r_2)^{K+1},\ \cdot\ \cdot\ \cdot\ \cdot\ (r_u)^{(k+1)}]^t \tag{21}$$

과 같이 되는데 이것을 개선도 벡터(Residual vector)라고 한다.

$$R^{(k+1)}=B^{(k+1)}-AX^{(k)}$$

이 말의 의미는 $=B^{(k+1)}-B^{(k)}$이므로

$(k+1)$번째 신호가 (k)번째 신호보다 얼마나 개선되었는가를 나타낸다.

$R^{(k+1)}=0$이면 $X^{(k+1)}$가 정해지지만 대개는 $R^{(k+1)}=0$이 아니다.

특히 Gauss-Seidel 모델에서의 $(r_i)^{k+1}$은

$$(r_i)^{k+1} = b_i - \sum_{j=1}^{i-1} a_{ij} \cdot X_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} \cdot X^k - a_{ii} \cdot X_j^{(k)} \tag{22}$$

이므로

$$a_{ii} \cdot X_i^k + (r_i)^{k+1} = b_i - \sum_{j=1}^{i-1} a_{ij} \cdot X_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} \cdot X_j^k \tag{23}$$

(19)식과 (23)식에서

$$a_{ii} \cdot X_i^k + (r_i)^{k+1} = a_{ii} \cdot X_i^{k+1} \tag{24}$$

$$즉, X_i^{k+1} = X_i^k + \frac{(r_i)^{k+1}}{a_{ii}} \tag{25}$$

개선도 벡터가 0일 경우에 반복이 멈추어 진다. 그러나 보통 이 값은 0이 아니므로 수렴성의 확실성을 희박하게 한다. 그러므로 여기서 개선도 벡터(Residual vector) R^{k+1}의 노름을 감소시키기 위하여 식 (25)을 수정하여

$$(X_i)^{k+1} = (x_i)^k + W \cdot \frac{(r_i)^{k+1}}{a_{ii}} \tag{26}$$

로 하여 W의 값에 따라 수렴속도를 조절할 수 있는 방법을 제시한다. W의 도입은 $(r_i)^{(k+1)}$의 개선도 vector가 실제로 0이 안됨으로 수렴성을 가속화하기 위한 상수값이다.

여기서 W값을 구하기 위하여

$$\rho(A) = \max \{ |\lambda| \} \tag{27}$$

와 같이 정의한다.

단, λi는 A의 고유치이고 만약 λ=a+bi이면 $|\lambda_i| = \sqrt{a^2+b^2}$으로 정의하면

n차 정방 행렬

$$A = \begin{vmatrix} a_{11} & a_{12} & \cdot & \cdot & \cdot & \cdot & a_{1n} \\ & & & \vdots & & & \\ a_{n1} & \cdot & \cdot & \cdot & \cdot & \cdot & a_{nm} \end{vmatrix} \text{에서}$$

$$L = \begin{vmatrix} 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ a_{21} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & \vdots & & & & & \\ a_{n1} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{nn-1} & 0 \end{vmatrix},$$

$$D = \begin{vmatrix} a_{11} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ a_{22} & & & \cdot & \cdot & & & \\ & & & \vdots & & & & \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{nn} \end{vmatrix}$$

$$U = \begin{vmatrix} 0 & \cdot & \cdot & \cdot & \cdot & \cdot & a_{1n} \\ & & & \vdots & & & \\ 0 & \cdot & \cdot & \cdot & a_{n-1n} & & \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \end{vmatrix}$$

$$A = L + D + U \tag{28}$$

$$T = D^{-1}(L + U) \text{라 할 때} \tag{29}$$

$$W = \frac{2}{1+\sqrt{1-\rho(T)^2}} \tag{30}$$

로 하면 수렴 속도가 가장 빠른것이 증명되어 있다.[11,12]

여기에서 ρ(T)(행렬 T의 최대반경)를 구할시에는 실수 요소로 된 n차 정방행렬 A의 고유치 $(\lambda_i)_{i=1}^{n}$에 대하여

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq + \cdot \cdot \cdot \geq |\lambda_n| \tag{31}$$

를 가정하고 A는 1차 독립인 고유벡터 $(U_i)_{i=1}^{n}$을 갖는다고 하자.

또 n차 열 벡터 X_0를

$$X_0 = \alpha_1 U_1 + \alpha_2 U_2 + \alpha_3 U_3 + \cdots\cdots + \alpha_n U_n \qquad (32)$$

이라 하고

$$AU_i = \lambda_i U_i \quad \text{에서}$$

$$A(AU_i) = A\lambda_i U_i = \lambda_i(AU_i)$$

$$\text{즉, } A^2 U_i = (\lambda_i)^2 U_i \qquad (33)$$

$$A^3 U_i = (\lambda_i)^3 U_i$$

$$\vdots$$

$$A^k U_i = (\lambda_i)^k U_i \text{이며,}$$

다음과 같은 수열

$$X_0 = A$$

$$X_1 = AX_0$$

$$X_2 = AX_1, \ (X_2 = A^2 X_0) \qquad (34)$$

$$\vdots$$

$$X_k = AX_{k-1}, \ (X_k = A^k X_0)$$

를 만들면 식 (32), (33), (34)에서

$$X_k = A^k \cdot X_0$$

$$= \alpha_1 A^k U_1 + \alpha_2 A^k U_2 + \cdots + \alpha_n A^k U_n$$

$$= \alpha_1 (\lambda_1)^k U_1 + \alpha_2 (\lambda_2)^k U_2 + \cdots + \alpha_n (\lambda_n)^k U_n$$

$$\vdots$$

$$= (\lambda_1)^k \left(\alpha_1 U_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k U_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k U_n\right) \qquad (35)$$

가 된다.

식 (34)의 조건하에서 K가 증가하면

$X_k = (\lambda_1)^k \alpha_1 \cdot U_1$이 된다.

또 $X^{k+1} = (\lambda_1)^{k+1} \alpha_1 U_1$ 또는 $X^{k+1} = (\lambda_1)^k (\lambda_1) \cdot \alpha_1 U_1 = (\lambda_1)^k X_k$이므로 이것을 이용하여 λ_1을 구할 수 있다.

그러나, $(\lambda_1)^k = \dfrac{X^{k+1}}{X^k}$에 의하여 λ_1을 구하려면 (36)

매 단계마다 벡터계산이 필요하다. 이런 점을 제거하기 위하여 예를 들어,

$$X_k^t \cdot X_{k+1} = (\lambda_1)^k \cdot X_k^t \cdot X_k \text{이므로}$$

(X_k^t는 X_k의 전치벡터)

$\lambda_1 = \dfrac{X_k^t X_{k+1}}{X_k^t X_k}$로써 구하면 scalar 계산이 된다.

또 X_{k+1}의 절대치 최대요소를 X_k의 절대치 최대요소로 구하면 λ_1을 구할수 있으나 이 방법은 매 단계마다 절대치 최대요소를 판별해서 구해야 하고 또 $|\lambda_1| > 1$이면 k가 증가 할수록 X_k로 커지고 $|\lambda_1| < 1$이면 k가 감소하여 자칫 컴퓨터의 계산범위를 넘을 수도 있다.

이상의 문제점을 해결하기 위하여

$X_0^t X_0 = 1$ (즉 $\| X_0 \| = 1$) 인 초기치 X_0를 선정하고 매 단계마다 l_2 norm 을 이용한 정규화(normalization) 또는 scaling을 통하여 X_k를 구하여 식(36)으로써 scalar계산을 통하여 λ_1을 구한다.

이 알고리즘은

(1) $X_0^t X_0 = 1$인 초기치 벡터X_0 선정

(2) $i = 1$

(3) $Y_i = AX_{i-1}$

(4) $\beta_i = X_{i-1}^t Y_i$ (고유치λ_1)

(5) $\eta_i = \sqrt{Y_i^t Y_i}$ (l_2 *norm*)

(6) $X_i = \dfrac{Y_i}{\eta_i}$ (X_i는$\| X_i \| = 1$인 고유벡터)

(7) 수렴판정, λ_i 이 수렴하면 끝.

(8) $i=i+1$ $(i+=1)$

(9) (3)부터 반복

알고리즘에서 계산되는 β_i는

$$\beta_i = \frac{X^t_{i-1} Y_i}{X^t_{i-1} X_{i-1}}$$ 이어야 한다.

여기서 구한 β_i값을 $\rho(T)$에 대입한다.

여기에서 구한 W와 식 (26)와 식 (22)에서 신호처리에 적용될 최적화 알고리즘의 식을 구하면

$$(\hat{x})^{k+1} = (1-W)\cdot(\hat{x})^k + W[\ \hat{x}^k - W_2\cdot(y-H\ \hat{x}^k)] \tag{37}$$

이다.(여기서 W_2위의의 상수)

이 방법은 수렴 속도가 늦은 Gauss-Seidel방법을 개선하기 위하여 최적화 한 것이다.

(37)와 (17)을 비교하여 보면 $(1-W)\cdot(X_i)^k$ 와 W, W_2가 사용되었다.

이 W는 상수값으로 주어지므로 연산 속도에는 크게 미치지는 못하나 $(1-W)\cdot(X_i)^k$때문에 같은 반복횟수에 비하여 처리속도가 가속화되고 특이점, 악조건 W는 상수이므로 빨리 수렴하게 된다.

반복 Tikhonov-Miller방법의 알고리즘 역시 Gauss-Seidel방법을 수정보완한 것이므로 가속상수의 도입과 신호처리의 최적화가 가능하다.

즉, 반복 Tikhonov-Miller방법의 장점을 도입하기 위해 식 (15)를 연속적으로 근사화함으로써

$$\hat{x}_0 = H^t\cdot y \tag{38}$$

$$x^{(k+1)} = (I-\alpha L^t L)\ \hat{x}^k + \beta H^t(y-H\ \hat{x}^k) \tag{39}$$

와 같이 구해진다.

여기서 전체 처리시간과 기억 용량의 증가함이 없이 특이점이나 악조건일 경우에도 수렴성을 보장받고, 가산잡음에 대해서도 처리정도를 현저히 개선시킬 뿐만 아니라 처리시간상의 문제점을 해결하기 위하여 식(37)과 (39)를 정리하여 최적화하면 다음과 같다.

$$x^{(k+1)} = (1-W_1)\hat{x}^k + W_1[\ (I-\alpha L^t L)\hat{x}^k + \beta H^t(y-H\ \hat{x}^k)] \tag{40}$$

여기서 β값은 전체연산속도를 최적화하는 값을 사용하였다.

여기서 W_1과 β는 상수로써 전체 연산속도에는 영향을 주지 않는다.

Ⅳ. 시뮬레이션 결과 및 고찰

본 논문에서 제안한 알고리즘의 유효성을 확인하기 위하여 영상신호처리에 적용시켜 보았다.

사용된 영상신호는 256×256 해상도 및 256의 계조도를 갖는 표준 영상을 사용하였다.

영상을 처리할 경우, 영상처리장치의 불완전함에 의해 흐려지고 가산잡음에 의해 훼손되어 지는데, 가산잡음은 영상의 형성, 전달, 저장과정에서 임의로 영상에 섞이게 된다.

본 논문에서는 표준 원 영상신호를 점확산 함수로써(PSF)로써 흐려지게 하고 흐려진 영상에 가우스 분포를 갖는 백색 잡음(Gaussian White Noise)를 가산하였다. 이 때 흐려진 영상의 신호 대 잡음비 BSNR의 식은

$$BSNR = \frac{\underline{variance\ of\ the\ blurrel\ imaged}}{varriance\ of\ the\ additive\ noise} = \frac{\sigma^2 H\cdot x}{\sigma^2 n}$$

(41)

과 같다.

본 논문에서 사용된 가산잡음은 20dB의 신호 대 잡음비를 가지는 백색잡음을 사용하였다.

다음과 같이 정의한다.

$$MSE = \frac{1}{M*N}\sum_{i=1}^{N}\sum_{j=1}^{M}(X(i,j)-X(i,j))^2 \tag{42}$$

훼손된 영상의 MSE는 268.42588이다.

표(1)과 그림(1)은 Tikhonov-Miller방법과 제안된 알고리즘의 평균 자승오차(MSE)를 시간에 대해 비교하여 보았다.

제안된 알고리즘에서 사용된 가속상수는 W_1=0.5 β = 9.0값을 사용하였으며, IBM PC 486기종을 사용하여 시뮬레이션한 처리시간을 측정, 비교하였다. 보다 나은 처리속도를 가진 컴퓨터를 사용한다면, 처리시간은 단축될 수 있을 것이다.

표 1. 처리시간에 대한 평균자승오차(MSE)의 비교
TABLE 1. Compared MSE for processed time

반복횟수	기존의 방법 Tikhonov_miller	제안된 알고리즘 proposed	처리시간 (sec)
1	170.51864	153.78665	9.0
2	161.51880	162.10504	18.0
3	158.12446	166.82234	27.0
4	157.05715	170.49918	35.0
5	157.03966	173.55008	44.0
6	157.51382	176.21303	54.0
7	158.21780	178.62766	63.0
8	159.02305	180.91017	72.0
9	159.86465	183.09803	80.0
10	160.70941	185.21865	89.0

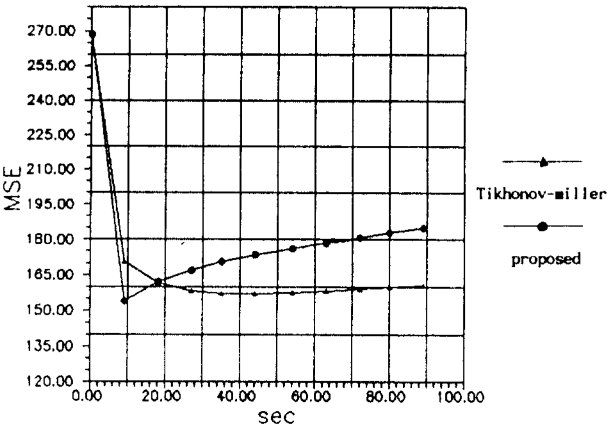


그림 1. 처리시간에 대한 평균자승오차(MSE)의 비교
Fig. 1. Compared MSE for processed time

또 가속상수 W_1, β에 따른 영상신호의 오차와 처리시간을 전 구간에 걸쳐서 모두 구해본 결과 제안된 최적화 알고리즘의 유효성을 확인할 수 있었다.

표 2. 기준시간 (9초) 에서의 가속상수에 따른 영상신호처리의 처리 오차(MSE) 비교
Table 2. Compared MSE of image singal process by the standard time (9sec)

W_1 / β	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
0.5	186.86985	180.19097	174.84111	170.51864	167.01668	164.17286	161.86176	159.98537	158.46642	157.24340
1.0	180.19097	170.51864	164.17286	159.98537	157.24340	155.49749	154.45664	153.92281	153.78665	153.89608
1.5	174.84111	164.17286	158.46642	155.49749	154.13497	153.78665	154.04498	154.78541	155.80466	157.08779
2.0	170.51864	159.98537	155.49749	153.92281	153.89608	154.78541	156.20756	158.05611	160.23810	162.73122
2.5	167.01668	157.24340	154.13497	153.89608	156.08001	157.08709	159.66322	162.73122	166.33662	170.63282
3.0	164.17286	156.49749	153.76665	154.76541	157.08709	160.23810	164.10152	168.82265	174.79666	183.35735
3.5	161.86176	154.45664	154.04498	156.20756	159.66322	164.10152	169.70872	177.24789	189.80166	538.15806
4.0	159.98537	153.92281	154.76541	158.05611	162.73122	168.82265	177.24789	192.84519	1111.25856	4803.56830
4.5	158.46642	153.78665	155.80465	160.23810	166.33662	174.79666	189.80166	1111.25856	5067.25632	6487.25064
5.0	157.24340	153.89608	157.08709	162.73122	170.63282	183.36736	538.15806	4803.56836	6487.25064	7480.50009
5.5	156.26702	154.24336	158.57189	165.56573	175.98392	201.96784	3906.80349	6344.73474	7456.31082	8101.65146
6.0	155.49749	154.78641	160.23810	168.82265	183.36736	1111.25856	5878.96930	7174.98876	7982.12342	8681.88336
6.5	154.90248	155.42832	162.07856	172.61098	196.60086	4412.28207	6588.24087	7888.63520	8444.75453	9026.45425
7.0	154.45664	156.20756	164.10152	177.24789	538.15806	5878.96920	7306.60006	8273.59780	8897.35626	9387.18410
7.5	154.13497	157.08709	166.33662	183.36736	2726.76296	6487.25064	7849.54623	8681.88336	9242.72364	9732.87820
8.0	153.92281	158.05611	168.82265	192.84519	4803.56836	7174.98876	8273.59780	8974.63294	9637.66796	9997.38162
8.5	153.80417	159.10770	171.50856	244.41066	6106.01985	7662.66609	8650.03100	9297.15070	9806.61678	10231.06460
9.0	153.78665	160.23810	174.79566	1111.25856	6487.25064	7982.12342	8897.35626	9637.66796	10022.99088	10124.52865
9.5	153.79387	161.44581	178.59949	3148.29857	7006.14725	8379.56146	9223.74886	9772.98622	10211.45204	10637.46374
10.0	153.88508	162.73122	183.36736	4803.56836	7480.50009	8681.88336	9387.18410	9997.38162	10424.52865	10802.15829

W_1 / β	5.5	6.0	6.5	7.0	7.5	8.0	8.5	9.0	9.5	10.0
0.5	156.26702	155.49749	154.90248	154.45664	154.13497	153.92280	153.80417	153.78665	153.79087	153.88608
1.0	154.24336	154.78641	155.42832	156.20756	157.08709	158.05611	159.10770	160.23810	161.44581	162.73122
1.5	158.57180	160.23810	162.07856	164.10152	166.33662	168.82265	171.50855	174.79566	178.59949	183.36736
2.0	165.56573	168.82265	172.61098	177.24478	183.36736	192.84519	244.41056	1111.25856	3148.29857	4803.50836
2.5	175.96392	183.35736	196.60086	538.15806	2726.76298	4803.56836	6016.01985	6487.25064	7006.14725	7480.50009
3.0	201.96784	1111.25856	4412.29207	5878.96930	6487.25064	7174.98876	7662.66689	7982.12342	8379.56146	8681.88336
3.5	3906.80349	5878.98920	6588.24067	7306.60006	7849.54623	8273.59780	8650.03100	8907.36626	9223.74806	9387.18410
4.0	6344.73474	7174.98876	7888.63620	8273.59780	8681.88336	8971.63294	9297.15070	9637.66796	9772.98622	9997.38162
4.5	7456.31682	7982.12342	8444.75483	8897.35626	9242.72364	9637.66736	9806.61678	10022.99088	10211.45204	10424.52865
5.0	8101.65146	8681.88336	9026.45425	9387.18410	9732.87820	9997.38162	10231.06460	10424.52865	10637.46374	10802.15829
5.5	8732.28528	9144.09202	9473.57029	9827.31778	10117.90338	10426.98203	10568.94439	10775.46182	10992.17904	11129.78573
6.0	9144.09292	9637.66736	9988.15336	10157.33563	10424.52866	10708.72712	10880.58638	11051.65751	11196.93147	11363.90201
6.5	9473.57029	9988.15336	10171.98242	10475.76879	10746.63806	10964.12254	11160.97171	11300.98646	11452.68787	11598.52747
7.0	9827.31778	10157.33563	10475.76879	10729.85202	11003.86775	11201.22864	11343.18730	11513.76325	11611.45678	11740.79777
7.5	10117.90338	10424.52866	10746.63806	11003.86775	11153.78630	11363.90201	11555.94291	11662.99243	11759.86518	11887.68434
8.0	10426.98203	10708.72712	10964.12254	11201.22864	11363.90201	11564.14283	11656.74299	11775.98664	11927.22772	12036.77147
8.5	10558.94439	10880.58638	11160.97171	11343.18730	11556.94291	11656.74299	11702.27080	11916.15720	12065.63457	12137.08606
9.0	10775.46182	11051.65951	11300.98646	11513.76325	11662.99243	11775.98664	11916.15720	12072.17172	12143.20654	12227.58408
9.5	10992.17934	11196.93147	11452.68767	11611.45678	11759.88618	11927.22772	12065.63457	12143.20654	12229.86166	12343.32336
10.0	11129.78573	11363.90200	11598.52747	11740.79777	11887.68434	12036.77147	12137.08506	12227.58408	12343.32335	12114.42587

가속상수와 반복처리시간에 따른 전체처리결과를 구하였으나 너무 방대한 양이므로 기준시간(9sec)에서의 가속변수에 따른 영상신호처리의 처리오차(Mean Square Error)를 그림(2)에서 도식화하고 표(2)에 자료를 정리하였다.

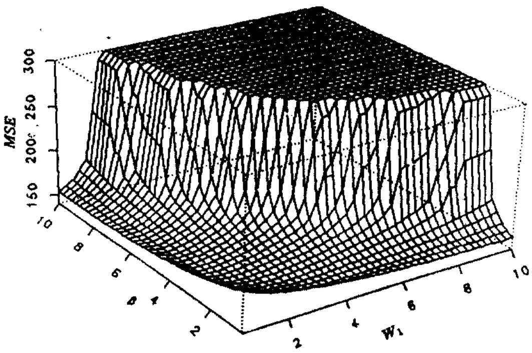


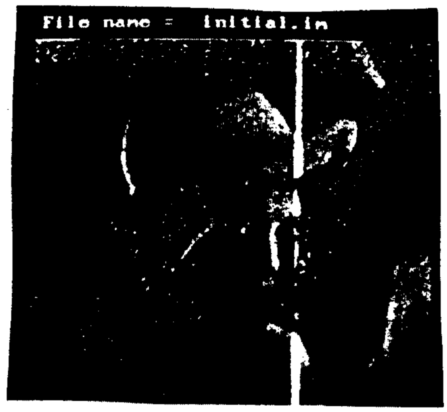
그림 2. 기준시간(9초)에서의 가속상수에 따른 영상신호처리의 처리오차(MSE)비교
Fig. 2. Compared MSE of image singal process by the standard time (9sec)

그림 [3]은 원영상과 훼손된 영상을 나타내었다.

여기서 (c)는 Tikhonov-Miller의 초기근사해를 위해 초기화한 영상이다.

(a) original image

(b) oserved image(blurred-noisy image)



(c) initinal image

그림 3. 원 영상과 훼손된 영상
Fig. 3. Original image and noisy-blurred image for BSNR=20dB

그림 [4]는 기존의 반복 Tikhonov-Miller방법을 사용하여 처리영상을 반복횟수(K)에 따라 나타내었고 그림 [5]는 제안된 방법에 의해 처리된 영상을 나타내었다.

(a) k=1

(b) k=2

(c) k=3

(d) k=4

(e) k=5

(f) k=6

(g) k=7

(h) k=8

(i) k=9

(j) k=10

그림 4. 기존의 반복 Tikhonov-Miller 방법에 의한 복원영상(k: 반복횟수)

Fig 4. Restored image by conventional(Tikhonov-Miller) iterative method (k: iterative numbers)

그림 5. 제안된 방법에 의한 복원영상

Fig. 5. Restored image by proposed method

V. 결 론

신호는 그 형성, 감시, 기록, 전송의 단계를 거치는 동안 여러가지 원인에 의해 훼손되며, 훼손된 신호를 처리하여 원래의 입력신호와 가장 근사한 신호를 재현하는 기존의 많은 복원 방법은 여러가지 단점들이 발생하였다.

기존의 처리 방법들, 즉 가산잡음에 의해 훼손된 신호를 처리하는 여러가지 방법들은 선형적 흐림에 의한 훼손을 처리하지 못함으로써 실제 응용에 많은 제약들이 있으며, 흐림에 의한 훼손을 처리하는 반복 복원방법들은 가산잡음이 있을 시 그 복원과정에서 가산잡음의 증폭을 초래하여 오히려 더 훼손된 결과를 가져온다.

즉, 흐림 훼손과 가산잡음에 의한 훼손은 복원시 서로 상보적인 관계를 가진다. 또한, 흐림 훼손 신호를 반복복원 할 때 악조건이나 특이점을 가질 경우 연산이 수렴하지 않는 단점을 가지고 있으며, 이러한 악조건과 특이점 문제를 해결하기 위한 공정이 필요하다.

그러나, 이러한 공정은 전체 처리시간을 증가시키는 문제에서 단점을 가져온다. 본 논문에서 제안한 알고리즘은 흐림훼손과 가산잡음이 있는 실제의 신호를 처리함에 있어 악조건이나 특이점에서도 수렴하는 Tikhnov-miller의 반복 복원 방법에 새로운 두 가지 개념을 도입하여 반복처리를 하지 않고 직접적으로 수렴함으로써 더 적은 신호처리시 걸릴 부계산이 되어 총 처리시간을 해결하였으며, 가산잡음에 의한 훼손도 현저히 감소시킨 방법이다. 본 논문에서 제시된 수학적 알고리즘은 기존의 알고리즘에 W_1, β의 값만을 부가적으로 사용하였다.

이것은 기존의 알고리즘의 모든 장점들, 가산잡음에 대한 처리, 특이점이나 악조건에서의 수렴성 등을 그대로 간직한채 상수만을 사용함으로써 메모리의 증가를 필요로 하지않고 반복처리시의 단점인 전체 처리시간을 고속화시킨 방법이다.

제안된 알고리즘과 Tikhonov miller방법을 시스템 함수에 의해 흐려지고 20dB의 가산잡음이 있는 영상신호를 복원하는데 비교해 본 결과 전체 처리시간의 급격한 감소와 객관적인 평가 척도인 평균 자승 오차(MSE)도 매우 좋은 유용한 방법임을 확인할 수 있었다.

참 고 문 헌

[1] H. C. Andrews and B. R. Hunt, Digital Image Restoration. Englewood Cliffs,NJ:Prentice Hall, 1977.

[2] J. Biemond, F. G. Von der Putten, and J. W. Woods, "Identification and restoration of images with symmetric non causal blur," IEEE Trans. Circuits Syst., vol. CAS-35, pp 385-394, 1988.

[3] P. H. Westerink, J. Biemond, and P. H. L. de Bruin, "Digital color imge restoration," in Signal Processing III:Theories and Applications, I. T. Young et al., Eds. Amsterdam, The Netherlands:Elsevier North Hollands, 1986, pp. 761-764

[4] A. L. Steven, W. Zucker, and A. Rossenfeld, "Iterative enhancement of noisy images":IEEE Trans. Systems, Man and Cybernetics SMC-7, 1977, pp. 435-442.

[5] H. J. Trussell and M. R. Civanlar, "The feasible solution in signal restoration," IEEE Trans. Acoust., speech, signal Processing, vol. ASSP-32, pp. 201-212, 1984.

[6] A. K. katsaggelos, J. Biemond, R. M. Mersereau, and R. W. Schafer, "A general formulation of constrained iterative image restoration algorithms," in Proc. IEEE Int. conf. Acoust., Speech, Signal Processing 1985, Tmapa, Fl, 1985. pp. 700-703.

[7] K. Miller, "Least squares methods for ill posed problems with a prescribed bound," SIAM J. Math., Anal., vol. 1, pp. 52-74, 1970.

[8] Reginald L. Lagendijk and Jan Biemond, "Regularized iterative image restoration with ringing reduction," IEEE Trans. ASSp, vol 36, NO. 12, December 1988.

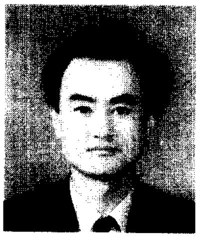
[9] A. N. Tikhonov and V. Y.. Arsenin, solutions of Ill posed problems. New York:Willey, 1977.

[10] Lee W. Johnson, R. Dean Riess:"Numerical Analysis", Addison-Wesley Publishing Company, 1982.

[11] Richard L. Burden, J. Douglas Faires, Albert C. Reynolds:"Numericla Analysis", Prindle, Weber & Schmidt Boston, Massachussetts, 1981.

[12] Melvin J. Maron:"Numerical Analysis", Macmillan publishing Co., Inc. New York..

[13] Francis Sc[illegible]d:"Numerical Analysis", Schaum's outline ser[illegible]



權起弘(Kee Hong Kwon) 정회원
1989년 : 영남대학교 공과대학 전자공학과 학사
1991년 : 영남대학교 공과대학 전자공학(전자통신 전공)석사
1991년 : 영남대학교 공과대학 전자공학(전자통신 전공) 박사과정 입학
1991년 ~ 현 : 신일전문대학 전자계산과 전임강사 재직