AN AFFINE SCALING ALGORITHM FOR MINIMIZING TOTAL VARIATION IN IMAGE ENHANCEMENT

YUYING LI* AND FADIL SANTOSA†

Abstract. A computational algorithm is proposed for image enhancement based on total variation minimization with constraints. This constrained minimization problem is introduced by Rudin et al [13, 14, 15] to enhance blurred and noisy images.

Our computational algorithm solves the constrained minimization problem directly by adapting the affine scaling method for the unconstrained l_1 problem [3]. The resulting computational scheme, when viewed as an image enhancement process, has the feature that it can be used in an interactive manner in situations where knowledge of the noise level is either unavailable or unreliable.

This computational algorithm can be implemented with a conjugate gradient method. It is further demonstrated that the iterative enhancement process is efficient.

Key Words. image enhancement, image reconstruction, deconvolution, minimal total variation, affine scaling algorithm, projected gradient method

* Department of Computer Science and Advanced Computing Research Institute, Cornell University, Ithaca, NY 14853, yuying@cs.cornell.edu. Research partially supported by the Applied Mathematical Sciences Research Program (KC-04-02) of the Office of Energy Research of the U.S. Department of Energy under grant DE-FG02-90ER25013.A000, and in part by NSF, AFOSR, and ONR through grant DMS-8920550, and by the Cornell Theory Center which receives major funding from the National Science Foundation and IBM corporation, with additional support from New York State and members of its Corporate Research Institute.

† Department of Mathematical Sciences and Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, santosa@math.udel.edu. Research partially supported by the Air Force Office of Scientific Research under grant F49620-93-I-0500, the Department of Energy under grant DE-FG02-94ER25225, the National Science Foundation under grant DMS-9210489.

1. Introduction. Let $u_{\text{true}}(x,y)$ specify the grey levels of an unknown black-and-white image in $\Omega \stackrel{\text{def}}{=} [0,a] \times [0,b]$. Assume that we are given an initial image $u_{\text{init}}(x,y)$, which represents the blurred and noisy version of some true image $u_{\text{true}}(x,y)$, i.e.,

$$u_{\text{init}}(x,y) = (Au_{\text{true}})(x,y) + \varepsilon(x,y), \tag{1.1}$$

where $(Au)(x,y)$ denotes a blurring convolution operation and $\varepsilon(x,y)$ a random noise. The objective is to recover an image $u(x,y)$ that is as close to the true image $u_{\text{true}}(x,y)$ as possible.

In [14], Rudin et al suggest an image enhancement method by solving a constrained minimization problem:

$$\begin{aligned} &\min_u \int_\Omega \sqrt{u_x^2 + u_y^2}\, dxdy \\ \text{subject to} \quad &\int_\Omega ((Au)(x,y) - u_{\text{init}}(x,y))^2\, dxdy = \sigma^2, \end{aligned} \tag{1.2}$$

where the subscripts x and y denote the corresponding partial differentiation. The functional being minimized in (1.2) is the *Total Variation* of an image $u(x,y)$:

$$\text{TV}(u) \stackrel{\text{def}}{=} \int_\Omega \sqrt{u_x^2 + u_y^2}\, dxdy. \tag{1.3}$$

A solution of (1.2) provides an image with the least total variation among all the images with the standard deviation σ. The original method [13] was developed to sharpen images that have been blurred. It is important to note that the functional being minimized is the integral of the 2-norm of the gradient $[u_x; u_y]$. It is well known that if we wish to have a smooth image, the appropriate functional is the integral of the square of the 2-norm of the gradient. Others [11, 16, 17] have suggested the integral of the square of the Laplacian as the cost functional.

Osher and Rudin [13] propose to solve the minimization problem (1.2) by computing the solution of the associated Euler-Lagrange equation. The computational procedure is based on finite differences and seeks the steady state of a nonlinear diffusion equation with constraints. The enhancement technique has also been applied to remove noise from an image [13, 15], and to simultaneously deblur and denoise [14]. Numerous examples given in these papers [13, 14, 15] provide convincing evidence that the method works effectively in many situations. A comparison of this image enhancement method with other methods is given in [15]. Dobson and Santosa [5] investigated the theoretical limitations of such a method.

The main objective of our work is to develop an efficient computational algorithm for the minimization problem (1.2). In §2, we briefly describe the computational algorithm used in [13, 14, 15] and discuss its limitations. This is followed, in §3, by a motivation for our minimization method which solves a discretized problem directly. In §4, we describe an affine scaling method, which is a generalization from the method proposed by Coleman and Li [3] for an unconstrained l_1 problem. The resulting iterative method can be regarded as an image enhancement procedure. Every iteration produces an image that has either less noise or less blurring. Based on this incremental nature, we propose in §5 to use our computational method in an interactive manner, suited for the situation when knowledge about the size of a random noise σ is unavailable or unreliable. Computational experience is reported in §6.

2. A Nonlinear Diffusion Approach. In [13, 14, 15], the constrained nondifferentiable minimization problem (1.2) is solved by computing the steady state of a nonlinear partial differential (diffusion) equation. Based on the Euler-Lagrange equation:

$$0 = \frac{\partial}{\partial x}\left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}}\right) + \frac{\partial}{\partial y}\left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}}\right) - \lambda A^*(Au - u_{\text{init}}), \tag{2.1}$$

with the homogeneous Neumann boundary condition on $u(x, y)$, (1.2) is solved by computing the steady state of the nonlinear diffusion equation for $u(x, y, t)$

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}}\right) + \frac{\partial}{\partial y}\left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}}\right) - \lambda A^*(Au - u_{\text{init}}). \tag{2.2}$$

The operator A^* is the adjoint of A; λ is the Lagrange multiplier and must be chosen so that the constraint $\int_\Omega (Au - u_{\text{init}})^2\, dxdy = \sigma^2$ is satisfied. An artificial time evolution process is introduced. Numerical implementation in [13, 15, 14] is carefully designed, using finite difference scheme, to step forward in time. If the constraint $\int_\Omega (Au - u_{\text{init}})^2\, dxdy = \sigma^2$ is satisfied at each iteration, then the procedures in [13, 15, 14] correspond to the classical Rosen's projected gradient method [12].

There are a few difficulties with this artificial time evolution method for (1.2). Firstly, nondifferentiability occurs when $u_x = u_y = 0$. In [13, 15, 14], a small perturbation is used to avoid nondifferentiability numerically but this can lead to very slow convergence, and possibly represent a significant alteration to the original cost functional. Secondly, there is no guarantee that the constraint $\int_\Omega (Au - u_{\text{init}})^2 = \sigma^2$ will be satisfied. Thirdly, it is difficult to choose time steps for both efficiency and reliability.

Attempts have been made to overcome these difficulties in minimizing total variation. In [7], an active-set type method is proposed to overcome nondifferentiability. The method applies only to the case when A is the identity, i.e., when only random noise is present. Another approach is to modify the cost functional to make it differentiable. Vogel [18] solves the perturbed problem

$$\int_\Omega \sqrt{u_x^2 + u_y^2 + \beta^2}\, dxdy,$$

where β is a small parameter. This modified problem is solved using a fixed-point method [18].

We develop an efficient and reliable computational algorithm for the constrained minimization problem using an affine scaling strategy. Moreover, our computational scheme, when viewed as an image enhancement process, has the added feature that it can be used interactively in situations when the information about the variance in the noise is unreliable or unknown.

Next we describe a discretization for (1.2) and motivate our computational algorithm.

3. Solving A Discretized Minimization Problem. Instead of a continuous picture plane $\Omega = [0, a] \times [0, b]$, we consider an array of pixels with m columns and n rows. We use i for the horizontal index and j for the vertical, counting from the upper lefthand corner. The grey level value on pixel (i, j) is U_{ij}. Each rectangular pixel has width a/m and height b/n. The total variation of a image $u(x, y)$ is therefore given by

$$\text{TV}(u) = \sum_{j=1}^{n}\sum_{i=1}^{m-1} \frac{b}{n}|U_{i+1,j} - U_{i,j}| + \sum_{i=1}^{m}\sum_{j=1}^{n-1} \frac{a}{m}|U_{i,j+1} - U_{i,j}|. \tag{3.1}$$

Our subsequent presentation follows many Matlab [10] notations. Let an mn-by-1 vector u denote an m-by-n image matrix U. More precisely, in Matlab, $u = U(:)$. We can calculate the total variation of a discrete image vector u by computing the l_1 norm

$$\|Bu\|_1 \stackrel{\text{def}}{=} \sum_{j=1}^{n}\sum_{i=1}^{m-1} \frac{b}{n}|U_{i+1,j} - U_{i,j}| + \sum_{i=1}^{m}\sum_{j=1}^{n-1} \frac{a}{m}|U_{i,j+1} - U_{i,j}|$$

where B is a matrix of size $((m-1)n + m(n-1)) \times mn$ corresponding the the difference operations on u in (3.1). Note that $\text{TV}(u)$ is not differentiable whenever a component of Bu is zero.

It is worth noting that our discretization can be given another interpretation. Instead of assuming $u(x, y)$ to be piecewise constant, we work with its sample values at the pixel centers $u_{i,j} \stackrel{\text{def}}{=} u(x_i, y_j)$. Then the total variation $\int_\Omega \sqrt{u_x^2 + u_y^2}\, dxdy$ can be viewed as having been replaced by

$$\int_\Omega |u_x| + |u_y|\, dxdy.$$

This is a small departure from the original total variation but we expect it to have little effect on image enhancement. The main advantage to this choice of discretization is that the objective function of a discretized problem is piecewise linear instead of piecewise quadratic.

Similarly, the vector $u_{\text{init}} \in I\!R^{mn}$ denotes the initial m-by-n image. Assume that A is an mn-by-mn matrix in $I\!R^{mn\times mn}$ and Au is a discretized approximation to $(\mathcal{A}u)(x, y)$. For a convolution operator, A is symmetric.

Instead of the equality constraint (1.2), we propose to relax it to an inequality constraint. Specifically, we solve the following constrained linear l_1 problem

$$\min_{u\in I\!R^{mn}} \|Bu\|_1 \quad \text{subject to} \quad \|Au - u_{\text{init}}\|_2 \le \sigma. \tag{3.2}$$

The replacement of the equality constraint by an inequality constraint is certainly consistent with the objective of minimizing the total variation. Computationally, however, the formulation (3.2) is preferable since a nonlinear **equality** constraint is difficult to follow. Our numerical experience indicates that the images of least total variation are often found on the boundary of the feasible set, which implies that the original problem has been solved.

Problem (3.2) is a linear l_1 minimization with a single quadratic constraint. The nondifferentiability is inherent in both formulations, (1.2) and (3.2), but the discretized problem of (3.2) has a linear objective function instead of nonlinear. The difficulties due to nondifferentiability of the 1-norm and the quadratic constraint call for a special algorithm. Computationally, solving (3.2) presents a great challenge since the discretized problem (3.2) is typically very large (e.g., B is 130816-by-65536 for a 256-by-256 image), A may be dense due to convolution, and A may not be explicitly given. We assume that the product Au is computable for any u. We note however that even this computation can be very expensive for a convolution.

To simplify the constrained minimization problem (3.2), we reformulate it using an exact penalty function approach. Specifically, we solve

$$\min_{u\in I\!R^{mn}} \psi(u) \stackrel{\text{def}}{=} \|Bu\|_1 + \mu \max(0, \|Au - u_{\text{init}}\|_2^2 - \sigma^2), \tag{3.3}$$

where μ is a large positive constant. For a sufficiently large μ, (3.3) and (3.2) are equivalent, e.g., [6].

Solving problem (3.3) typically involves generating a sequence of approximate solutions $\{u_k\}$ with the property that

$$\psi(u_{k+1}) < \psi(u_k). \tag{3.4}$$

In the parlance of image enhancement, (3.4) implies that either the new image u_{k+1} has less noise or less blurring or both. This is certainly a desirable property for an image enhancement process.

A new image u_{k+1} can be obtained by computing a descent direction s_k for $\psi(u)$ at u_k and performing a line search

$$u_{k+1} = u_k + \alpha_k s_k.$$

If $\psi(u)$ is differentiable at x_k, we say a direction s_k is a descent direction if $\nabla\psi(u_k)^T s_k < 0$. The stepsize α_k can be obtained by a simple line search procedure to approximate the minimizer of $\psi(u_k + \alpha s_k)$. Given any descent direction s_k, a line search for $\min_{\alpha \geq 0} \psi(u_k + \alpha s_k)$ can be performed relatively inexpensively.

The difficulty with solving problem (3.3) is the nondifferentiability of $\psi(u)$ arising when a component of $\|Bu\|_1$ is zero, or $\|Au - u_{\text{init}}\|_2^2 = \sigma^2$. At a nondifferentiable point u_k, a descent direction may lead to no progress or small progress since the stepsize from $\min_{\alpha \geq 0} \psi(u_k + \alpha s_k)$ may be zero or extremely small.

A projected gradient method, e.g.,[2], can certainly be adapted to (3.3). At a typical iteration, a projected gradient method handles nondifferentiability by following all the nondifferential hyperplanes exactly, if possible. At a vertex (mn nondifferentiable hyperplanes for (3.3)), all but one nondifferentiable hyperplanes are followed exactly based on Lagrangian multipliers. Since the size of an image enhancement problem is typically large, the possibility of a projected gradient type method visiting a combinatorial number of vertices seems formidable. Furthermore, we do not assume explicit form of the matrix A. Hence an alternative approach is needed.

Recently, interior point methods have become a popular alternative for overcoming difficulties due to linear constraints or nondifferentiability, e.g., [8, 1]. In addition to important theoretical advantages, an interior point method can typically solve a large problem in a small number of iterations. Affine scaling methods are particularly attractive interior point methods because of their simplicity. In particular, interior point methods for solving a linear l_1 problem of the form ($\beta \in I\!R^n$, $X \in I\!R^{m\times n}$, $y \in I\!R^m$),

$$\min_{\beta} \|X\beta - y\|_1,$$

have been considered by a few researchers, e.g., [9, 19, 3]. Amongst these, the algorithm [3] by Coleman and Li is particularly attractive for the image enhancement minimization problem (3.3) for the following reasons:

- Algorithm [3] works on an l_1 problem directly; hence adaptation to the enhancement problem (3.3) can be done in a natural fashion;
- Unlike other interior point methods, algorithm [3] seems less sensitive to a starting point; thus the blurred and noisy image u_{init} can be used as the starting point, each subsequent image will be incrementally enhanced;

- Algorithm [3] has appealing convergence properties: global and quadratical convergence;
- In computations, algorithm [3] exhibits the typical affine scaling method behavior: a small number of iterations are required to solve a large problem.

In the next section, we describe how we adapt the affine scaling algorithm in [3] to our image enhancement minimization problem (3.2) using the formulation (3.3).

4. An Affine Scaling Method. The central idea of Coleman-Li's method [3] is to employ affine scaling to generate a good descent direction so that nondifferentiability does not immediately prohibit decrease of the objective function $\psi(u)$. Subsequently, we adapt this approach to the image problem (3.3). We assume that the quadratic nondifferentiable function $\max(0, \|Au - u_{\text{init}}\|_2^2 - \sigma^2)$ is linearized at each iteration.

Let r_k denote the residual and g_k denote the sign of the residual, i.e.,

$$r_k \stackrel{\text{def}}{=} \begin{bmatrix} \|Au_k - u_{\text{init}}\|_2^2 - \sigma^2 \\ Bu \end{bmatrix}, \tag{4.1}$$

$$g_k \stackrel{\text{def}}{=} \begin{bmatrix} \mu \max(0, \text{sgn}(r_k(1))) \\ \text{sgn}(r_k(2\!:\!\text{length}(r_k))) \end{bmatrix}. \tag{4.2}$$

Hence nondifferentiability of $\psi(u)$ occurs when a component of the residual r is zero.

Let $J_k \stackrel{\text{def}}{=} \begin{bmatrix} 2A^T(Au_k - u_{\text{init}}), & B^T \end{bmatrix}$ denote the Jacobian. It is easy to see that, if $\psi(u)$ is differentiable at u_k, $J_k g_k$ equals the gradient $\nabla\psi(u_k)$. For simplicity, we refer to $J_k g_k$ as the gradient of $\psi(u)$ at u_k.

The difficulty with the negative gradient direction, $-J_k g_k$, is that it may be an extremely poor descent direction because it can immediately encounter a nondifferentiable hyperplane and becomes ascent. Thus it is possible that no progress or only extremely small progress can be made.

We observe that a collection of descent directions can be generated by considering solutions to

$$J_k D_k^{-2} J_k^T s = -J_k g_k, \tag{4.3}$$

where D_k is a positive diagonal matrix. We illustrate next that D_k can be defined to yield good descent directions.

It can be easily verified that the solution to (4.3) satisfies the following property:

$$J_k^T s = -D_k^2(g_k - \lambda_{k+1}), \tag{4.4}$$

where λ_{k+1} can be considered as an approximation to the Lagrangian multiplier vector with respect to (3.3).

Denote $d_k \stackrel{\text{def}}{=} J_k^T s_k$. Let the vector β_k denote the stepsizes to nondifferentiable hyperplanes $r = 0$, i.e.,

$$\beta_k \stackrel{\text{def}}{=} -r_k \cdot/ d_k$$

where $\cdot/$ denotes the componentwise division, as in Matlab [10]. If we let $D_k = \text{diag}(|r_k|)$ in (4.3), using (4.4), we have

$$\beta_k = e \cdot/ (g_k - \lambda_{k+1}),$$

where e is a vector of all ones and $*$ denotes the componentwise vector division as in Matlab [10]. Under reasonable assumptions (see [3] for details), the Lagrangian multipliers $\{\lambda_k\}$ are bounded. Hence $\{\beta_k > 0\}$

are bounded away from zero. In other words, sufficient progress can be made along a solution s_k of (4.3) before nondifferentiability can stop decreasing of $\psi(u)$. Geometrically, this means that solutions of (4.3) will provide descent directions which are sufficiently angled away from nondifferentiable hyperplanes to allow for large stepsizes.

Mathematically, s_k corresponds to a projected gradient under the affine transformation $\hat{r} = \text{diag}(|r_k|^{\frac{1}{2}})r$ (for more detailed motivation, see [3]). The importance of this transformation can be appreciated from the observation that, in the transformed space, the corresponding point $\hat{r}_k$ of r_k is relatively remote from all the nondifferentiable hyperplanes.

Let the parameter θ_k denote, asymptotically, a measurement of optimality, e.g.,

$$\theta_k \stackrel{\text{def}}{=} \frac{\max(\|\text{diag}(r_k)(g_k - \lambda_k)\|_\infty, \max(\max(-\lambda_k(1), 0), \max(\max(|\lambda_k| - |g_k|, 0))))}{0.9 + \max(\|\text{diag}(r_k)(g_k - \lambda_k)\|_\infty, \max(\max(-\lambda_k(1), 0), \max(\max(|\lambda_k| - |g_k|, 0))))}, \tag{4.5}$$

where 0 is a vector of all zeros and the operation max is as defined in Matlab [10]. By considering a Newton step for the complementarity condition of (3.3), a better choice for D_k is a combination of r_k and the Lagrangian multiplier λ_k:

$$D_k \stackrel{\text{def}}{=} \text{diag}(|r_k|^{\frac{1}{2}})\text{diag}(|\theta_k \max(e, |g_k|) + (1 - \theta_k)(e - \text{sgn}(g_k) \cdot\!* \lambda_k)|^{-\frac{1}{2}}). \tag{4.6}$$

Here $*$ denotes the componentwise vector multiplication as in Matlab [10]. Globally, the affine scaling matrix D_k in (4.6) is roughly $\text{diag}(|r_k|)$. Asymptotically, this affine scaling matrix D_k makes it possible to obtain quadratic convergence for an l_1 problem. We refer the reader to [3] for more motivation.

Our affine scaling algorithm for image enhancement minimization problem (3.3) works in a simple fashion. At each iteration, we solve a system of linear equations (4.3) with D_k defined by (4.6), update λ_k and perform a simple line search for minimizing $\psi(u)$ along s_k. Since (4.3) can be considered a weighted least squares problem, our approach can also be regarded as an iteratively reweighted least squares method.

Taking the special structure of (3.3) into account, it is advantageous to satisfy the single quadratic constraint $\|Au - u_{\text{init}}\|_2 \leq \sigma$ first. Computationally, this can be achieved in the process of solving

$$\min_{u \in \mathbb{R}^{mn}} \|Au - u_{\text{init}}\|_2^2,$$

or equivalently

$$A^T A u = A^T u_{\text{init}}. \tag{4.7}$$

We choose a conjugate gradient (CG) method for this task (more details can be found in the appendix). Incorporating this computation into the algorithm, our enhancement method is summerized in FIG.1.

The proposed enhancement procedure has an attractive feature: enhancement is achieved in an incremental fashion. **Step 0** corresponds to deblurring of the initial image u_{init} (referred to as Stage 1). Subsequently, the procedure iterates through **Step 1** to **Step 5** (referred to as Stage 2). The main computation/iteration of Stage 2 is solving (4.3) and performing an one dimensional line search on $\psi(u)$. At this second stage, the noise is decreased at each iteration.

Implementation details for **Step 0**, **Step 3** and **Step 4** (including issues of preconditioning) can be found in the appendix.

An Affine Scaling Method for Image Enhancement

Step 0 Starting from u_{init}, compute an image u_0 satisfying $\|Au_0 - u_{\text{init}}\| < \sigma$ by applying a CG method for solving

$$\min_{u \in \mathbb{R}^{mn}} \|Au - u_{\text{init}}\|_2^2;$$

Set $k := 0$; $\lambda_k := 0$;

Step 1 Compute

$$r_k = \begin{bmatrix} \|Au_k - u_{\text{init}}\|_2^2 - \sigma^2 \\ Bu_k \end{bmatrix},$$

$$g_k = \begin{bmatrix} \mu \max(0, \text{sgn}(r_k(1))) \\ \text{sgn}(r_k(2:\text{length}(r_k))) \end{bmatrix};$$

Step 2 Compute θ_k by (4.5); Set up affine scaling matrix D_k:

$$D_k := \text{diag}(|r_k|^{\frac{1}{2}})\text{diag}(|\theta_k \max(e, |g_k|) + (1 - \theta_k)(e - \text{sgn}(g_k).*\lambda_k)|^{-\frac{1}{2}}).$$

Step 3 Solve

$$J_k D_k^{-2} J_k^T s = -J_k g_k;$$

Compute

$$d_k := J_k^T s_k;$$
$$\lambda_{k+1} := g_k + D_k^{-2} d_k;$$

Step 4 Perform a line search to obtain α_k

$$\min_{\alpha > 0} \psi(u_k + \alpha s_k)$$

and let $u_{k+1} := u_k + \alpha_k s_k$;

Step 5 Check stopping: if $\frac{\|\alpha_k s_k\|_2}{\|u_k\|_2} \leq tol$, terminate; otherwise, $k := k + 1$ and go to **Step 1**;

FIG. 1. *An Affine Scaling Method For An Image Enhancement Problem*

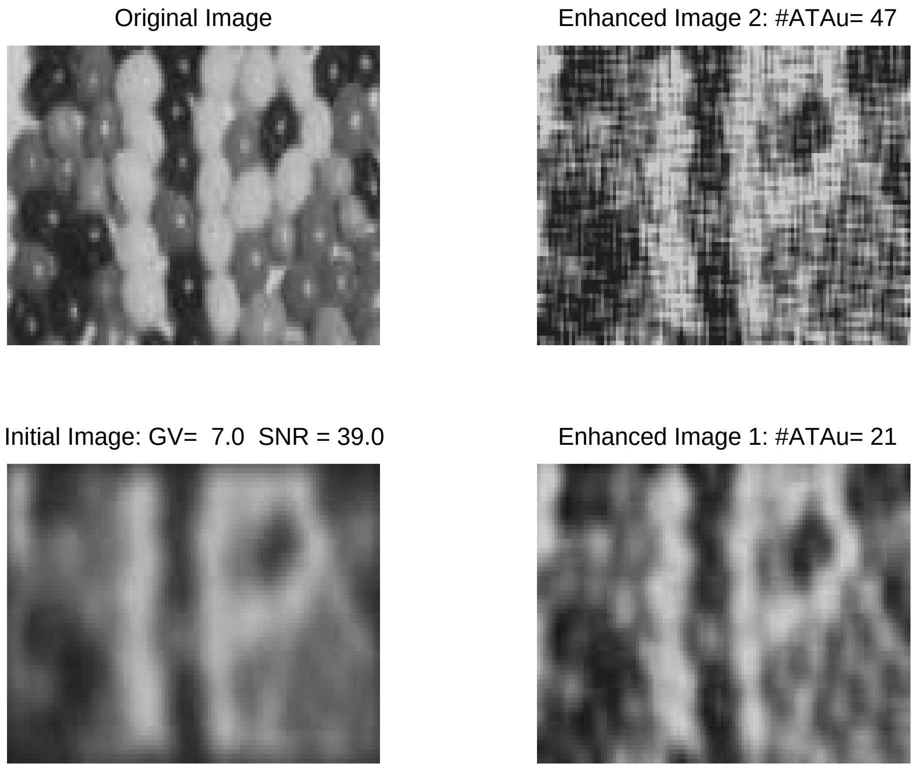


FIG. 2. *Enhanced Image 1 ($\sigma = \|\varepsilon\|_2$) and Enhanced Image 2 ($\sigma = 0.9\|\varepsilon\|_2$)*

5. Dynamic/Interactive Image Enhancement. Thus far we have assumed that u_{init}, Au and $\sigma = \|\varepsilon\|_2$ are available. In practice, however, σ is often estimated. The example in FIG. 2 indicates that an error in estimation of σ can significantly affect efficiency and quality of enhancement.

In FIG. 2, Enhanced Image 1 is computed with the correct size of the random noise $\sigma = \|\varepsilon\|_2$ while Enhanced Image 2 is obtained with estimation $\sigma = 0.9\|\varepsilon\|_2$. This example demonstrates that enhancement can be poor and computation can be more costly, if σ is set too small. Enhanced Image 1 is clearly preferable to Enhanced Image 2 both in quality and cost.

Since our enhancement procedure is incremental by nature, we propose to determine σ in a dynamic/interactive fashion. When a CG method is applied in **Step 0**, each successive new image appears less blurred initially and $\|Au - u_{\text{init}}\|_2$ has smaller value. At certain point of computation in **Step 0**, we observe that the image becomes increasingly noise corrupted. Therefore, we propose to examine each successive image and stop the computation in **Step 0** when an image u_0 seems to be sufficiently deblurred and continuation of computation introduces additional noise. Subsequently, we can set $\sigma = \rho\|Au_0 - u_{\text{init}}\|$ for subsequent steps with $\rho \geq 1$, depending on the amount of noise in the image u_0 and a user's desire for removing it.

6. Computational Experiments. We now report our computational experience with the proposed enhancement procedure in Matlab [10].

As described, our experiment assumes the following model

$$u_{\text{init}} = Au_{\text{true}} + \varepsilon, \tag{6.1}$$

where $\varepsilon \in I\!R^{mn}$ subsequently denotes a random Gauss noise.

Three types of blurring operations Au are considered in our experiment. Let U and $\hat{U}$ denote m-by-n matrices with $U = u(:)$ and $Au = \hat{U}(:)$. We consider:

1. pure noise: $Au = u$;
2. motion blurring: for a positive integer l (e.g., $l = 21$),

$$\hat{U}_{i,j} = \frac{1}{l}\sum_{k=-l}^{l} U_{i,j-k}, \quad i = 1\!:\!m,\ j = 1\!:\!n;$$

3. Gaussian blurring:

$$\hat{U} = \frac{1}{\sum_{i,j}|X_{ij}|}\mathbf{conv2}(U, X),$$

where

$$X_{ij} = \exp\left(-\frac{((i-l-1)^2 + (j-l-1)^2}{4\vartheta}\right),\ \ i = 1\!:\!(2l+1),\ j = 1\!:\!(2l+1)$$

and ϑ is a Gaussian variance (GV) parameter. The Matlab function **conv2**(U, X) returns a 2-dimensional matrix convolution.

For subsequent presentation, a signal to noise ratio (SNR) for an image problem is defined as:

$$\text{SNR} \stackrel{\text{def}}{=} \frac{\|Au_{\text{true}}\|_2}{\|\varepsilon\|_2}.$$

In FIG. 3 – FIG. 12 we demonstrate the quality of enhancement for various two dimensional images. In addition, the amount of computation required for each enhancement is reported. The main computation of our enhancement procedure is the matrix-vector product A^TAu and Bu. The computation of A^TAu is usually significantly more expensive than Bu except in the pure noise case. Hence, in addition to the number of iterations, we report the number of A^TAu matrix-vector multiplications required for each enhancement except in the pure noise case in which the number of Bu matrix-vector multiplications is reported.

FIG. 3 illustrates an enhancement of a 432-by-512 image corrupted with severe Gaussian blurring and mild random noise. Significant enhancement is achieved with 37 matrix-vector multiplications A^TAu.

FIG. 4 illustrates an enhancement of a 200-by-320 image corrupted with severe motion blurring and mild random noise. Significant enhancement is achieved with 24 matrix-vector multiplications A^TAu.

FIG. 5 shows an enhancement of a 256-by-256 image corrupted with severe Gaussian blurring and mild random noise. Significant enhancement is achieved with 21 matrix-vector multiplications A^TAu.

FIG. 8 indicates an enhancement of a 64-by-128 image in the case of severe pure noise and mild Gaussian blurring together with significant noise corruption. In the pure noise corruption case, no computation is necessary for **Step 0**.

Image enhancement examples with pure noise are shown in FIG. 7, 11 and 12.

A small number of iterations are typically required for an image enhancement problem using our procedure. We also observe that many nondifferentiable hyperplanes can be crossed at each iteration (i.e., many residual components r_i change their signs at each iteration).

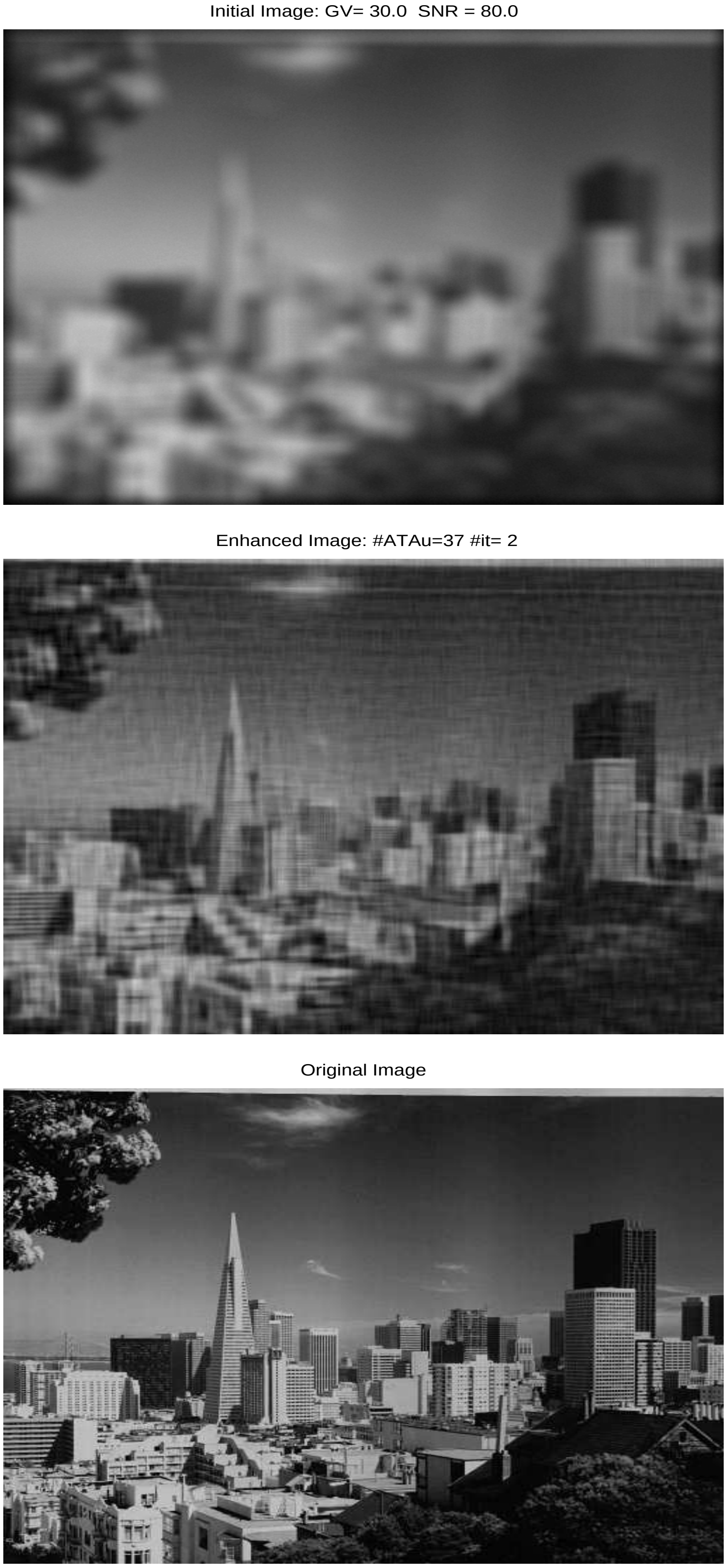


FIG. 3. *San Francisco (432-by-512 pixels)*

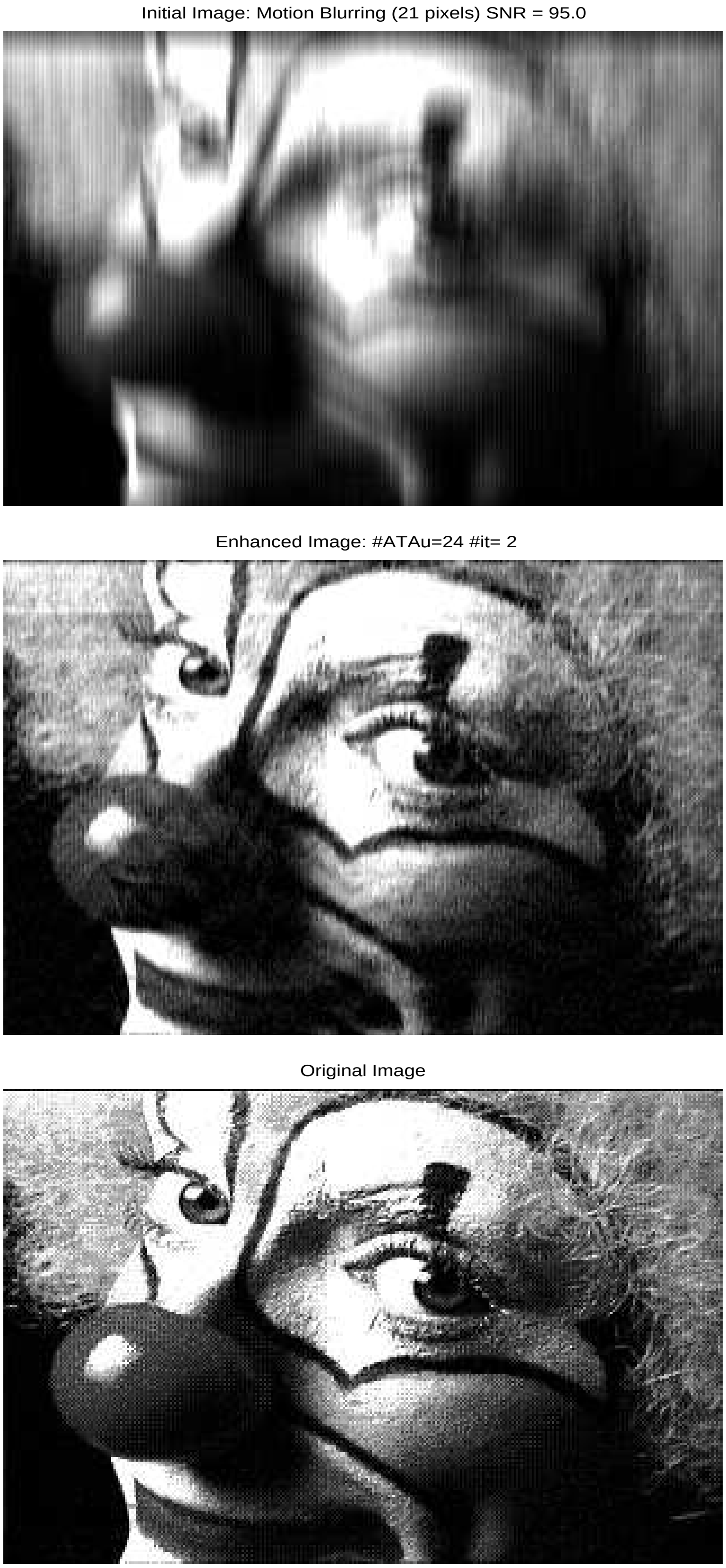


FIG. 4. *Clown (*200*-by-*320 *pixels)*

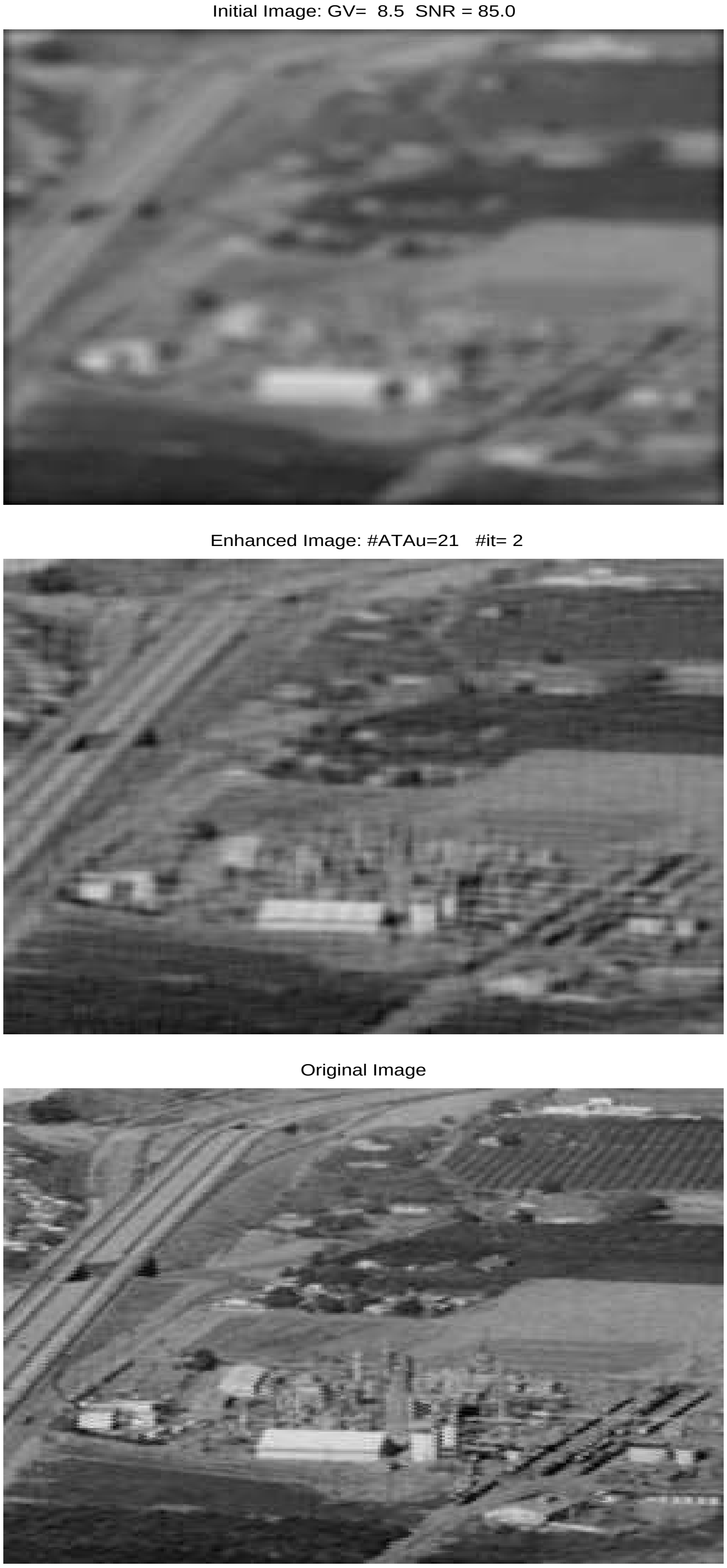


FIG. 5. *Chemical Plant (*256*-by-*256 *pixels)*

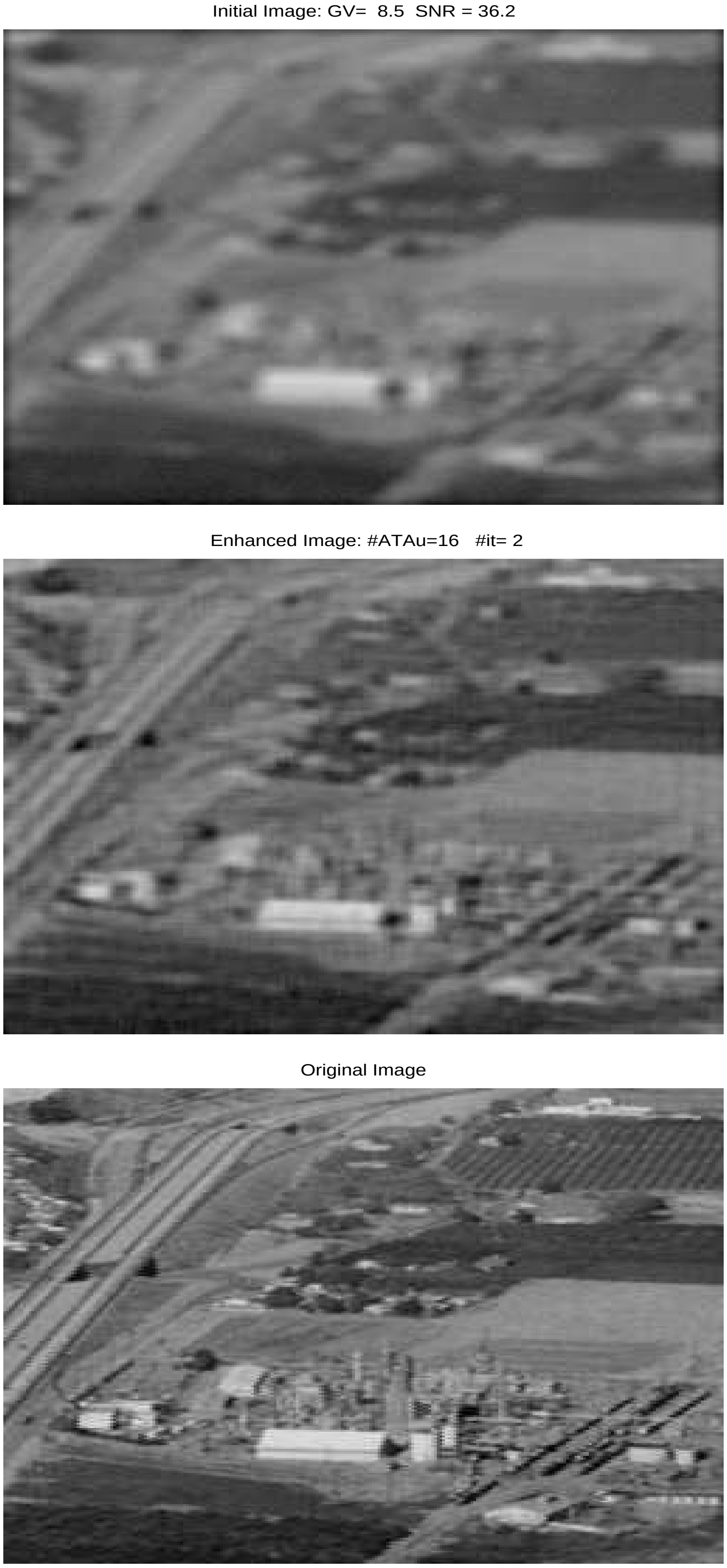


FIG. 6. *Chemical Plant (256-by-256 pixels)*

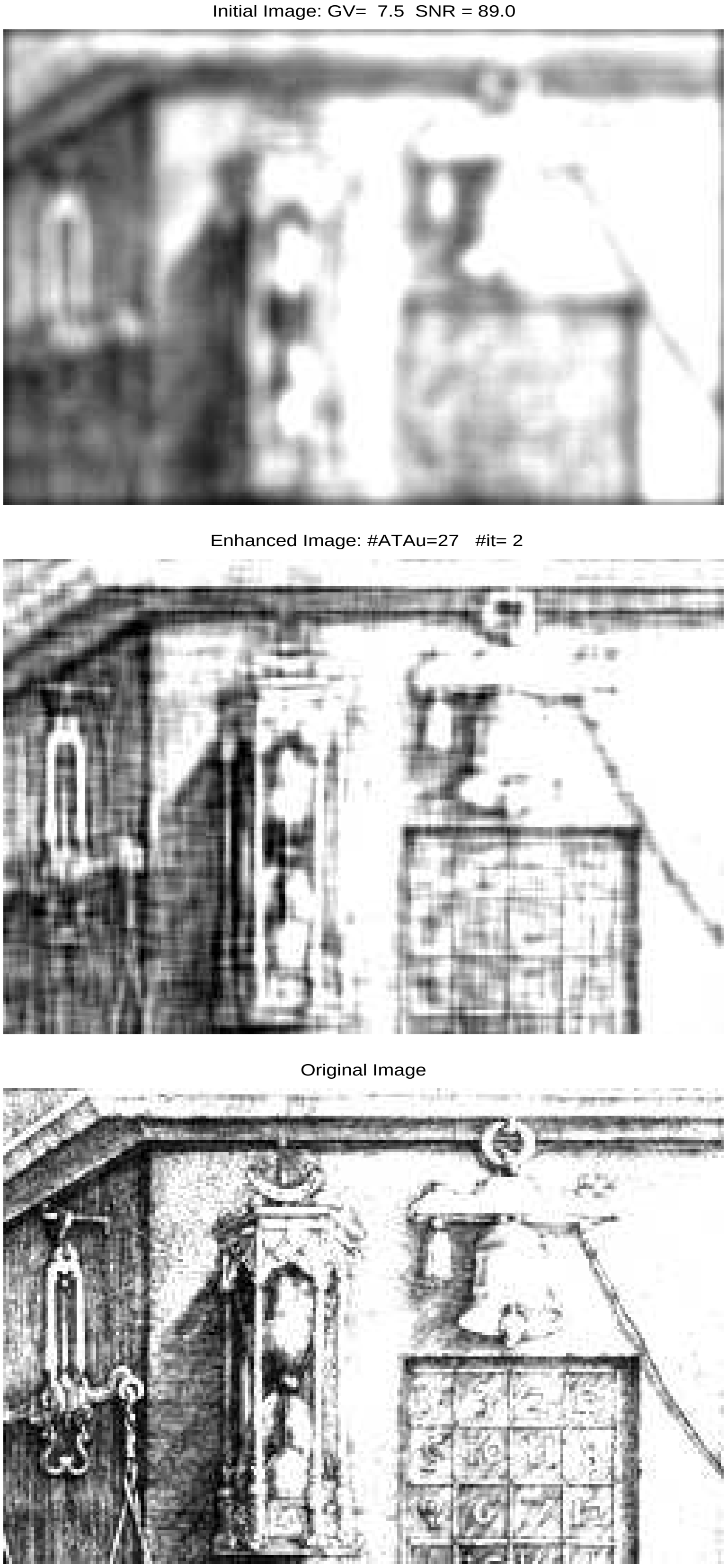


FIG. 7. *Durer Etiching (*156*-by-*256 *pixels)*

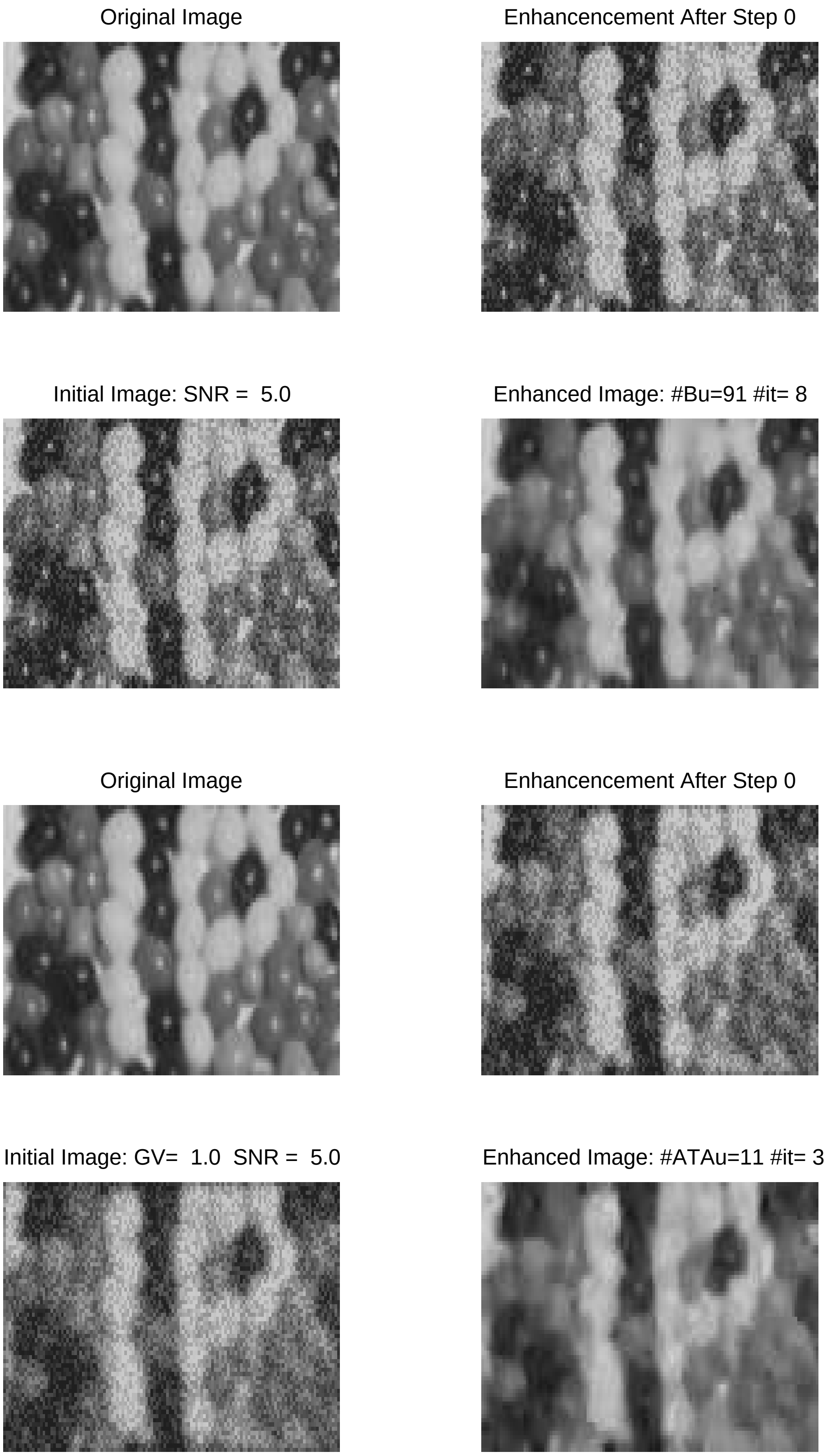


FIG. 8. *Jelley Beans (*64*-by-*128 *pixels)*

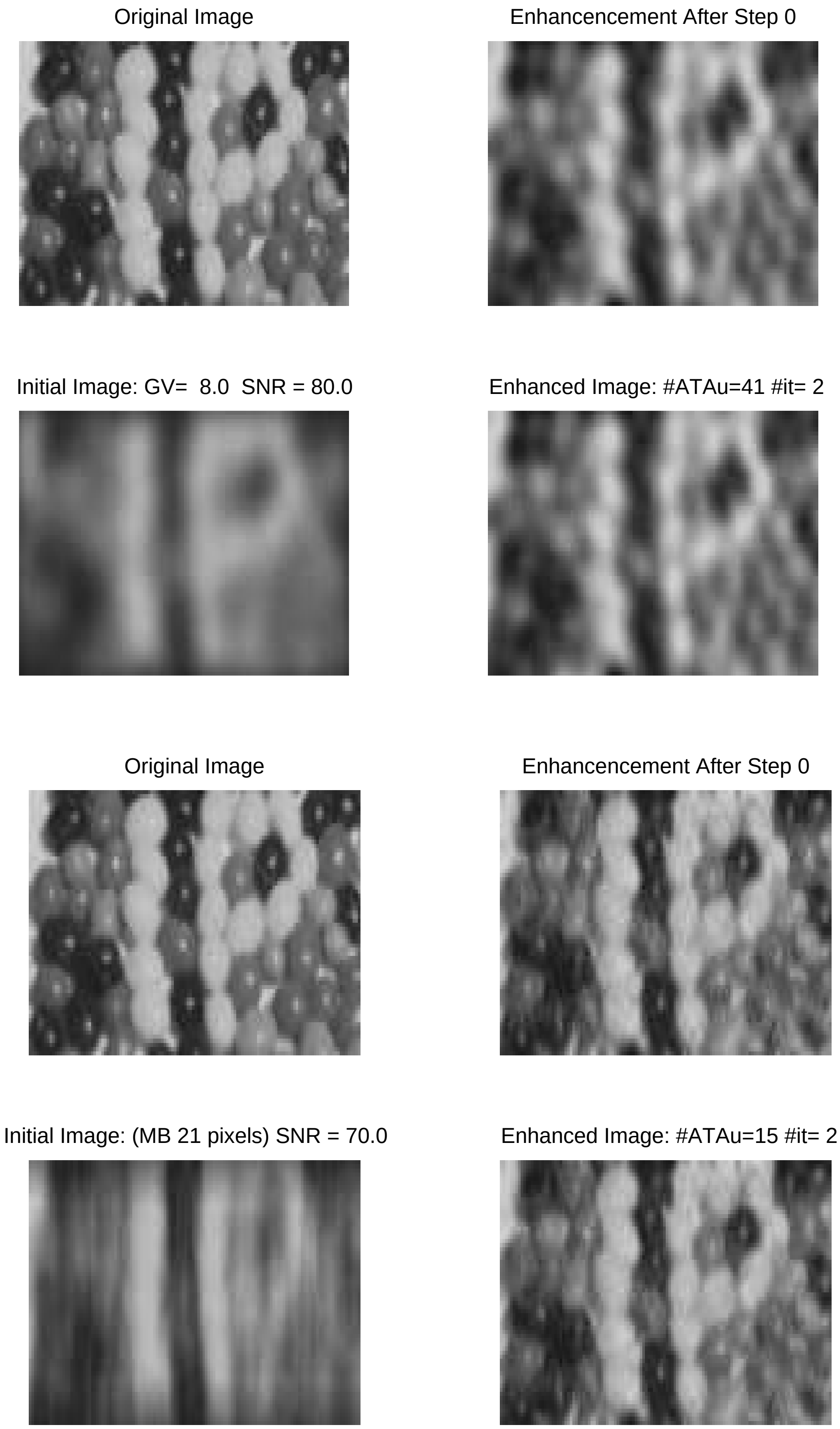


FIG. 9. *Jelley Beans (*64*-by-*128 *pixels)*

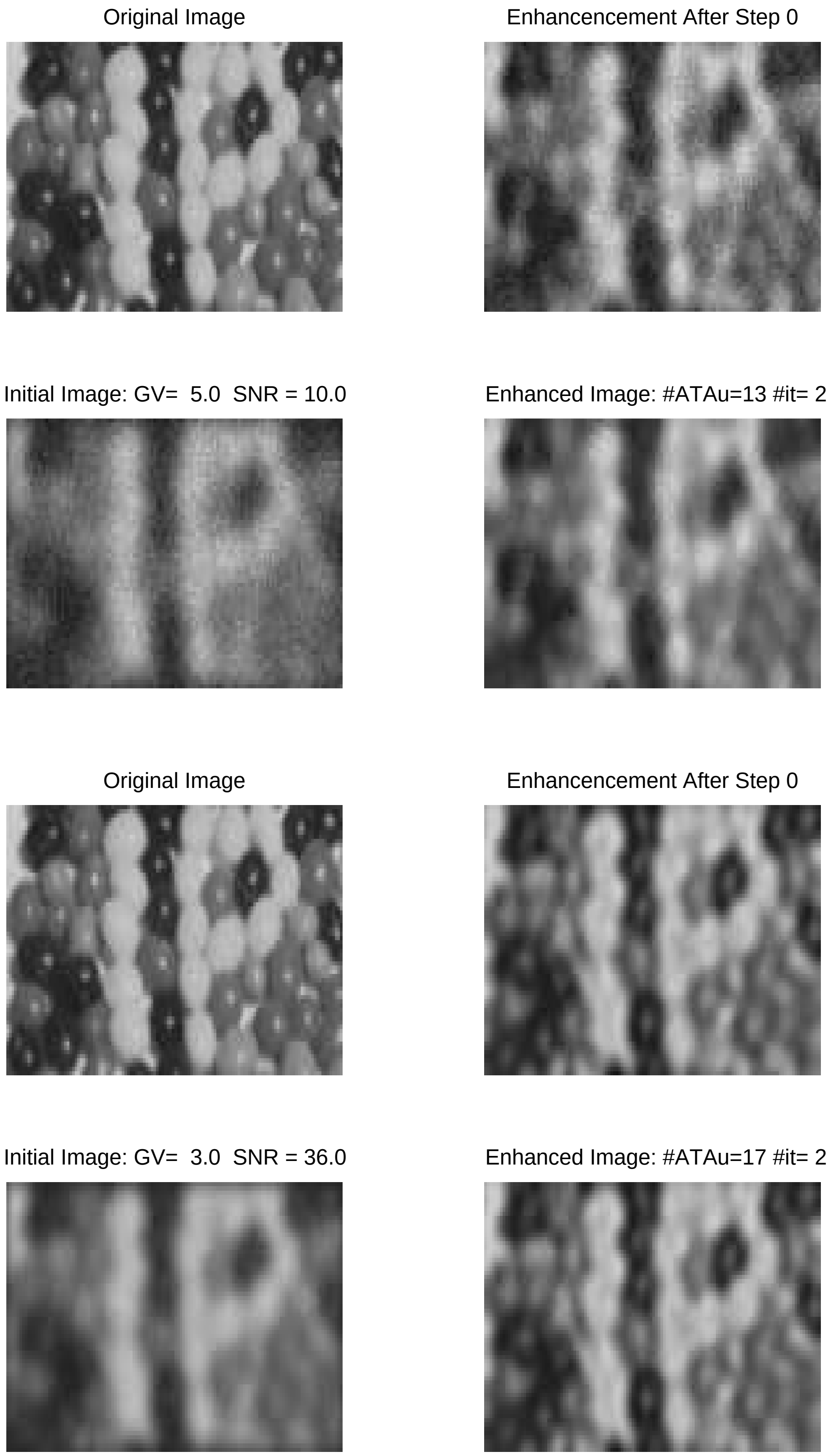
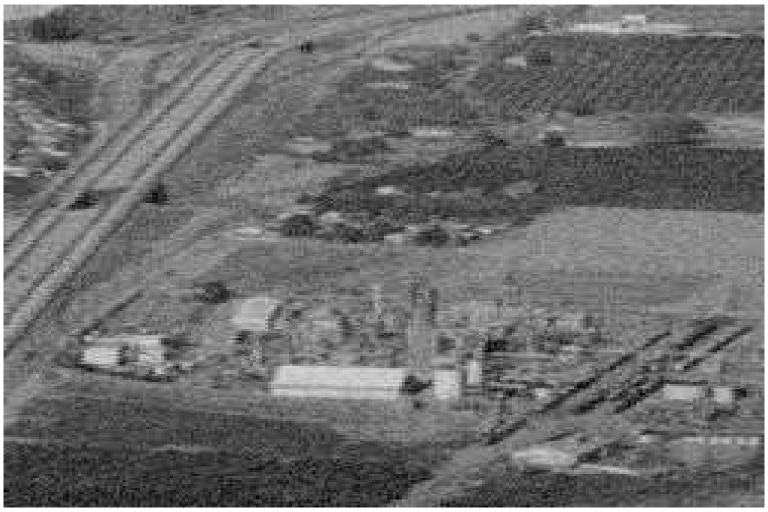


FIG. 10. *Jelley Beans (*64*-by-*128 *pixels)*

FIG. 11. *Lenna (*372*-by-*346 *pixels)*

Initial Image: SNR = 7.5



Enhanced Image: #Bu=75 #it= 7

Original Image

FIG. 12. *Chemical Plant* (256-*by*-256 *pixels)*

7. Concluding Remarks. We propose a computational algorithm for a constrained nondifferentiable minimization problem for image enhancement. Based on minimizing total variation proposed by Rudin et al in [13, 14, 15], we directly solve a discretized linear l_1 minimization problem with a quadratic inequality constraint to achieve image enhancement.

Our computational algorithm uses an affine scaling technique for nondifferentiability to achieve fast convergence. It is based on the globally and quadratically convergent method proposed by Coleman and Li [3] for a linear l_1 problem. A correction technique is also used for the quadratic constraint in the line search. Because each iteration in our algorithm decreases the cost function, each iteration incrementally improves the image. The incremental enhancement property of our method leads naturally to interactive procedure that can be used when the amount of noise is unknown or poorly estimated.

Computationally, the algorithm is simple: the main computation for each iteration are matrix-vector products corresponding to blurring and difference operations. It can also be regarded as an iterative reweighted least-squares approach. Our algorithm is also suitable for a conjugate gradient implementation. Diagonal preconditioners are used in the current implementation. Alternative preconditioners may lead to better efficiency.

Preliminary experience with our algorithm suggests that it is effective. Significant enhancement is typically achieved in a small number of iterations.

Additional constraints can be easily incorporated in our formulation and computational methods.

Finally, we would like to comment on the potential of our computational method for other applications. The total variation approach can be applied to a variety of inverse problems for which resolution loss or blurring is inherent. Examples of such problems include tomography and electrical impedance imaging, e.g., [4]. The present computational approach can be adapted naturally to such inverse problems.

Acknowledgements. The authors express their gratitude to Tom Coleman for helpful conversation on this work. We are also grateful to Gonzalo Arce for making some of the images used in this work available.

Appendix. Conjugate Gradient Implementation and Line Search

Conjugate Gradient Implementation (Step 0 and Step 3). First we consider **Step 0**. Our objective is to compute u_0 such that $\|Au - u_{\text{init}}\|_2 < \sigma$. Even though we do not want to solve the equation

$$A^T Au = A^T u_{\text{init}},$$

a conjugate gradient (CG) method for solving the above equation decreases $\|Au - u_{\text{init}}\|_2$ at each step. Therefore, we can compute u_0 by applying a CG method for $A^T Au = A^T u_{\text{init}}$ until $\|Au - u_{\text{init}}\|_2 \le \sigma$.

Recall our assumption that computation of Au, instead of A, is specified. Fortunately, it is usually straightforward to figure out computation of $A^T u$, given a description for computing Au. For example, let us assume that

$$Au = \textbf{conv2}(X, u),$$

where $X \in I\!R^{l\times l}$ and **conv2** is a convolution operation in Matlab [10]. Then it can be easily verified that

$$A^T u = Au.$$

Now we consider **Step 3**. Since the coefficient matrix of (4.3) is at least positive semidefinite, we again use a CG method for solving a system of linear equations (4.3) to compute a descent direction. We point out, however, that it is possible to form the coefficient matrix of (4.3) explicitly. Moreover, this coefficient matrix has a special structure: a tridiagonal matrix plus a rank one update. Hence direct methods (e.g.,Cholesky) can be used for (4.3) as well.

It is well known that performance of a CG method typically depends heavily on a preconditioner. For computation in **Step 0**, we do not have an explicit representation for $A^T A$. Hence it is difficult to obtain a preconditioner. However, it is usually possible and inexpensive to compute the diagonal of $A^T A$. Consider again the case that

$$Au = \textbf{conv2}(X, u).$$

Then it can be easily verified that

$$\text{diag}(A^T A) = \textbf{conv2}(X \cdot\!* X, \text{ones}(m, n)).$$

In our implementation, we use diagonal preconditioners for both **Step 0** and **Step 3**.

Line Search. After a descent direction s_k is computed in **Step 3**, we perform a line search procedure to decrease $\psi(u)$ along s_k as much as possible. If there were no quadratic penalty term for $\|Au - u_{\text{init}}\|_2 \le \sigma$, this can be done with low cost in a straightforward fashion: update the ratio of decrease after crossing each nondifferentiable hyperlane until the direction ceases to be descent.

Unfortunately, the penalty term from $\|Au - u_{\text{init}}\|_2 \le \sigma$ excludes this simple line search procedure. Nonetheless, we can perform a correction step based on the described stepsize calculation for an l_1 problem. Correction steps are often used for a nonlinearly constrained problem, e.g., [6].

function [new] = Line($u, u_{\text{init}}, s, d, r, \theta, g,$ idx);
ratio := $g^T * d$;
j := 0; α := 0; α_1 := 0;
while ratio < 0 **do**
    ratio := ratio $- 2 * g(\text{idx}(j)) * d(\text{idx}(j))$;
    $\alpha_1 := \alpha;\ \alpha := \beta(\text{idx}(j))$;
end
$\alpha := \alpha_1 + \max(0.9, 1-\theta) * (\alpha - \alpha_1)$;
$\alpha_q :=$ **StepsizeQ**(u, s);
if $\alpha <= \alpha_q$ & $r(1) > 0$
    new := $u + \alpha * s$;
    return
end
$d_n := A^T * (A * (u + \alpha * s) - u_{\text{init}})$;
new := $u + \alpha * s +$ **StepsizeQ**$(u, d_n) * d_n$;
while $\psi(\text{new}) > \psi(u)$ **do**
    $\alpha := 0.1 * \alpha$;
    $d_n := A^T * (A * (u + \alpha * s) - u_{\text{init}})$;
    new := $u + \alpha * s +$ **StepsizeQ**$(u, d_n) * d_n$;
end

FIG. 13. *A Line Search Procedure*

Since it is relatively easy to compute a stepsize along any direction to the quadratic constraint $\|Au - u_{\text{init}}\|_2^2 = \sigma^2$, correction steps can be performed along the normals of $\|Au - u_{\text{init}}\|_2^2 = \sigma^2$.

Assume that $\beta(\text{idx})$ lists the positive stepsizes to the hyperplanes $Bu = 0$ in ascending order and the function [new] = **StepsizeQ**(u, d) returns the stepsize along the normal $A^T(Au - u_{\text{init}})$ from u to the quadratic curve $\|Au - u_{\text{init}}\|_2^2 = \sigma^2$. The pseudo-Matlab program in FIG. 13 provides a detailed description of our line search procedure. We note that efficiency of this line search can be further improved with additional heuristic techniques, e.g., a binary search.

REFERENCES

[1] E. BARNES, *A variation on karmarkar's algorithm for solving linear programming problems*, Mathematical Programming, 36 (1986), pp. 174–182.

[2] R. H. BARTELS, A. R. CONN, AND J. W. SINCLAIR, *Minimization techniques for piecewise differentiable functions: the l_1 solution to an overdetermined linear system*, Siam J. Numer. Anal., 15 (1978), pp. 224–240.

[3] T. F. COLEMAN AND Y. LI, *A globally and quadratically convergent affine scaling method for linear l_1 problems*, Mathematical Programming, 56 (1992), pp. 189–222.

[4] D. DOBSON AND F. SANTOSA, *An image enhancement technique for electrical impedance tomography*, Inverse Problems, 10 (1994), pp. 317–334.

[5] D. DOBSON AND F. SANTOSA, *Recovery of blocky images from noisy and blurred data*, Tech. Rep. 94-7, submitted to SIAM J. Appl. Math.

[6] R. FLETCHER, *Practical Methods of Optimization: Volume 2, Constrained Optimization*, John Wiley and Sons, 1981.

[7] K. ITO AND K. KUNISCH, *An active set strategy for image restoration based on the augmented lagrangian formulation*, tech. rep.

[8] N. KARMARKAR, *A new polynomial-time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[9] M. MEKETON, *Least absolute value regression*, tech. rep., AT&T Bell Laboratories, Holmdel, 1988.

[10] C. B. MOLER, J. LITTLE, S. BANGERT, AND S. KLEIMAN, *ProMatlab User's guide*, MathWorks, Sherborn, MA, 1987.

[11] D. PHILLIPS, *A technique for the numerical solution of certain integral equations of the first kind*, J. ACM, 10 (1962), pp. 84–97.

[12] J. ROSEN, *The gradient projection method for nonlinear programming, part ii, nonlinear constraints*, J. Soc. Ind. Appl. Math., 9 (1961), pp. 514–532.

[13] S. OSHER AND L. RUDIN, *Feature-oriented image enhancement using shock filters*, SIAM J. Numer. Anal., 27 (1990), pp. 919–940.

[14] L. RUDIN, S. OSHER, AND C. FU, *Total variation based restoration of noisy blurred images*, SIAM J. Num. Anal., p. to appear.

[15] L. RUDIN, S.OSHER, AND E. FATEMI, *Nonlinear total variation based noise removal algorithms*, Physica D., 60 (1992), pp. 259–268.

[16] S. TWOMEY, *On the numerical solution of fredholm integral equations of the first kind by the inversion of the linear system produced quadrature*, J. ACM, 10 (1963), pp. 97–107.

[17] ———, *On the numerical solution of fredholm integral equations of the first kind by the inversion of the linear system produced quadrature*, J. Franklin Inst., 297 (1965), pp. 95–109.

[18] C. VOGEL AND M. OMAN, *Iterative methods for total variation denoising*, preprint 1994.

[19] Y. ZHANG, *A primal-dual interior point approach for computing the l_1 and l_∞ solutions of overdetermined linear systems*, Tech. Rep. Technical Report, Department of Mathematics and Statistics, Univerisity of Maryland, 1991.