9-1-1995

A MULTISTAGE APPROACH TO THE HOPFIELD MODEL FOR BI-LEVEL IMAGE RESTORATION

Mona S. Badie
Purdue University School of Electrical and Computer Engineering

Okan K. Ersoy
Purdue University School of Electrical and Computer Engineering

A Multistage Approach to the Hopfield Model for Bi-Level Image Restoration

Mona S. Badie
Okan K. Ersoy

TR-ECE 95-22
September 1995

School of Electrical
and Computer Engineering
Purdue University
West Lafayette, Indiana 47907-1285

A MULTISTAGE APPROACH TO THE HOPFIELD MODEL FOR BI-LEVEL IMAGE RESTORATION

Mona S. Badie and Okan K. Ersoy

School of Electrical and Computer Engineering
1285 Electrical Engineering Building
Purdue University
West Lafayette, IN 47907-1285

TABLE OF CONTENTS

ABSTRACT

Badie, Mona. MSEE Purdue University, May 1995. A Multistage Approach to the Hopfield Model for Bi-level Image Restoration.
Major Professor: O.K. Ersoy

It was shown in previous published work that the neural networks Hopfield model can be an efficient tool in grey level image restoration by regarding the problem as a minimization of a two part cost measure, in which one component measures roughness, and the other measures distance from the original image. In this thesis, a multistage approach to the Hopfield network to restore bi-level images degraded by noise is considered where the problem of error minimization is addressed locally within each partition while the other partition is frozen. The natural choice for partitioning was into two stages, minimizing the odd data first followed by the even. A natural extension followed, which is splitting into four stages. Simulations were carried for different levels of noise, and different values of the regularization constant and the regularization matrices. The Multistage technique, in general, was proven successful in pushing the error function to a deeper minima than the one reached by the classical single stage Hopfield model.

CHAPTER I

INTRODUCTION

The issue of image restoration is a problem of combinatorial optimization. A combinatorial optimization problem is either a minimization or a maximization problem specified by the pair (S,c), where S represents the solution space, a finite but exponentially large set of possible solutions, and c is the cost function. For a minimization problem, we seek $x_{opt} \in S$ such that

$$c(x_{opt}) \leq c(x) \qquad x \in S$$

The solution x_{opt} is called the globally optimal solution or optimum and $c(x_{opt})$ is the optimal cost.

It has been shown that several theoretical and practical combinatorial optimization problems belong to the class of NP-complete (Non-deterministic Polynomial Time) problems. This suggests that optimal solutions cannot be attained in reasonable amounts of computation time. Further, this necessitates consideration of the trade-off between optimality and rapidly obtainable solutions. A class of algorithms called optimization algorithms seeks the former goal while the class of approximation algorithms pursue the latter.

A combinatorial optimization with constraints is known as a constrained combinatorial problem, while one without any constraints is an unconstrained combinatorial optimization problem. In some cases, a constrained optimization problems can be approximated by an unconstrained optimization problem by the penalty method.[1]

I.1 CURRENT OPTIMIZATION STRATEGIES

Traditional approaches to seeking minima include the steepest descent, conjugate descent, restart and simulated annealing algorithms. A brief discussion of each case is presented below.

I.1.1 Descent Algorithms

Presuming that the objective function possesses a bounded rate of change, steepest descent is successful for most problems but displays slow convergence properties. For a gradient function which is linear with respect to the variables, as in,

$$g(x) = \nabla E(x) = Ax - b \quad (2)$$

the rate of convergence, which is controlled by

$$\frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}} \quad (3)$$

where Ą and λ_{min} are, respectively, the largest and smallest eigen values of A, can approach unity. Further, the local minimum may not even be accurately attained owing to round-off errors. By incorporating the descent information from a previous iteration, the conjugate gradient descent algorithm overcomes the pitfalls of the steepest descent scheme and, for a quadratic objective function attains a local minimum in at most n steps, where n is the number of parameters in the objective function. Thus, the iteration is expressed as $x_{k+1} = x_k + \alpha_k d_k$

where, for steepest descent,

$$d_k = -g(x_k) = -g_k \quad (4)$$

and for conjugate gradient descent,

$$d_k = -g_k + \beta_k d_{k-1} \quad (5)$$

and α_k is chosen to minimize the objective function, $F(x_k + \alpha_k d_k)$ as a function of α_k.

Conjugate gradient descent owes its name to the fact that d_k is A-orthogonal to all previous descent directions, d_i, $i=0,1,..,k-1$. This ensures quadratic convergence in n steps.

For the more general non-quadratic nonlinear objective function, the above strategies regard the behavior as quadratic-dominant near each point. The factor β_k is chosen appropriately to exploit the orthogonality property stated above. One such choice

$$\beta_k = \frac{\| g_k \|^2}{\| g_{k-1} \|^2} \quad (6)$$

using Euclidean vector norms is improved upon using,

$$\beta_k = \frac{g_k^T [g_k - g_{k-1}]}{\| g_{k-1} \|^2} \quad (7)$$

The latter expression forces the search directions close to the direction of steepest descent while the former selection can stagnate when g_k and d_k approach orthogonality.

I. 1.2 Restart Algorithms

Although stability and convergence have been proven for clescent strategies, tracking down deeper local minima is largely governed by being in their vicinity. Restart procedures have been formulated to improve both the accuracy and efficiency of the descent methods. The use of an augmented descent direction , say for iterations $k>t$,

$$d_k = -g_k + \beta_k d_{k-1} + \gamma_k d_t \quad (8)$$

such that d_r, $r=t, t+1, t+2, \ldots$ are mutually conjugate and d_t is the restart direction. The selection of β_k is made to force d_k to be orthogonal to all past directions and that of γ_k to ensure the orthogonality of $\gamma_k d_k$ to d_k.

Restart procedures perform more effectively when the second derivative information is considered. The restart based on steepest descent can lead to an increase in the objective function since it does not account for the previous descent directions. An essential requirement for the success of restart procedures is that the new direction, d_k for $k>t$ satisfy, $d_k^T g_k<0$.

It has been established,that the conjugate gradient descent algorithm applied after restart will proceed to termination whence either the above condition is met or $g_k=0$. However, unless the transformation is appropriately chosen, the convergence could be to a point other than the true minimum. This could occur since restarts made using projected gradients, $\hat{g}_k$ do not force

$$\frac{g_k^T g_{k+1}}{\|g_k\|^2} \quad (9)$$

to zero but rather a similar expression in $\hat{g}_k$ which, in turn, takes g_k to a non-zero limiting value. A suitable criterion for restart is provided by observing the behavior of (9).. A nonzero trend implies that gradient orthogonality is failing and suggests the initiation of restart.

There also exist algorithms based on coordinate descent schemes wherein the optimization process is conducted on subsets of parameters at a time, while holding the others constant. Such technique do not require gradient information but display slower convergence features. Further, objective functions which depend on a very large number of uncoupled parameters are better suited for cyclic coordinate descent searches. Cyclic coordinate descent is useful where the line search along a coordinate direction can be performed efficiently acd effectively.

I.1.3 Simulated Annealing Algorithms

Multivariate optimization involving the minimization of a function depending on many parameters finds close ties to a thermodynamical system exhibiting several degrees of freedom which exist in thermal equilibrium at a specific finite temperature. The characteristics of large scale optimization of complex systems are likened to the temperature-sensiitive behavior of materials. The annealing process in solids is an effort to attain the lowest energy configuration of its atomic states through macroscopic temperature variations. However, such a procedure requires careful control of these fluctuations in order to avoid creating structural defects in crystallinity. This notion has been found applicable to optimization problems. The simulated annealing process is a stochastic search algorithm with the following requirements:

(i) A well-defined notion of a system configuration

(ii) A stochastic rearrangement of the elements within a configuration

(iii) A quantitative objective measure of the trade-offs.

(iv) A detailed "annealing" schedule identifying the temperature variation and duration of the system evolution at a specific temperature.

A mechanism for overcoming the entrapment of a system at local minima should be there. This allows for increases in the energy level as a means to reconfigure the system into a possibly lower energy state (the annealing action). The convergence to the lowest energy state is likely to be very slow since the search procedure must exhaust the likelihood of deeper minima. Through probabilistic state changes, the temperature deviation is reduced or increased based on the trend displayed by the energy function. The use of an iterative simulated annealing procedure to avoid a local "freeze" employs the study of the energy function at each of a sequence of diminishing temperature settings.

1.2 NEURAL NETWORKS AS AN OPTIMIZATION STRATEGY

Artificial Neural Networks (ANN), with their parallel computing capabilities, are an attractive tool for optimization problems.

[2],[3],[4],[5],[6],[7],[8],[9],[10]

The Hopfield network has been used to solve a diverse range of optimization tasks, e.g, the travelling salesman problem (TSP) [11] and the four color problem [12].

There have been particular interest in the hardware implennentation of the neural networks. It was shown that highly interconnected networks of simple analog processors can collectively compute good solutions to difficult optimization problems [24]. The dynamics of such networks, generated by the analog response, high interconnectivity, and the existence of feedback connections, produce a path through the space of independent variables that tends to minimize the objective function value. Eventually, a stable steady-state configuration is reached, which corresponds to a local minimum of the objective function.[13] For example, a network was designed to provide solutions to the travelling salesman problem and the network could provide good solutions during an elapsed time of only a few characteristic time constants of the circuit. Other less complicated problems which are not of NP-complete class, such as linear programming problems can be solved by networks of analog processors.

In some cases, more than one optimization technique is combined. Although simulated annealing is a powerful method for optimization, it is essentially sequential. This means that for problems with a large data set, the computation time required is quite high. With the emergence of neural networks and their offering of parallel computation, simulated annealing and neural networks were united to exploit the power of both these tools. Boltzmann machines are Hopfield networks with a probabilistic transition of states of neurons instead of deterministic transitions. The probabilistic transition is based on simulated annealing. This effective union has been utilized successfully in a number of optimization problems [2].

The linear Hopfield network is primarily applied to the solution of combinatorially complex decision problems. Jeffrey and Rossner [14] extended the Hopfield technique to the nonlinear unconstrained optirnization problem. Kennedy and Chua [15] presented an analog implementation of a network solving a nonlinear optimization problem.

In this thesis, we rephrase the engineering issue of image restoration as a problem of combinatorial optimization.

The energy function relates to the original problem as the mean square reconstruction error function whose minimization is being sought for the

purpose of signal recovery from noisy received image.

CHAPTER II

IMAGE SAMPLING, QUANTIZATION AND HALFTONING

The most basic requirement for computer processing of images is that the images be available in digital form, that is, as arrays of finite length binary words. For digitization, the given image is sampled on a discrete grid and each sample or pixel is quantized using a finite number of bits. The digitized image can then be processed by the computer. To display a digital image, it is first converted to an analog signal, which is scanned onto a display.

II.1 IMAGE SCANNING

A common method of image sampling is to scan the image row by row and sample each row. An example is the television camera with vidicon camera tube or an image dissector tube. An object, film, or transparency is continuously illuminated to form an electron image on a photosensitive plate called the target. A finite-aperture electron beam scans the target and generates current which is proportional to the light intensity falling on the target.[16]

II. 2 IMAGE SAMPLING

The digitization process for images can be understood by modeling them as bandlimited signals. Although real-world images are rarely bandlimited, they can be approximated by bandlimited functions.

A function $f(x,y)$ is called bandlimited if its Fourier transform $F(\varepsilon_1, \varepsilon_2)$ is zero outside a bounded region in the frequency plane, for instance,

$$F(\varepsilon_1, \varepsilon_2) = 0 \qquad |\varepsilon_1| > \varepsilon_{xo}, \quad |\varepsilon_2| > \varepsilon_{yo} \quad (10)$$

The quantities ε_{xo} and ε_{yo} are called the x and y bandwidth of the image.

The Fourier transform of an arbitrary sampled function is a scaled, periodic replication of the Fourier transform of the original function.

From the uniqueness of the Fourier transform, we know that if thee spectrum of the original image could be recovered somehow from the spectrum of the sampled image, then we would have the interpolated continuous image from the sampled image. If the x, y sampling frequencies are greater than twice the bandwidth,

$$\varepsilon_{xs} > 2\,\varepsilon_{xo} \quad , \quad \varepsilon_{ys} > 2\,\varepsilon_{yo} \qquad (11)$$

then $F(\varepsilon_1 , \varepsilon_2)$ can be recovered by a low pass filter. The lower bounds on the sampling rates, that is,

$$2\,\varepsilon_{xo} \quad , \quad 2\,\varepsilon_{yo} \qquad (12)$$

are called the Nyquist rates or the Nyquist frequencies. The sampling theory states that a bandlimited image sampled above its x and y Nyquist rates can be recovered without error by low-pass filtering the sampled image. However, if the sampling frequencies are below the Nyquist frequencies,

$$\varepsilon_{xs} < 2\,\varepsilon_{x0} \qquad \varepsilon_{ys} < 2\,\varepsilon_{y0} \quad (13)$$

then the periodic replications of $F(\varepsilon_1, \varepsilon_2)$ will overlap, resulting in distorted spectrum $F_s(\varepsilon_1, \varepsilon_2)$ from which $F(\varepsilon_1, \varepsilon_2)$ is irrevocably lost. The frequencies above half the sampling frequencies are called the foldover frequencies. This overlapping of successive periods of the spectrum causes the foldover frequencies in the original image to appear as frequencies $\varepsilon_{xs}/2, \varepsilon_{ys}/2$ in the sampled image. This phenomenon is called aliasing.

Aliasing errors cannot be removed by subsequent filtering. Aliasing can be avoided by low-pass filtering the image first so that its bandwidth is less than on-half of the sampling frequency.

II.3 IMAGE QUANTIZATION

The step subsequent to sampling in image digitization is quantization. A quantizer maps a continuous variable u into a discrete variable u', which takes values from a finite set $\{r_1, \dots r_L\}$ of numbers. This mapping is generally a staircase function and the quantization rule is as follows: Define $\{t_k, k=1,\dots,L+1\}$ as a set of increasing transition or decision levels with t_1 and t_{L+1} as the minimum and maximum values, respectively, of u. If u lies in interval $[t_k, t_{k+1})$, then it is mapped to r_k the kth reconstruction level.

II.4. IMAGE SEGMENTATION AND BINALIZATION

Storing gray level images requires a lot of memory because of the image depth. An image scanned in 256 grey levels requires 10 bits per pixel. Simplifying an image while retaining information regarding shapes and geometric structures is necessary in many image analysis and pattern recognition problems. Reducing the number of grey levels (possibly to two as in the case of bi-level images) could be an advantageous tradeoff.

Recently, image processing devices such as facsimiles, copiers, and scanners are used everywhere and the display devices which are able to reproduce halftone images in addition to text and figure images in high quality is required.

Generally, document images contain the text, the figure, and the photographic area. Because of their different characteristics, adaptive processing of such areas are needed when the document images are displayed, stored, edited and transmitted.[17][18][19].

At first, a document image which is scanned in 256 grey levels is divided into

M x M pixel block. The first is the ratio of the number of pixels which have greater density than a predetermined threshold value within a block. The second is the distribution of the maximum value in the differential ones to the density of 8-neighbor pixels. The first feature is called the black density and the second one the density difference hereafter. Generally, continuous-tone areas have high black density and low density difference. On the other hand, text areas have low black density and high density difference. Then, an attribute of a block, which is photo, text, or vacant, is decided by using the above two features, where label "photo" means continous -tone or screened halftone block, " text" states text or figure and " vacant " represents that the block is background. Finally, the attribute is corrected in comparison to the attributes of neighbor blocks.

It is sometimes difficult to decide the attributes of blocks precisely from the local information within a block. For example, few blocks in a text area may have the same features as photo and may be mislabeled as photo blocks and vice versa. So the correction of attributes is necessary to get precise segmentation of documents. The correction process is conducted by comparing an attribute of a block to its neighbors. More concretely, an attribute of a block which was incorrectly decided as text might be changed to photo if the attributes of the nearest neighbor blocks were photo and so on.

An appropriate binalization method is then used. For the photographic areas, the systematic dither is suggested (but will not be discussed in this thesis, being out of the scope of our work).

For text areas, several bi-level quantization methods could be used.[20] A method based on the discriminant and least squares criteria(DLSCM) is one of them.

II.4.1 DLSCM Method

If two categories such as characters and background exist in a block, DLSCM gives the threshold level which minimizes the variance in each category and maximizes the variance between two categories. But all blocks do not always include two categories because each block is an
M x M pixels sub-part of the original image. Then the following two limitations could be applied in the implementation of DLSCM to the text images.

(1) If the threshold value calculated by DLSCM is very low because of too few number of pixels belonging to the characters in a block, the lower limit, TH1, is set in the threshold value.

(2) If a threshold value calculated by DLSCM is less than TH1, the threshold value of a neighbor block which has already been processed is used in order to maintain the continuity of the threshold value.

Furthermore, DLSCM gives an appropriate threshold value for bi-level quantizing, when the number of pixels belonging to each category is approximately same. But the threshold value is generally shifted to the background level, because text images have very few number of pixels belonging to the characters compared to the existing number of background pixels.

II.4.2 Improved DLSCM

It uras found that the DLSCM does not always give the appropriate threshold value for binalizing the text images. An improvement of the DLSCM is suggested in [21]. If the number of all pixels in a block is A, the number of pixels which have lower density than the value calculated by DLSCM is B, the given maximum offset value is Z, then the variable offset V is represented as $Z*B/A$, where Z is constant. The value calculated by the DLSCM plus the offset value V was used as a slice level for binaliization of text images. This method does not require extra calculation because the value B was already calculated in the process of DLSCM.

II.5 NOISE CLEANING

An image may be subject to noise and interference frorn several sources including electrical sensor noise, photographic grain noise, and channel errors. The noise effects can be minimized by classical filtering techniques summarized below.

Image noise arising from a noisy sensor or channel transmission errors usually appears as discrete isolated pixel variations that are not spatially correlated. Pixels that are in error often appear markedly different from their neighbors. This observation is the basis of many noise-cleaning algorithms. [22] proposes a simple " out-range" noise cleaning method. With this technique, each pixel is sequentially examined, and if the magnitude of a pixel is greater than the average brightness of its immediate neighbors by some threshold level, it is replaced by the average value.

II.5.1 Low-Pass masks

Noise in an image generally has a higher spatial frequency spectrum than the normal image components because of its spatial decorrelatedness. Hence, simple low-pass filtering can be effective for noise smoothing. An output N x N image array Q is formed by discrete convolution of the input N x N image array F with the L x L convolution array H according to the relation,

$$Q(m_1, m_2) = \sum_{n_1} \sum_{n_2} F(n_1, n_2) H(m_1 - n_1 + 1, m_2 - n_2 + 1) \qquad (14)$$

For noise smoothing, H should be a low-pass form with all positive components. Several convolution arrays of low-pass form are suggested,

Mask1:

$$H = 1/9 \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Mask2:

$$H = 1/10 \quad \begin{matrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Mask3:

$$H = 1/16 \quad \begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$

These arrays, which are often called noise-cleaning masks, are normalized to unit weighting so that the noise-cleaning does not introduce a brightness bias in the processed image.

For binary images, the linear filtering operation should be followed by some kind of clipping, which usually results in a distortion of many important image characteristics.

II.5.2 Median Filters

Median filtering is a nonlinear signal processing technique developed by Tukey [23] that is useful for noise suppression in images. In one-dimensional form the median filter consists of a sliding window encompassing an odd number of pixels. The center pixel in the window is replaced by the median of the pixels within the window. The median of a discrete sequence $a_1, a_2, \ldots, a$, for N odd is that member of the sequence for which $(N-1)/2$ elements are smaller or equal in value, and $(N-1)/2$ elements are larger or equal in value. For example, if the values of the pixels within a window are 80,90,200,110,120, the center pixel would be replaced by the value 110, which is the median value of the sorted sequence 80,90,110,120,200. In this example, if the value 200 were a noise spike in a monotonically increasing sequence, the median filter would result in considerable improvement. On the other hand, the value 200 might represent a valid signal pulse for a wide-bandwidth sensor, and the resultant image would suffer some loss of resolution. Thus in some cases the median filter will provide: noise suppression, and in other cases it will cause signal suppression. (versus our method shown later where a weight is given to the distance between the restored and the received images to prevent this case from happening).

Operation of the median filter can be analyzed to a limited extent. It can be shown that the median of the product of a constant K and a sequence f(j) is

$$\text{med}\{ Kf(j) \} = K\ \text{med}\{ f(j) \}$$

Furthermore,

$$\text{med}\{ K + f(j) \} = K + \text{med}\{f(j)\}$$

However, for two arbitrary sequences f(j) and g(j) it does not follow that the median of the sum of the sequences is equal to the sum of their medians. That is in general,

$$\text{med}\{ f(j) + g(j) \} \neq \text{med}\{ f(j) \} + \text{med}\{ g(j) \}$$

The sequence 80,90,100,110,120 and 80,90,100,90,80 are examples for which the additive linearity property does not hold.

There are various strategies for application of the median filter for noise suppression. One method would be to try a median filter with a window of length three. If there is no significant signal loss, the window length could be increased to five for median filtering of the original. The process would be terminated when the median filter begins to do more harm than good. It is also possible to perform cascaded median filtering on a signal using a fixed or variable length window. In general, regions that are unchanged by a single pass of the filter will remain unchanged in subsequent passes. Regions in which the signal period is lower than one half the window width will be continually altered by each successive pass. Usually, the process will continue until the resultant period is greater than one-half the window width, but it can be shown that some sequences will never converge. The concept of the median filter can be easily extended to two dimensions by utilizing a two-dimensional window of some desired shape such as a rectangle. It is obvious that a two-dimensional L x L median filter will provide a greater degree of noise suppression that sequential horizontal and vertical processing with L x 1 median filters, but two-dimensional processing also results in greater signal suppression. In general, the median filter is very effective in reducing 'salt & pepper' noise, and will be used in our simulations for comparison with other methods.

CHAPTER III

NEURAL NETWORKS

III.1 ARTIFICIAL VS. BIOLOGICAL NETWORKS

Anyone can see that the human brain is superior to a digital computer at many tasks. A good example is the processing of visual information , a one year old baby is much better and faster at recognizing objects and faces than even the most advanced AI system running on the fastest supercomputer. The brain has many other features that would be desirable in artificial systems[24],

* It is robust and fault tolerant. Nerve cells in the brain die every day without affecting its performance significantly.

* It is flexible. It can easily adjust to a new environment by "learning". It does not have to be programmed in C or Pascal.

* It can deal with information that is fuzzy, probabilistic, noisy or inconsistent.

* It is highly parallel.

* It is small, compact, and dissipates little power.

Only in tasks based primarily on simple arithmetic does the computer outperform the brain!

This was the real motivation for studying neural computation.It is an alternative computational paradigm to the usual one (based on a programmed instruction sequence), which was introduced by von Neumann and has been used as the basic of almost all machine computation to date. It is inspired by knowledge from neuroscience, though it does not try to be biologically realistic in detail. The science of neural computation was originally aimed more towards modelling networks of real neurons in the brain. The models are extremely simplified when seen from a neurophysiological point of view, though we believe that they are still valuable for gaining insight of the separate principles of biological "computation".

III.1.1 The **Human Brain**

The human brain is composed of about 10^{11} neurons or nerve cells of many different types. Tree-like networks of nerve fiber called dendrites are connected to the cell body or soma, where the cell nucleus is located. Extending from the cell body is a single long fiber called the axon, which eventually branches or arborizes into strands and substrands. At the ends of these are the transmitting ends of the synaptic junctions, or synapses to other neurons. The receiving ends of these junctions on other cells can be found both on the dendrites and on the cell bodies themselves. The axon of a typical neuron makes a few thousand synapses with other neurons. The transmission of a signal from one cell to another at a synapse is a complex chemical process in which specific transmitter substances are released from the sending side of the junction. The effect is to raise or lower the electrical potential inside the body of the receiving cell. If this potential reaches a threshold, a pulse or action potential of fixed strength and duration is sent down the axon. We then say that the cell has "fired". The pulse branches out through the axonal arborization to synaptic Junctions to other cells. After firing, the cell has to wait for a time called the refractory period before it can fire again.

III.1.2 The McCulloch and Pitts model

McCulloch and Pitts(1943) proposed a simple model of a neuron as a binary threshold unit. Specifically, the model neuron computes a weighted sum of its inputs from other units, and outputs a one or a zero according to whether this sum is above or below a certain threshold

$$n_i(t+1) = \theta\left(\sum w_{ij} n_j(t) - \mu_i\right) \quad (15)$$

In this equation, n_i is either 1 or 0, and represents the state of neuron i as firing or not firing respectively. Time t is taken as discrete, with one time unit elapsing per processing step. $\theta(x)$ is the unit step function, or Heaviside function

$$\theta(x) = \begin{cases} 1 & if\ x \geq 0; \\ 0 & otherwise \end{cases}$$

The weight w_{ij} represents the strength of the synapse connecting neuron j to neuron i. It can be positive or negative corresponding to an excitatory or inhibitory synapse respectively. It is zero if there is no synapse between i and j. The parameter μ_i is the threshold value for unit i. The weighted sum of inputs must reach or exceed the threshold for the neuron to fire. Though simple, a McCulloch-Pitts neuron is computationally a powerful device. McCulloch and Pitts proved that a synchronous assembly of such neurons is capable in principle of universal computation for suitably chosen weights

w_{ij}. This means that it can perform any computation that an ordinary digital computer can, though not necessarily so rapidly or conveniently.

Real neurons involve may complications omitted from this simple description,

* Real neurons are often not even approximately ithreshold devices. Instead they respond to their input in a continuous way. This is referred to as a graded response.

* Many real cells also perform a nonlinear summation of their inputs, which takes us a bit further from the McCulloch-Pitts picture.

* A real neuron produces a sequence of pulses, not a simple output level. Representing the firing rate by a single number like n_i, even if continuous, ignores much information that might be carried by such a pulse sequence.

* Neurons do not all have the same fixed delay (t-->t+1). Nor are they updated synchronously by a central clock. We will in fact use asynchronous update in this thesis.

* The amount of transmitter substance released at a synapse may vary

unpredictably. This sort of effect can be modelled, by a stochastic generalization of the McCulloch-Pitts dynamics.

A simple generalization of the McCulloch-Pitts equation which includes some of these features is,

$$n_i = g\left(\sum_j w_{ij} n_j - \mu_i\right) \quad (17)$$

The number n_i is now continuous-valued and is called the state or activation of unit i. The threshold function $\theta(x)$ of (15) has been replaced by a more general nonlinear function $g(x)$ called the activation function, gain function, transfer function or squashing function. Rather than writing the time t or t+1 explicitly as we did in (15), we now simply give a rule for updating n_i whenever that occurs or asynchronously.

III.1.3 Parallel Processing

In computer science terms, we can describe the brain as a parallel .system of about 10^{11} processors. Using the simplified model (17) above, each processor has a very simple program: it computes a weighted sum of the input data from other processors and then outputs a single number, a nonlinear function of this weighted sum. This output is then sent to other processors, which are continually doing the same kind of calculation.

The high connectivity of the network means that errors in a few terms will probably be inconsequential. This tells us that such a system can be expected to be robust and its performance will degrade gracefully in the presence of noise or errors.The contrast between this kind of processing and the conventional von Neumann kind is very strong. Here we have very many processors, each executing a very simple program, instead of the conventional situation where one or at most a few processors execute very complicated programs. And in contrast to the robustness of a neural network, an ordinary sequential computation may easily be ruined by a single bit error.

Massive parallelism in computational networks is extremely attractive in principle. But in practice there are many issues to be decided before a successful implementation can be achieved for a given problem:

* What is the best architecture.
* What sort of activation function $g(x)$ should be used.
* What type of updating should be used: synchronous or asynchronous.

* How can a network be programmed? Can it learn a task or must it be predesigned.

* What can the various types of network do. How robust are they to missing information or incorrect data. Can they generalize to unknown examples.

* How can a network be built in hardware.

III.1.4 Programming

A very important question is: how do we choose the connection weights so the network can do a specific task?

In some examples, the weighs can be chosen a priori. This embeds some information into the network by design (This will be done in this thesis as shown later). In other cases, we can often teach the network to perform the desired computation by iterative adjustments of the w_{ij} strengths. This may be done in two main ways:

* Supervised learning. Here the learning is done on the basis of direct

comparison of the output of the network with known correct answers.

* Unsupervised learning. Sometimes the learning goal is not defined at all in terms of specific correct examples. The only available information is in the correlations of the input data or signals. The network is expected to create categories from these correlations, and to produce output signals corresponding to the input category.

CHAPTER IV

THE HOPFIELD NETWORK

Among the various neural networks architectures, the Hopfield neural network is very popular in real-world applications.

The Hopfield type networks can be used to implement ACAMS (Associative Content Addressable Memory). A Hopfield ACAM can be used as an image storage device in which stored images can be retrieved by their incomplete and/or distorted versions or as a part of an image classifier. The other important application of Hopfield networks is function optimization, which in image processing, can be used for the restoration of images from blurred and noisy versions. In [25], Hopfield describes how several optimization problems can be rapidly solved by highly interconnected networks of simple analog processors. The processing elements are modeled as amplifiers having a sigmoid monotonic input-output relation.

IV.1 HARDWARE REPRESENTATION OF THE HOPFIELD NETWORK

The function $V_j = g_j(u_j)$ which characterizes this input-output relation describes the output voltage V_j of amplifier j due to an input voltage u_j the time constants of the amplifiers are assumed negligible . However, each amplifier has an input resistor leading to a reference ground and an input capacitor. These components partially define the time constants of the network and provide for integrative analog summation of input currents are provided through resistors of conductance T_{ij} connected between the output and amplifier j and the input of amplifier i. In order to provide for output currents of both signs from the same processor, each amplifier is given two outputs, a normal output, and an inverted output. The minimum and maximum outputs of the normal amplifier are taken as 0 and 1, while the inverted output has corresponding values of 0 and -1. A connection between two processors is defined by a conductance T_{ij} which connects one of the two outputs of amplifier j to the input of amplifier i. This connection is made with a resistor of value $R_{ij}=1/|T_{ij}|$

If $T_{ij} > 0$, this resistor is connected to the normal output of amplifier j. If $T_{ij}<0$, it is connected to the inverted output of amplifier j. the rnatrix T_{ij} defines the connectivity among the processors. The net input current to any processor(and hence the input voltage u_i) is the sum of the currents flowing through the set of resistors connecting its input to the outputs of the other processors. Also, externally supplied input currents (I_i) are also present for each processor. In the circuits discussed here, these external inputs can be constant biases which effectively shift the input-output relation along the u_i axis and/or problem specific input currents which correspond to data in the problem.

Using the network parameters,i.e., states, interconnections and biases, one can define the energy of a Hopfield network. Under certain conditions,discussed below, the energy function always decreases upon the state update of any neuron. This enables us to develop a Hopfield network capable of processing the signal and evolving toward a minimum of its energy function.

IV.2 THE HOPFIELD ENERGY FUNCTION

Hopfield defined the energy function of the neural network with interconnection w_{ii} and bias θ to each neuron to be

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} z_i z_j - \sum_{i=1}^{N} \theta_i z_i \quad (18)$$

or

$$E = -\frac{1}{2} Z^T W Z - \theta^T Z \quad (19)$$

in vector form.

In his model, the neurons have no self feedback, i.e., the autoconnections w_{ii}, i=1,2,... M are zero and each neuron updates its stage randomly in time (asynchronously) according to the hard limit non linearity rule:

$$\sum_{j=1}^{N} w_{ij} z_j + \theta_i \geq 0 \qquad z_i^{new} = 1 \quad (20)$$

$$\sum_{j=1}^{N} w_{ij} z_j + \theta_i < 0 \qquad z_i^{new} = 0 \quad (20)$$

where binary states are assumed.

Hopfield showed that if $w_{ii} = 0$ and the interconnections are symmetric ($w_{ij}=w_{ji}$), then the energy change due to the state update of any neuron is always zero or negative. This is the so called energy reduction property. However, the condition of zero autoconnections is usually too restrictive in applications.

The relationship between stable states and energy local minima can be summarized [26]:

Fact 1: If $w_{ij} = w_{ji}$ and $w_{ii} >= 0$, then any neuron-state update according to the update rule (20) will monotonically decrease the energy of the system. Therefore local minima of the energy function are stable states. But if there is some wii strictly positive, then stable states may not all occur at local minima.

Fact 2: If $w_{ij} = w_{ji}$ and $w_{ii} <= 0$, then all stable states are local minima. But if there is some w_{ii} strictly negative, then the energy reduction property does not always hold if neurons are updated according to (20); therefore there may be local minima which do not correspond to stable states. :Furthermore, the system may not always be able to reach a stable state.

Fact 3: If $w_{ij} = w_{ji}$ and $w_{ii} = 0$, then stable states and local minima are identical, and the energy reduction property holds.

In our image restoration application, the diagonal terms w_{ii} are all negative. Since the autoconnections are negative, we have two potential probllems:

(i) Stable states of the network may occur at only some of the local minima and

(ii) the network may not be able to reach to a stable state at al'l. Both of these problems are induced by the fact that the energy reduction property may not hold with negative autoconnections (fact 1).

Many approaches have been suggested to overcome the problem of negative autoconnections. The EHE approach (Eliminating Highest error criterion) [27] suggests that at every update step the neurons, which have the highest percentage of back projection errors (where $H(k)= WX(k)- \theta$ is the projection error), are updated first if the connection matrix W is satisfied with

$$| W_{ij} | > | W_{ji} | \quad , i = 1 ,.....,n;$$
$$j \neq i, \; j = 1 ,...,n.$$

Another approach would be a stochastic decision rule [28],[29], where a Boltzmann distribution is defined by

$$\frac{P_{new}}{P_{old}} = e^{-\frac{\Delta E}{T}} \quad (22)$$

where P_{new} and P_{old} are the probabilities of the new and old global state, respectively, delta E is the energy change and T is the **parameter** which acts like temperature. **A** state $v_{new}(i,k)$ is taken if

$$\frac{P_{new}}{Pold} > 1 \ \textit{or if} \ \frac{P_{new}}{P_{old}} \leq 1 \ \textit{but} \ \frac{P_{new}}{Pold} > \epsilon \quad (23)$$

where epsilon is a random number uniformly distributed in the interval [0,1].

In this thesis, we use the more classical approach of enforcing the energy reduction property by modifying the update rule. The simplest way to achieve this is to update the state of a neuron according to (20), only if the update will decrease the energy, that is, if

$$\Delta E_i = -(\Delta z_i)\ [\sum_{j=1}^{N} w_{ij} z_j + \theta_i] - \frac{1}{2} w_{ii} (\Delta z_i)^2 \leq 0 \quad \textbf{(24)}$$

then the update is executed but if delta E > 0 then the update is not executed. In other words, the neural network has to have an inner mechanism to monitor the 'trial' energy change.

CHAPTER V

SINGLE STAGE VS. MULTISTAGE

V.1. THE SINGLE STAGE HOPFIELD NETWORK

Much work has been done on the restoration of degraded gray level images. In this thesis, we will consider bi-level (black and white) images degraded by speckle (salt and pepper) noise.

The error, as suggested in [30], representing a cost energy function E, is expressed as :

$$E = R + \lambda D \quad (25)$$

where λ is a number that parametrizes the desired trade-off between roughness and discrepancy for the smoothed image ..

For D we can take either the L^1 or the L^2 distance between K and M, K being our received degraded image, and M its estimate. L^1 and L^2 are equivalent for bi-level images since, for any values of K and M in (0,1), one has rrivially $|K-M|=(K-M)^2$. Thus we can choose

$$D=\sum_{i=1}^{N}\sum_{j=1}^{N}(k_{ij}-m_{ij})^2 \quad (26)$$

For R, the simplest choice proposed in [31] (The Digital Laplacian) results in:

$$R=\sum_{i_1=1}^{N}\sum_{j_1=1}^{N}\sum_{i_2=1}^{N}\sum_{j_2=1}^{N}C_{i_1j_1i_2j_2}(m_{i_1j_1}-m_{i_2j_2})^2 \quad (27)$$

In this way, the problem of finding M is well defined, and consists of minimizing E given by (25) for given K and λ, with D and R defined as in (26) and (27).

We reformulate the problem in terms of a bi-level image with values (-1,1) instead of (0,1). Thus we define

$$y_{ij} = 2\,k_{ij} - 1 \quad \textbf{(28)}$$

and

$$z_{ij} = 2\,m_{ij} - 1 \quad \textbf{(29)}$$

Since K and M can only take the values 0 and 1, Y and Z correspondingly take the values -1 and +1. We need to express D and R in terms of these new variables.

We first write the following identities, which can be easily verified:

$$y_{ij}^2 = 1 \quad \textbf{(30)}$$

$$z_{ij}^2 = 1 \quad (31)$$

$$(k_{ij} - m_{ij})^2 = \frac{1 - y_{ij} z_{ij}}{2} \quad (32)$$

$$(m_{i_1 j_1} - m_{i_2 j_2})^2 = \frac{1 - z_{i_1 j_1} z_{i_2 j_2}}{2} \quad (33)$$

Using these identities, we get the following new expressions for D

and R:

$$D = \frac{1}{2} N^2 - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_{ij} z_{ij} \quad (34)$$

$$R = \frac{1}{2}\sum_{i_1=1}^{N}\sum_{j_1=1}^{N}\sum_{i_2=1}^{N}\sum_{j_2=1}^{N} C_{i_1j_1i_2j_2} - \frac{1}{2}\sum_{i_1=1}^{N}\sum_{j_1=1}^{N}\sum_{i_2=1}^{N}\sum_{j_2=1}^{N} C_{i_1j_1i_2j_2}\, z_{i_1j_1}\, z_{i_2j_2} \quad (35)$$

The first term in R is simply half the number of neighbor :pairs and equals $N^2 (N^2 - 1)$ Thus,

$$R = N^2(N^2-1) - \frac{1}{2}\sum_{i_1=1}^{N}\sum_{j_1=1}^{N}\sum_{i_2=1}^{N}\sum_{j_2=1}^{N} C_{i_1j_1i_2j_2}\, z_{i_1j_1}\, z_{i_2j_2} \quad (36)$$

Expression (1) now becomes:

$$E = -\frac{1}{2}\sum_{i_1=1}^{N}\sum_{j_1=1}^{N}\sum_{i_2=1}^{N}\sum_{j_2=1}^{N} C_{i_1j_1i_2j_2}\, z_{i_1j_1}\, z_{i_2j_2} - \frac{\lambda}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_{ij}\, z_{ij} \quad (37)$$

where we dropped the constant terms which do not depend on Z. We can do that because addition of a term independent of Z does not change the point where E reaches its minimum.

Using Lexicographic notation, we change our image matrix into a vector, and accordingly, our weight matrix is changed into a two dimensional matrix.

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}\, z_i\, z_j - \frac{\lambda}{2}\sum_{i=1}^{N} y_i z_i \quad (38)$$

while the Hopfield model, as mentioned earlier in (17) takes the form,

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\, z_i\, z_j - \sum_{i=1}^{N} \theta_i\, z_i \quad (6)$$

Mapping to the Hopfield error function, we get:

$$W = C \qquad (26)$$

$$\theta = \frac{\lambda}{2}\, Y \quad (40)$$

V.2. PARTITIONING OF THE HOPFIELD NETWORK

The objective of partitioning the Hopfield network is to address the problem of error minimization locally within each partition while the other partition is frozen. The partitioning scheme actually resembles the iterative interlacing approach for the synthesis of computer-generated holograms[32]. By local minimization of state variables,, while the other partition is frozen, the network is able to reach a deeper minima.

V.2.1 Literature on Multistage approach

Various Divide and Conquer Algorithms in Neural Networks have been developped with different goals. In [33], the authors describe a DCN which creates a feedforward neural network architecture during training, based upon the training examples. The first cell introduced on any layer is trained on all examples. Further cells on a layer are trained primarily on examples not already correctly classified. The algorithm is designed to dynamically create a network architecture based upon the domain examples presented. This architecture enables learning to be effectively done, and does not require any preconceived guess or domain-specific knowledge and analysis. Training is done on the weights of the input links to one cell at any given time. Hence, back propagation of error through hidden cells is never required.

In [34], the Hopfield network multistage approach was discussecl for the restoration of gray level images blurred and degraded by uniform gaussian noise, for the purpose of ease of implementation in large scale problems. The authors also had problems with convergence, since the image vector was 128^3 X 1. (It is 128^2 X 1 for the bilevel images case.)

V.2.2 Two Stage Decomposition

In this study, we split our pixels into odd pixels (Z_o) and even pixels (Z_e)

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}\,(z_{oi}+z_{ei})\,(z_{oj}+z_{ej}) - \frac{\lambda}{2}\sum_{i=1}^{N} y_i\,(z_{oi}+z_{ei}) \tag{41}$$

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}\ (z_{oi}\,z_{oj} + z_{oi}\,z_{ej} + z_{ei}\,z_{oj} + z_{ei}\,z_{ej}) - \frac{\lambda}{2}\sum_{i=1}^{N} y_i\ (z_{oi} + z_{e}$$
(42)

$$E_{odd} = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}\,z_{oi}\,z_{oj} - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}z_{oi}z_{ej} - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}z_{ei}z_{oj} - \frac{\lambda}{2}\sum_{i=1}^{N} y_i z$$
(43)

$$= -\frac{1}{2}\,z_o^T C\,z_o - \frac{1}{2}\,z_o^T C\ z_e - \frac{1}{2}\,z_e^T C\ z_o - \frac{\lambda}{2}\ Y\ z_o \quad \textbf{(44)}$$

$$= -\frac{1}{2}\,z_o^T C\ Z_o - \frac{1}{2}\ (z_e^T C^T z_o)^T - \frac{1}{2}\,z_e^T C\,z_o - \frac{\lambda}{2}\ Y z_o \quad \textbf{(45)}$$

$$= -\frac{1}{2}\,z_o^T C\ Z_o - \frac{1}{2}\ (z_e^T C z_o)^T - \frac{1}{2}\,z_e^T C\,z_o - \frac{\lambda}{2}\ Y z_o \quad \textbf{(46)}$$

(C^T = C **,** since C is symmetrical)

$$= \frac{1}{2} z_o^T C z_o - z_e^T C z_o - \frac{\lambda}{2} Y z_o \quad (47)$$

$$= \frac{1}{2} z_o^T C z_o - (z_e^T C + \frac{\lambda}{2} Y) z_o \quad (48)$$

By mapping our error function to the Hopfield error function again, we get

$$\boxed{W = C \qquad ; \qquad \theta = + (z_e^T C + \frac{\lambda}{2} Y) \quad (49)}$$

The two partitions algorithm can be summarized as follows:

(a) Perform energy descent with respect to Even pixels to a minimum while Odd are frozen.

(b) Switch to Odd and repeat (a) with Even and Odd interchanged.

(c) Repeat steps (a) and (b) until there is no further descent in error energy.

V.2.3 Four Stage Decomposition

In this case we split our pixels into four groups (Z_1, Z_2, Z_3 and Z_4)

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}(z_{1i}+z_{2i}+z_{3i}+z_{4i})(z_{1j}+z_{2j}+z_{3j}+z_{4j}) - \frac{\lambda}{2}\sum_{i=1}^{N} y_i(z_{1i}+z_{2i}+z_{3i}+z_{4i}) \tag{50}$$

$$\begin{aligned} E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}(&z_{1i}z_{1j}+z_{1i}z_{2j}+z_{1i}z_{3j}+z_{1i}z_{4j}+z_{2i}z_{1j}+z_{2i}z_{2j} \\ &+z_{2i}z_{3j}+z_{2i}z_{4j}+z_{3i}z_{1j}+z_{3i}z_{2j}+z_{3i}z_{3j}+z_{3i}z_{4j}+z_{4i}z_{1j}+z_{4i}z_{2j} \\ &+z_{4i}z_{3j}+z_{4i}z_{4j}) - \frac{\lambda}{2}\sum_{i=1}^{N} y_i(z_{1i}+z_{2i}+z_{3i}+z_{4i}) \end{aligned} \tag{51}$$

to find E_1 we neglect the terms that do not include z_1 , which are constant in this case.

$$E_1 = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}(z_{1i}z_{1j} + z_{1i}z_{2j} + z_{1i}z_{3j} + z_{1i}z_{4j} + z_{2i}z_{1j} + z_{3i}z_{1j} + z_{4i}z_{1j}) - \frac{\lambda}{2}\sum_{i=1}^{N} y_i z_{1i} \tag{52}$$

back to vector notation

$$\begin{aligned} E_1 &= -\frac{1}{2}z_1^T C z_1 - \frac{1}{2}z_1^T c z_2 - \frac{1}{2}z_1^T C z_3 - \frac{1}{2}z_1^T C z_4 - \frac{1}{2}z_2^T C z_1 \\ &\quad -\frac{1}{2}z_3^T C z_1 - \frac{1}{2}z_4^T C z_1 - \frac{\lambda}{2}Y z_1 \\ &= -\frac{1}{2}z_1^T C Z_1 - \frac{1}{2}(z_2^T \mathbf{C}^T Z_1)^T - \frac{1}{2}(z_3^T \mathbf{C}^T z_1)^T - \frac{1}{2}(z_4^T \mathbf{C}^T z_1)^T \\ &\quad -\frac{1}{2}z_2^T C Z_1 - \frac{1}{2}z_3^T C z_1 - \frac{1}{2}z_4^T C z_1 - \frac{\lambda}{2}Y z_1 \end{aligned} \tag{53}$$

C being symetrical, $C^T = C$

$$E_1 = -\frac{1}{2} z_1^T C z_1 - \frac{1}{2}(z_2^T C z_1)^T - \frac{1}{2}(z_3^T C z_1)^T - \frac{1}{2}(z_4^T C z_1)^T - \frac{1}{2} z_2^T C z_1 - \frac{1}{2} z_3^T C z_1 - \frac{1}{2} z_4^T C z_1 - \frac{\lambda}{2} Y z_1 \quad (54)$$

The transpose of a number is the same as the number,

$$E_1 = -\frac{1}{2} z_1^T C z_1 - \frac{1}{2} z_2^T C z_1 - \frac{1}{2} z_3^T C z_1 - \frac{1}{2} z_4^T C z_1 - \frac{1}{2} z_2^T C z_1 - \frac{1}{2} z_3^T C z_1 - \frac{1}{2} z_4^T C z_1 - \frac{\lambda}{2} Y z_1 \quad (55)$$

$$E_1 = -\frac{1}{2} z_1^T C z_1 - z_2^T C z_1 - z_3^T C z_1 - z_4^T C z_1 - \frac{\lambda}{2} Y z_1 \quad (56)$$

▼≡ G

$$E_1 = -\frac{1}{2} z_1^T C z_1 - \{(z_2^T + z_3^T + z_4^T) C + \frac{\lambda}{2} Y\} z_1 \quad (57)$$

by mapping E_1 to the **hopfield** energy fuction we get'

$$\boxed{W = C \quad ; \quad \theta = (z_2^T + z_3^T + z_4^T) C + \frac{\lambda}{2} Y \quad \textbf{(58)}}$$

CHAPTER VI

OPERATING CONDITIONS

VI.1 The Matlab Software

In previous work, the problem of convergence was faced by researchers because of the large size of the images. This problem of convergence was not faced in our case, since a very powerful tool of the Matlab software was used, the block processing function. With this function, the image matrix is partitioned into blocks, which size is chosen by the user.

Since the blocks chosen were of reasonable size, [20x20] in our case, convergence to a minima was never a problem.

Besides helping with the convergence problem, the block processing function made it possible to process large images (128 X 128) on a 486 personal computer, which otherwise would have been impossible because of memory limitations.

The Matlab software, together with the image processing toolbox, proved themselves to be a very powerful tool for image processing applications.

Because if was designed for this kind of applications, this software offers a lot of convenient features,

* Vectorizing the operations saves on computational speed especially in those cases where the amount of data processed is very large.

* Block processing the images saves on speed and memory and makes the processing of any size image possible on a personal computer.

* Filtering functions are readily available and easy to use.

* Functions to show the processed images on the screen or send. them to the printer.

* Ease of programming

VI.2 Regularization Matrices

Two different regularization matrices were used in the: simulations. The digital Laplacian operator that takes the form,

$$d = \begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}$$

(where 4 neighboring pixels' effect is taken into account)

and the second-difference approximation of the **Laplace operator** that takes the form,

$$d = \begin{matrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{matrix}$$

(where 8 neighboring pixels' effect is taken into account:).

The weights matrix, consisting of the Laplacian operator is block circulant of the form:

$$
\begin{array}{ll}
 & Do\ D1\ \ 0\ \ 0\ \ 0\ \ldots\ldots D1\ D1 \\
 & D1\ Do\ D1\ \ 0\ \ 0\ \ldots\ldots\ldots D1 \\
 & 0\ \ D1\ Do\ D1\ \ 0\ \ldots\ldots\ldots \\
 & 0\ \ 0\ \ D1\ Do\ D1\ \ldots\ldots\ldots \\
Wt= & .\ \ 0\ \ 0\ D1\ Do\ D1\ \ldots\ldots \\
 & \\
 & \\
 & D1 \\
 & D1\ D1\ \ 0\ \ldots\ldots\ldots\ldots\ldots Do
\end{array}
$$

where in the case of the digital Laplacian Do and D1 take the form,

$$
\begin{array}{ll}
 & -4\ \ 1\ 0\ 0\ .\ .\ . \\
Do= & \ 1\ -4\ 1\ 0\ .\ .\ . \\
 & \ 0\ 1\ -4\ 1\ .\ .\ .
\end{array}
$$

$$D1 = \begin{matrix} 1 & 0 & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & \dots \end{matrix}$$

and in the case of the second-difference approximation of the Laplacian, Do and D1 take the form,

$$Do = \begin{matrix} -20 & 4 & 0 & 0 & .. \\ 4 & -20 & 4 & 0 & .. \\ 0 & 4 & -20 & 4 & .. \end{matrix}$$

$$D1 = \begin{matrix} 4 & 1 & 0 & 0 & .. \\ 1 & 4 & 1 & 0 & .. \\ 0 & 1 & 4 & 1 & .. \end{matrix}$$

VI.4. The Noise Level

In our simulations, 'Salt & Pepper ' noise was used. Each pixel was changed from 0 to 1 with probability r, r taking the values 0.3, 0.4, and 0.5.

In each of our images, the average value of y is r in originally white areas, 1-r in originally black areas. The difference between the two average values is 1-2r. The standard deviation of y in both the originally white and originally black areas is given by (sqrt r(1-r)). Thus we can define a signal to noise ratio as the ratio of the difference between the two average values and the standard deviation given by

$$\frac{(1-2r)}{\sqrt{r(1-r)}}$$

This gives signal-to-noise ratios of 0.87, 0.41,....

VI.5 The Weighting factor I,

Different values of the weighting factor I, were chosen, ranging from 0.5 to 3.5.

CHAPTER VII

RESULTS AND PERFORMANCE EVALUATION

We confine our experiments to a 120 x 120 bi-level image. Each pixel intensity can take a value of 0 (black) or 1 (white). The image restoration problem is considered for four diferent signal to noise ratios, a variety of regularization constants ranging from 0.5 to 3.5 (only the results for $\lambda=1$ and $\lambda=3$ are discussed below to avoid redundancy), and two different regularization matrices (four pixels neighborhood and eight pixels neighborhood).
A comparison was made among the following configurations:

(1) A single stage Hopfield network which addresses the total error

(2) A two stage Hopfield network using even numbered neurons first followed by the odd numbered neurons.

(3) A four stage Hopfield network using the pixels number 1,5,9,13,etc.. first, followed by pixels number 2,6,10,14,etc.. and so on

(4) A hybrid structure using a single stage approach followed by a two stage.

(5) A hybrid structure using a single stage approach followed by a four stage.

For all cases the direct methods start with the degraded image as the initial estimate, while the hybrid methods start with an image attained when the energy descent for the single stage slows down.

All results mentioned were obtained by averaging over a total of ten simulations.

Case I: $\lambda=1$; 4 pixels neighborhood Laplacian regularization matrix

The table on p.69 and the graph on p.70 show that the nnulti-stage approaches, whether direct or hybrid yield to a deeper minima than the one obtained by the single stage. The two stage and four stage approaches yield very comparable results. The results also suggest that the lower the signal to noise ratio (or the higher the noise density), the wider the gap between the single and multi-stage approaches.

Case II: $\lambda=3$; 4 pixels neighborhood Laplacian regularization matrix

The table on p.69 and the graph on p.71 show again that the multi-stages approaches yield to better results than the single stage. For this value of the regularization constant, the gap between the single stage and multi-stage energy functions doesn't increase with the increment of the noise density.

Case III: $\lambda=1$; 8 pixels neighborhood Laplacian regularization matrix

The table on p.69 and the graph on p.72 show one more time the lower energy function obtained by the multi-stage approach.

It is noticed that the values of the energy functions obtained in this case are lower than the ones obtained previously for the same value of the regularization constant, despite the fact that in both cases we start with the same image having the same initial energy function. This is due to the different nature of the Laplacian regularization matrix.

Another interesting issue was to find the optimal regularization constant for our processed image. To be able to compare the performance at different value of A without introducing a bias, a 'cost discrepancy' measure is introduced [35]

$$= (E_r - E_0) / |E_0|$$

Such a measure eliminates dependence on the choice of the parameters and the lower the cost discrepancy is, the better is the minimization of E (while a negative value implies that the restoration method has obtained a solution of lower cost than the original image).

E_r was taken to be the energy function resulting from the double stage approach. The values of E_0 and E_r were averaged over the different values of r to prevent any bias introduced by the value of r.

It was found that $\lambda=3$ was the optimal regularization constant for our image. It is understood that this is an experimental result and is dependent, amongst other factors, on the type of image used. We do not address this problem nnathematically in this thesis but allude to it as an important future objective.

CHAPTER VIII

CONCLUSIONS AND FUTURE CONSIDERATIONS

The problem of image restoration has been rephrased as a problem in combinatorial optimization using a partitioned Hopfield network. The results obtained show that the partitioning techniques developed are effective in reaching a deeper minima than the one obtained with single stage approaches. The tradeoff' is that complete parallelism is sacrificed since there is one active partition at each time instant while the other partitions are frozen. This tradeoff can be advantageous in applications in which optical processing is too fast with respect to the environment.

The above strategies should be tested on larger images. The implementations are computationally intensive and an effort to adopt parallel machines to accelerate the convergence should be investigated.

The choice of the regularization factor **I** was arbitrary, and many different values were experimented, leading to different results as was shown in the figures and graphs shown in the previous chapter. Some research has been done[26] on using a neural network to learn the optimal value of **I**, this could be incorporated in future research.

The idea of splitting a network into more than one stage was proven successful, and although the odd-even splitting is a most natural choice, natural extensions would include:

- Splitting the network into n subnetworks in order to adapt the reception of the image to the speed of the machine.
- Splitting the network into $n=2^m$ networks and attempt a recursive (binary tree) approach to the computation where each level of the computation is itself split into odd-even dat. The depth of the tree is then controlled by issues of availability of data and communication between processors.
- Alternate architectures could be developed to exploit the capabilities of each subnet.

TABLE 1
LAMBDA = 1 ; 4 PIXELS NEIBORHOOD

r	SINGLE STAGE	DOUBLE STAGE	1 + 2	FOUR STAGE	1 + 4
0.3	-0.30902	-0.32753	-0.33160	-0.32890	-0.33170
0.4	-0.25470	-0.27300	-0.27950	-0.27850	-0.27890
0.5	-0.16200	-0.18350	-0.19670	-0.19600	-0.19750

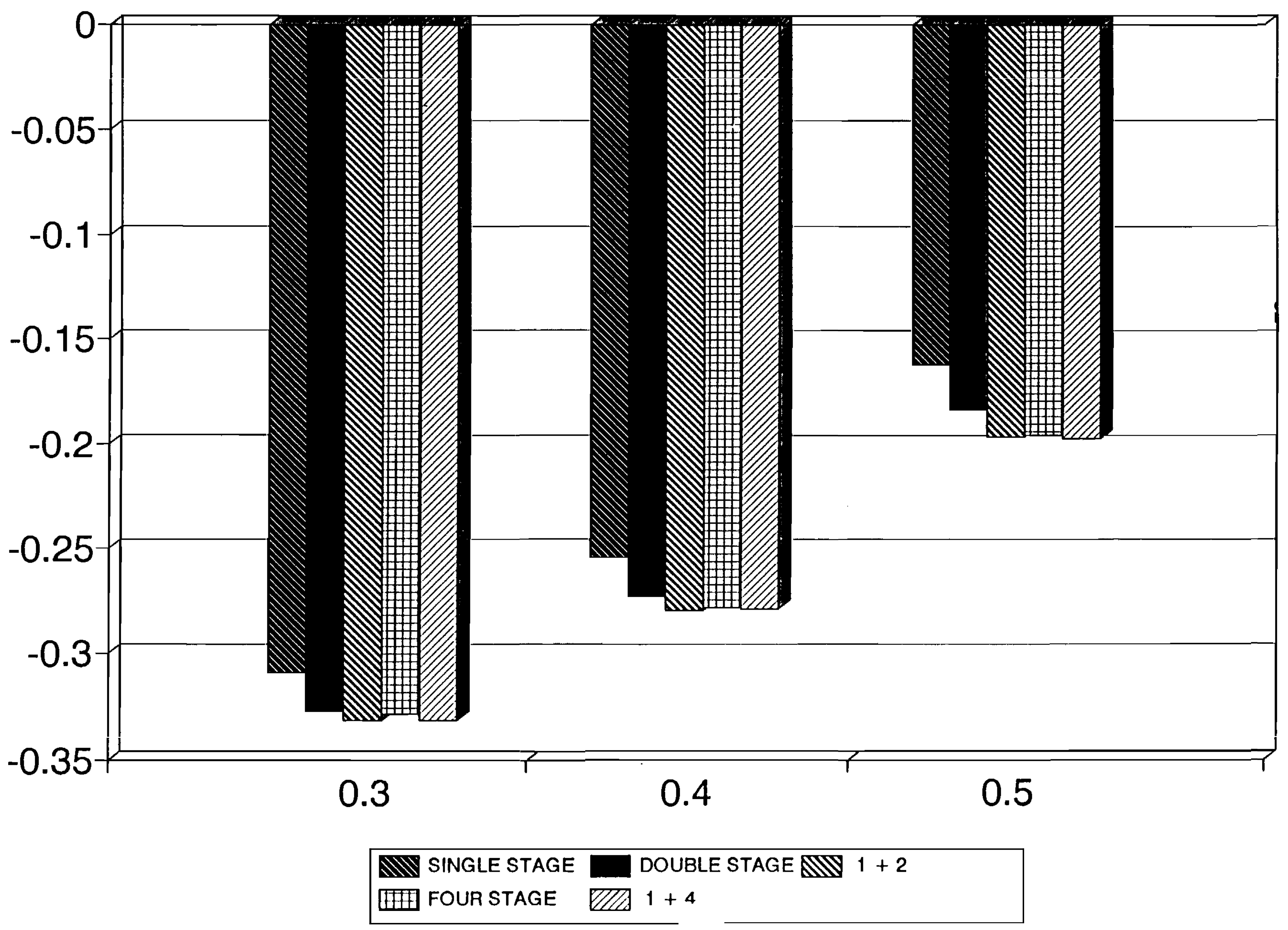
TABLE 2
LAMBDA = 3 ; 4 PIXELS NEIBORHOOD

r	SINGLE STAGE	DOUBLE STAGE	1 + 2	FOUR STAGE	1 + 4
0.3	-0.91120	-0.92110	-0.92590	-0.92340	-0.92560
0.4	-0.81590	-0.82820	-0.83490	-0.83160	-0.83470
0.5	-0.72510	-0.73620	-0.74760	-0.73800	-0.74750

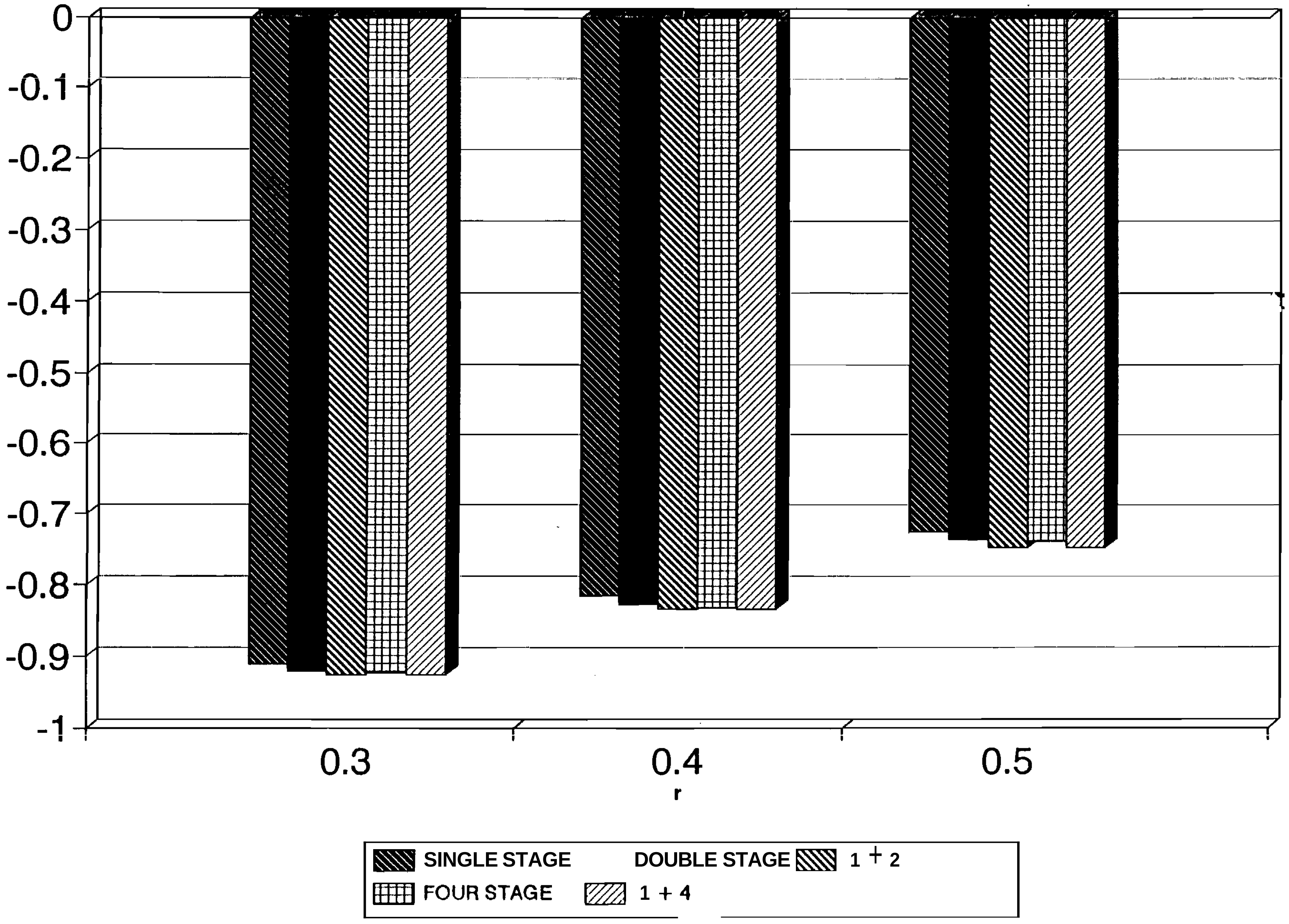
TABLE 3
LAMBDA = 1 ; 8 PIXELS NEIBORHOOD

r		DOUBLE STAGE	1 + 2	FOUR STAGE	1 + 4
0.3	-0.33890	-0.34400	-0.34710	-0.34420	-0.34710
0.4	-0.28940	-0.29250	-0.30100	-0.29370	-0.30100
0.5	-0.23460	-0.23950	-0.24930	-0.24020	-0.24880

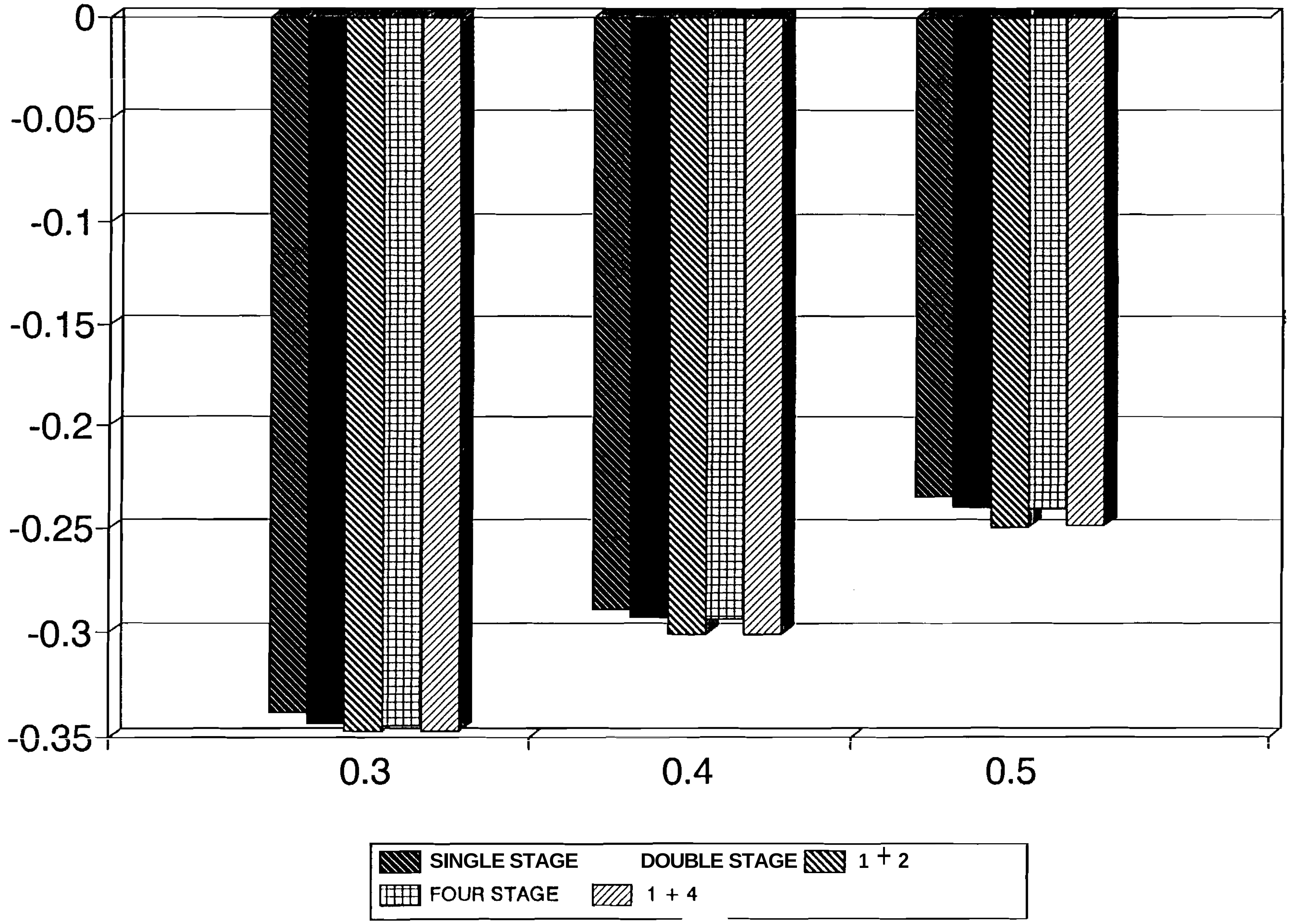
LAMBDA = 1 ; 4 PIXLES NEIBORHOOD
COMPARISON



LAMBDA = 3 ; 4 PIXLES NEIBORHOOD
COMPARISON



LAMBDA = 1 ; 8 PIXLES NEIBORHOOD
COMPARISON



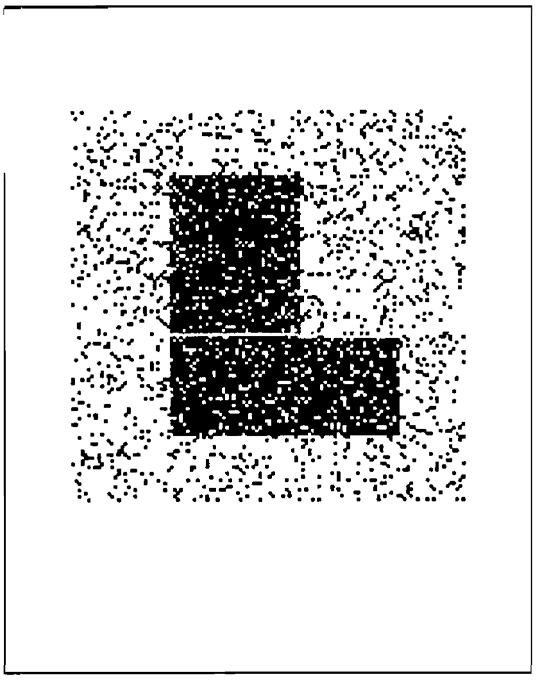


Figure (a)

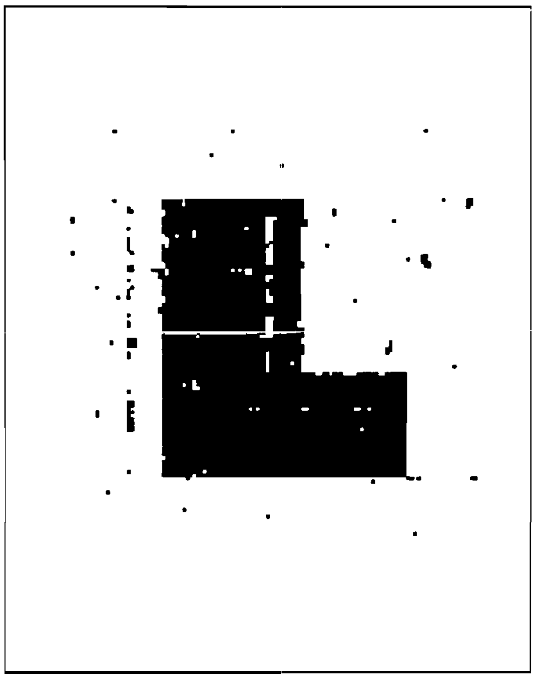


Figure (b)

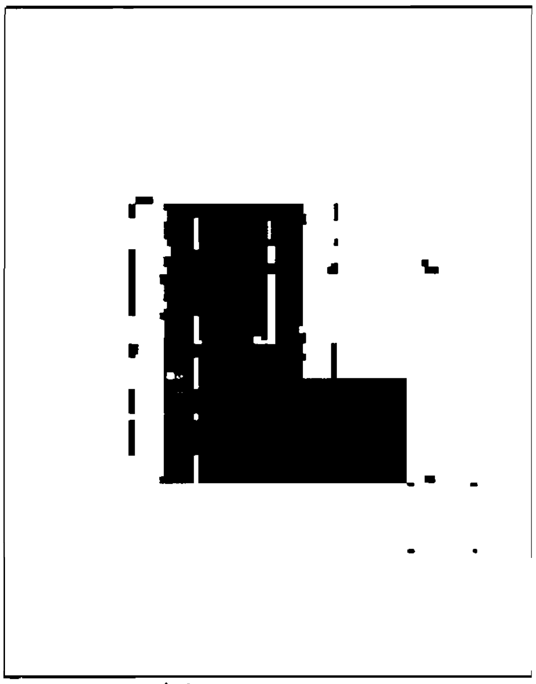


Figure (c)

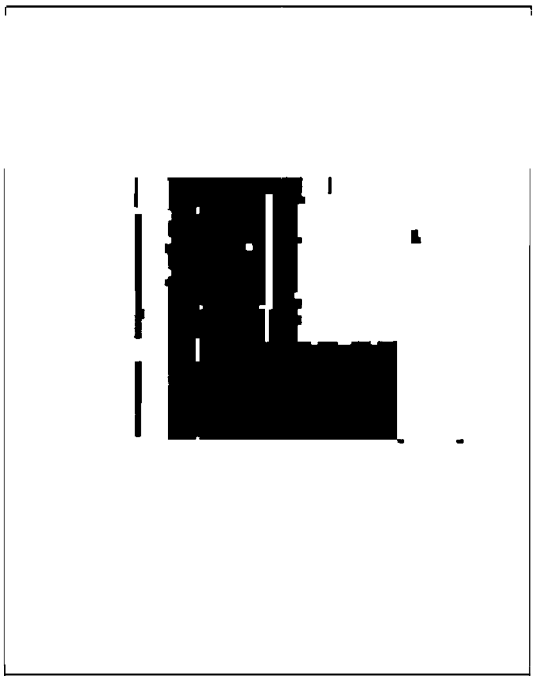


Figure (d)

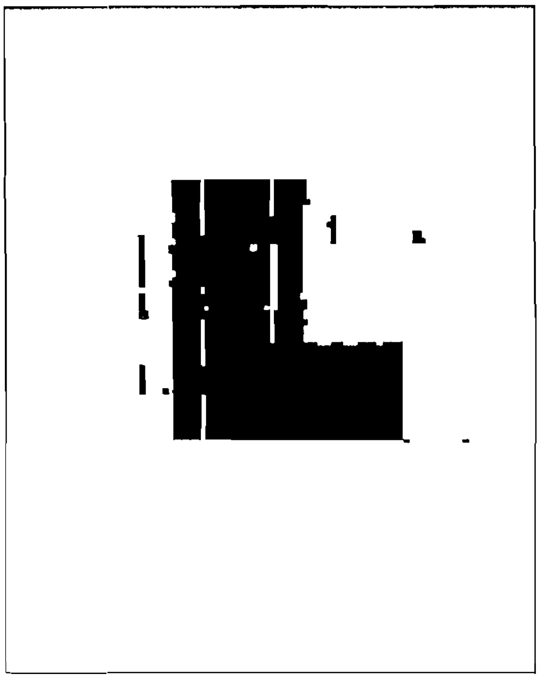


Figure (e)

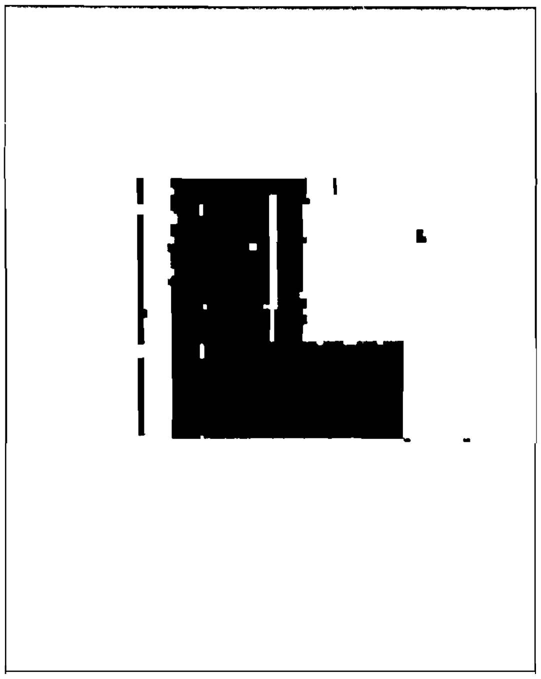


Figure (f)

Case 1:λ =1 ; 4-pixels neighborhood regularization matrix;r= **0.3**

Figure(a): Noisy Image

Figure(b): Siagle Stage Restoration

Figure(c): Double Stage Restoration

Figure(d): Single followed by Double Stage Restoration

Figure(e): Four Stage Restoration

Figure(f): Single followed by Four Stage Restoration

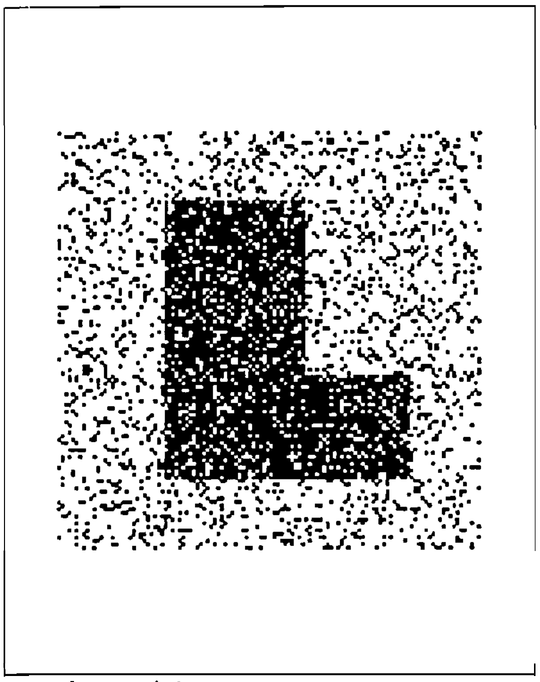


Figure (a)

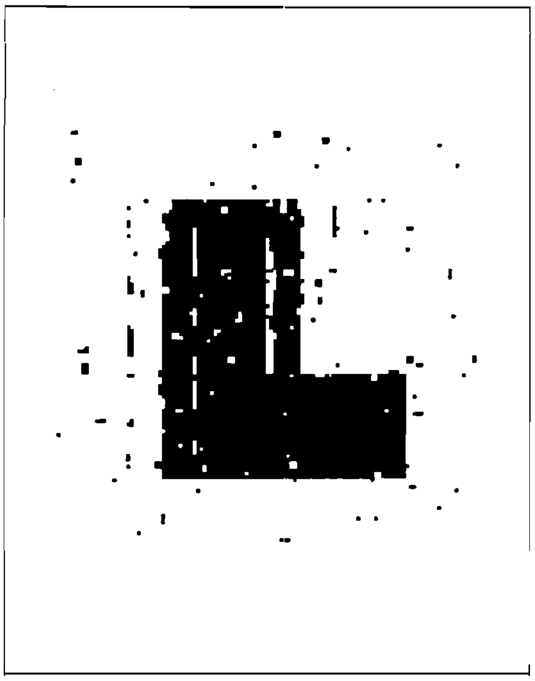


Figure (b)

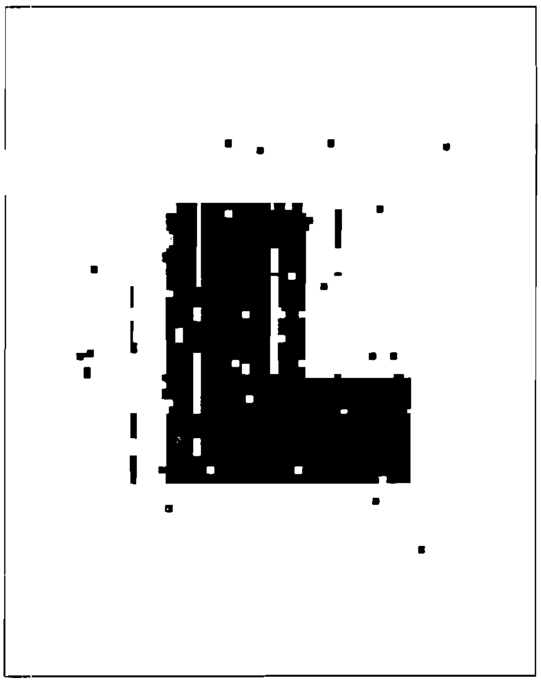


Figure (c)

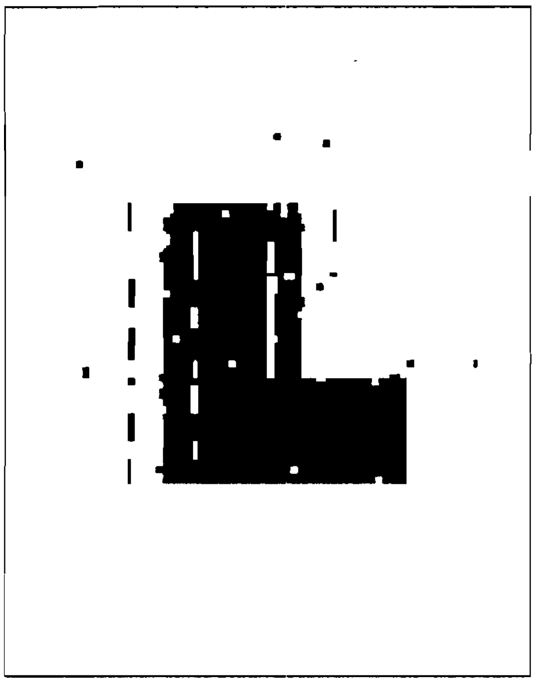


Figure (d)

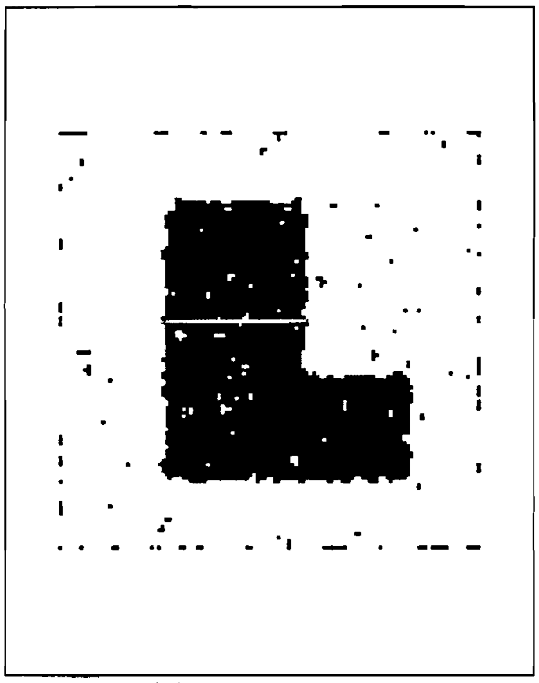


Figure (e)

Case 1:λ =1;4-pixel neighborhood regularization matrix; r=4

Figure(a): Noisy image

Figure(b): Single Stage Restoration

Figure(c): Double Stage Restoration

Figure(d): Single followed by Double Stage Restoration

Figure(e): Median filter (3x3 window) restoration

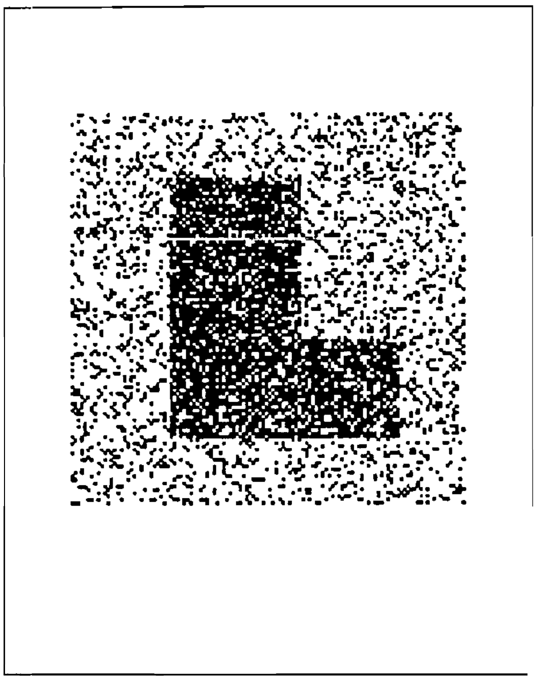


Figure (a)

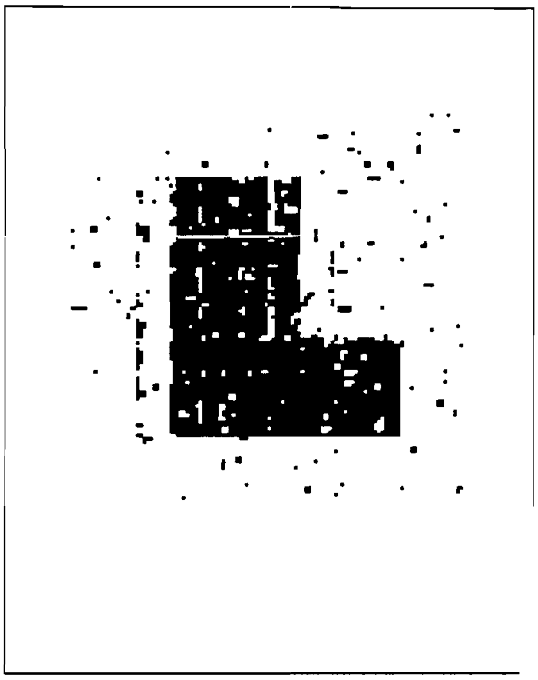


Figure (b)

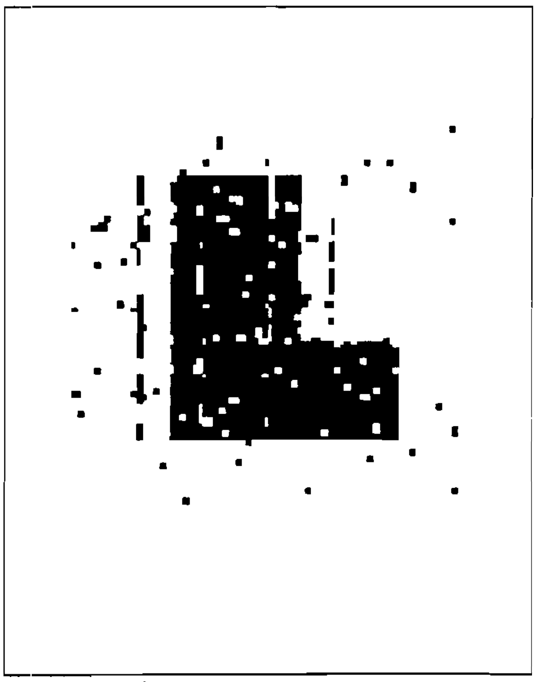


Figure (c)

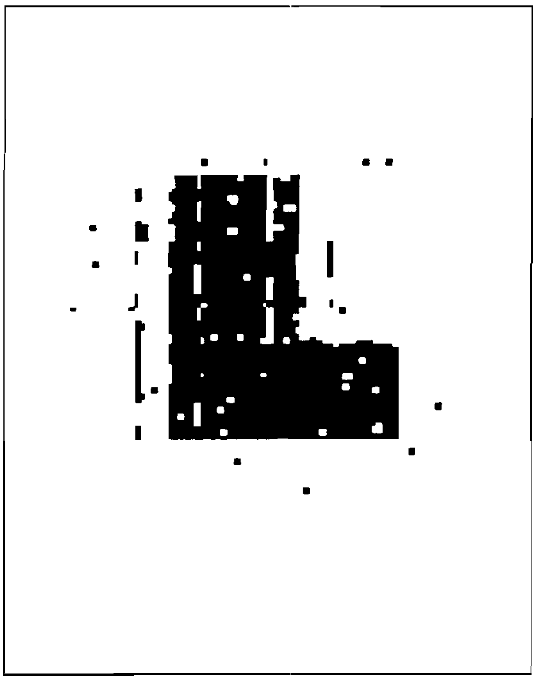


Figure (d)

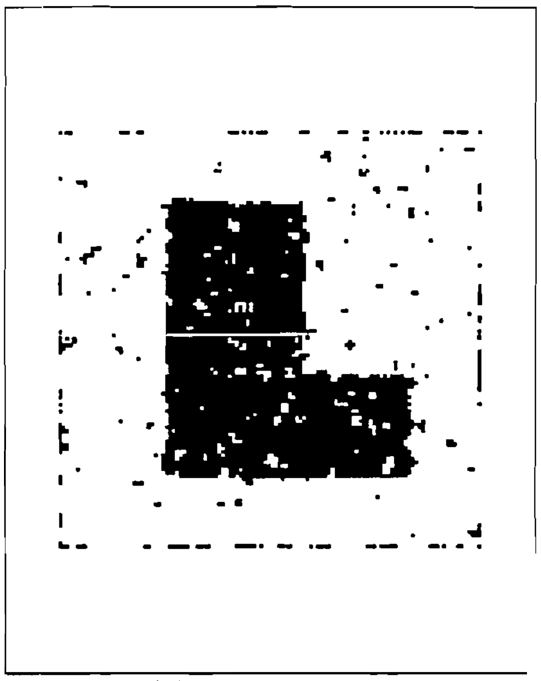


Figure (e)

Case 1;4-pixel neighborhood regularization matrix;λ = 1 r=0.5

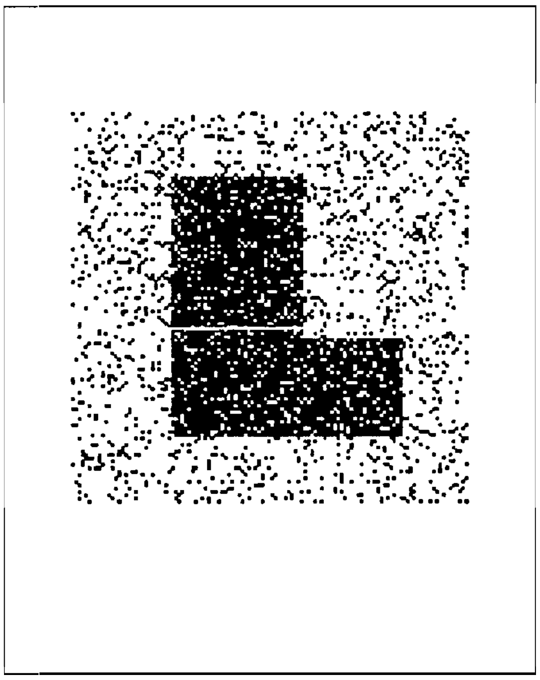
Figure(a): Noisy Image

Figure(b): Single Stage Recovery

Figure(c): Double Stage Recovery

Figure(d): Single followed by Double Stage Recovery

Figure(e): Median filter(3x3 window) recovery


Figure(a)

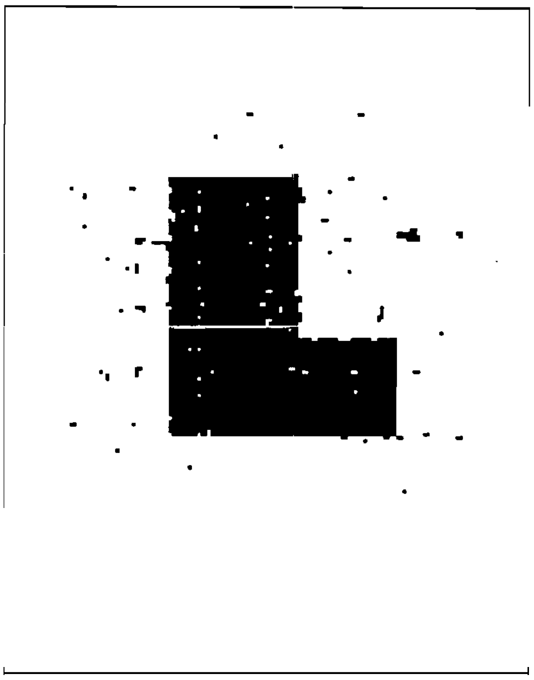


Figure (b)

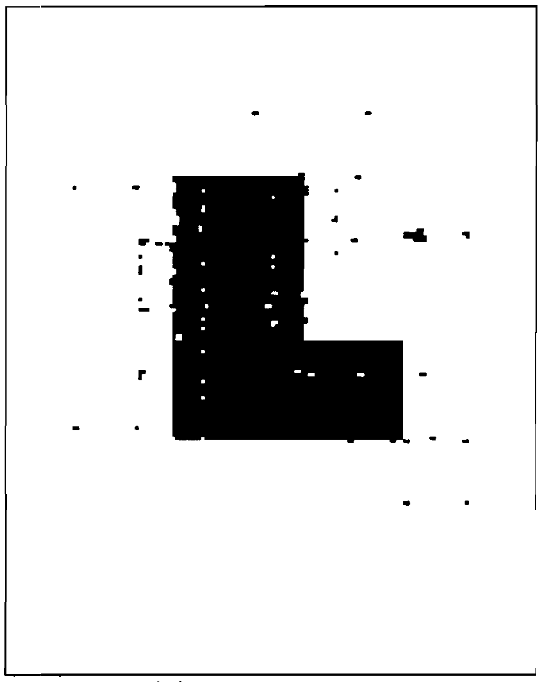


Figure (c)

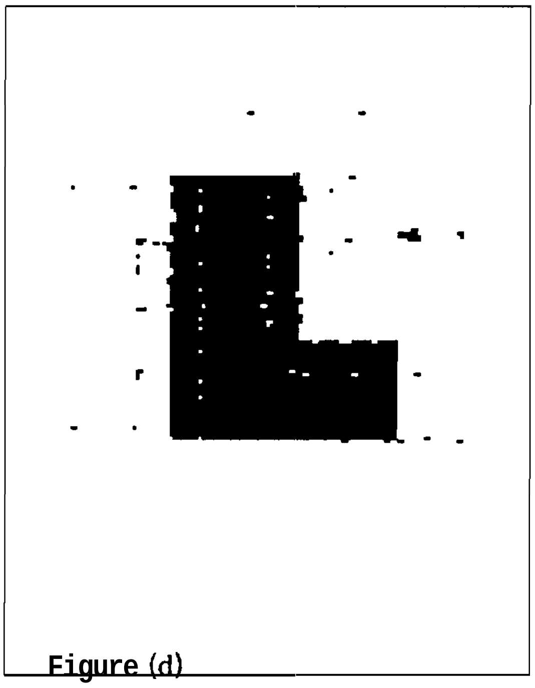


Figure (d)

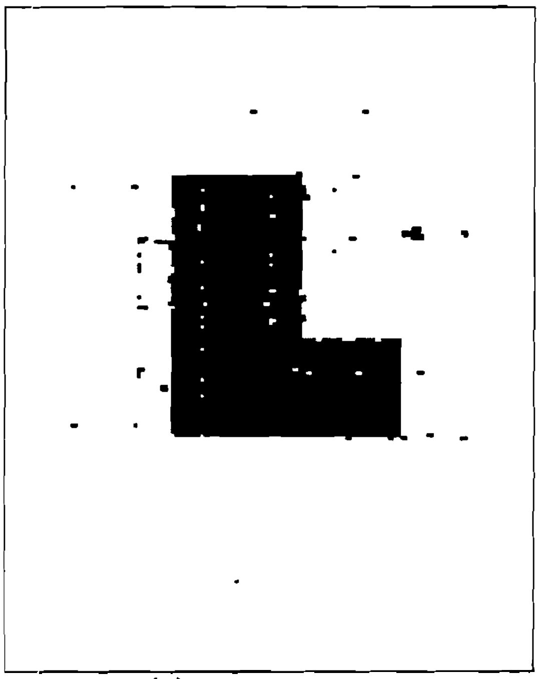


Figure (e)

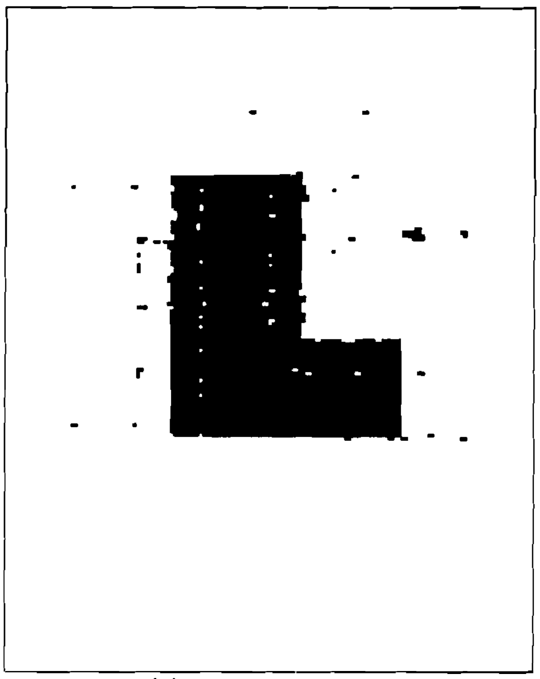


Figure (f)

Case 2:λ = 3;4-pixel neighborhood regularization matrix; r = 0.3

Figure(a): Noisy Image

Figure(b): Single Stage Recovery

Figure(c): Double Stage Recovery

Figure(d): Single followed by Double Stage Recovery

Figure(e): Four Stage Recovery

Figure(f): Single followed by Four Stage Recovery

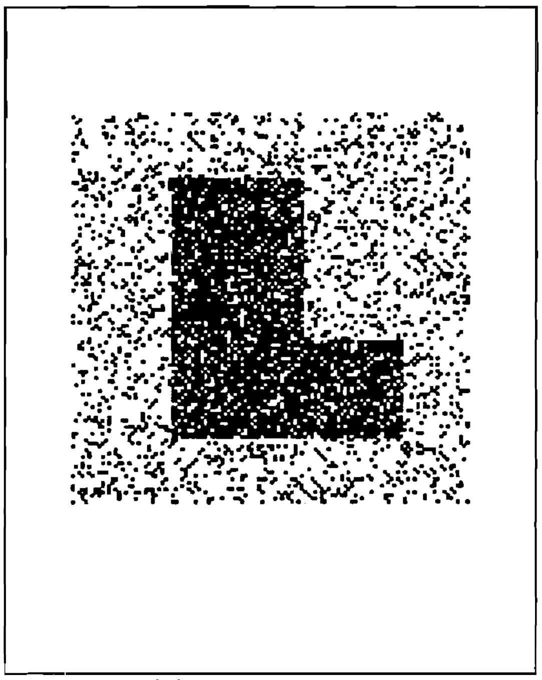


Figure (a)

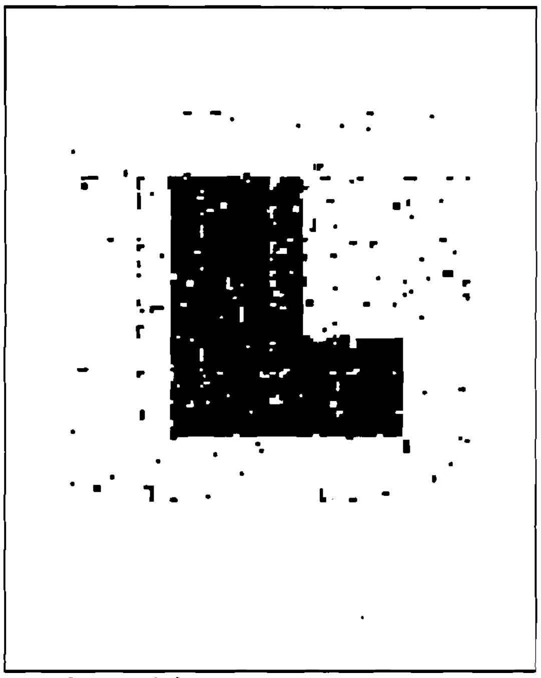


Figure (b)

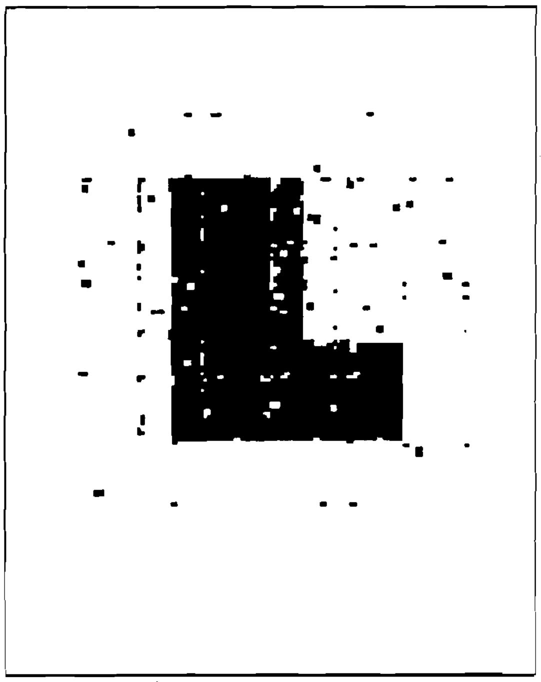


Figure (c)

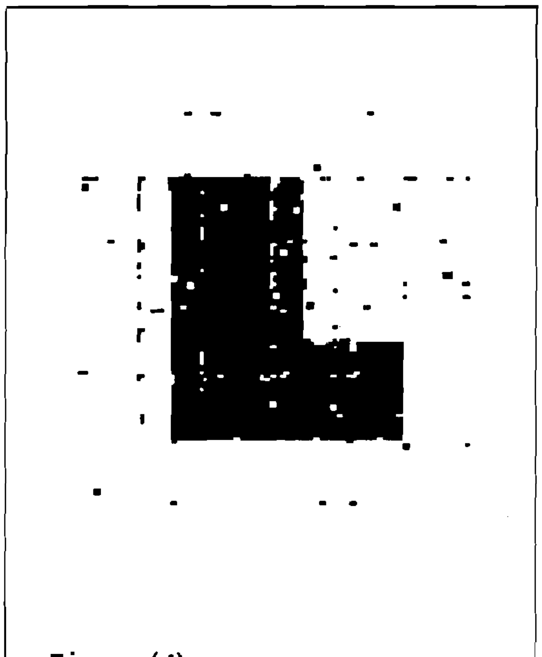


Figure (d)

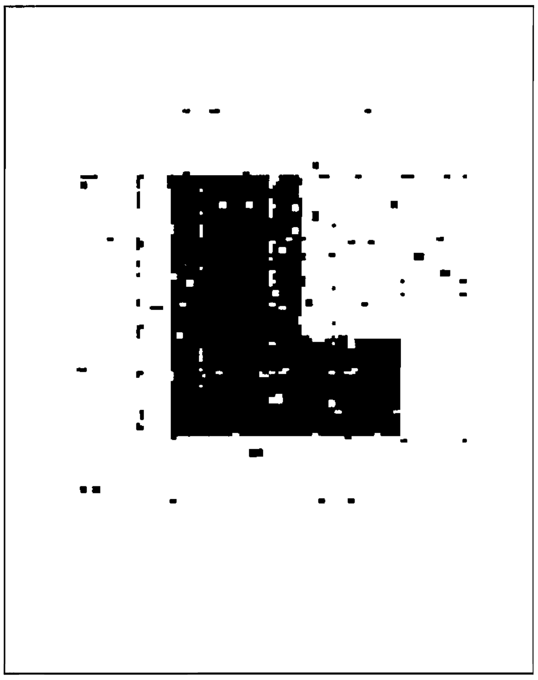


Figure (e)

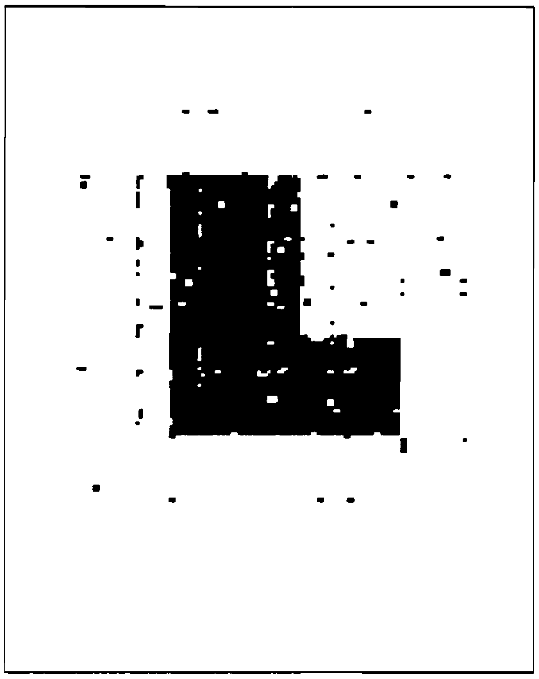


Figure (f)

Case2:λ = 3;4-pixel neighborhood regularization matrix; r = 0.4

Figure(a): Noisy Image

Figure(b): Single Stage Recovery

Figure(c): Double Stage Recovery

Figure(d): Single followed by Double Stage Recovery

Figure(e): Four Stage Recovery

Figure(f): Single followed by Four Stage Recovery

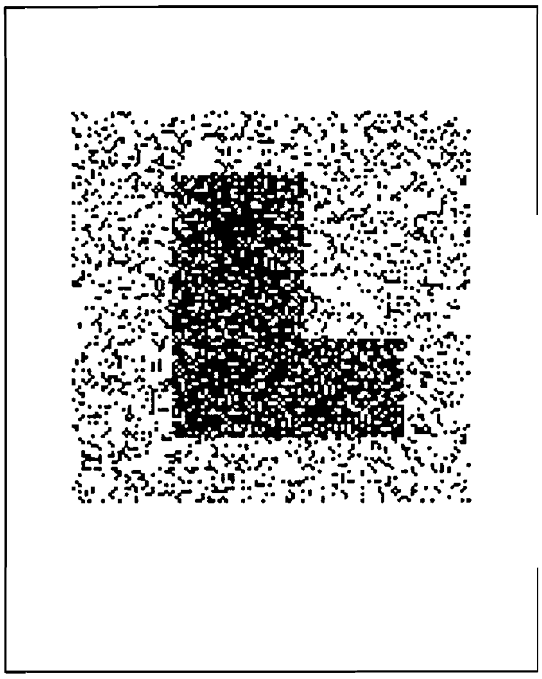


Figure (a)

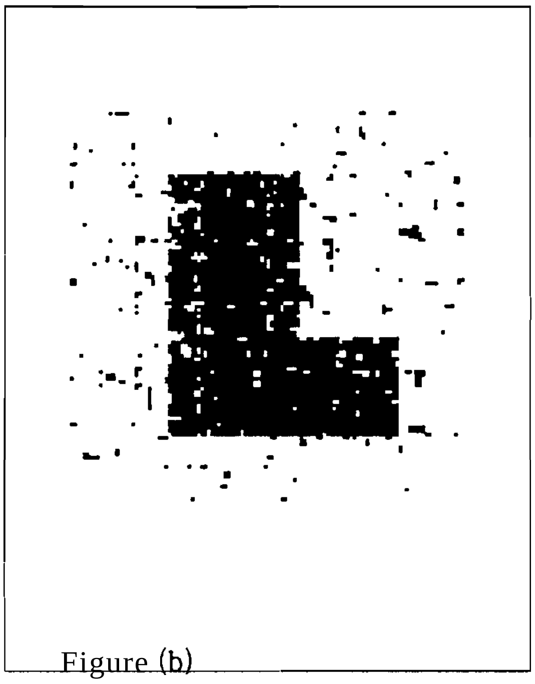


Figure (b)

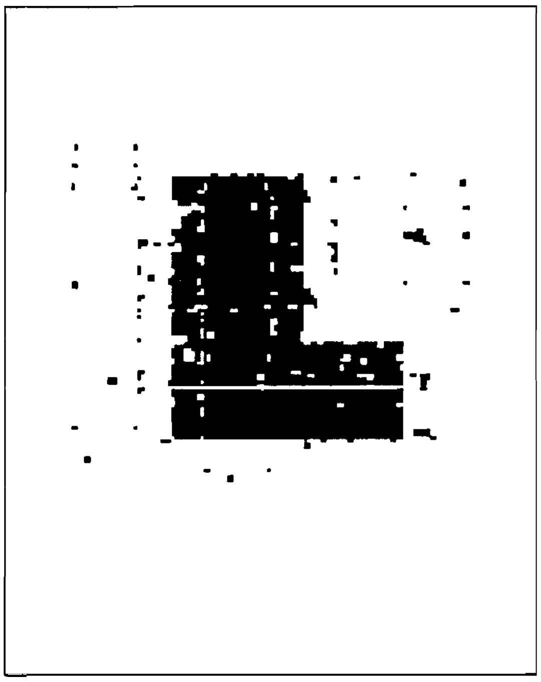


Figure (c)

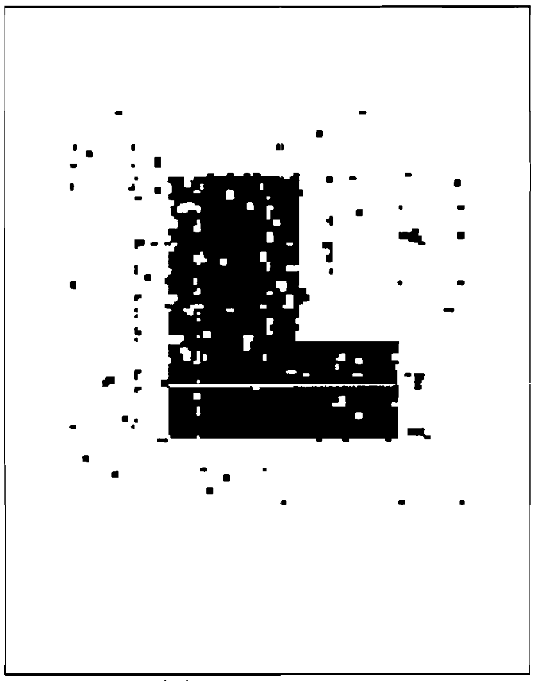


Figure (d)

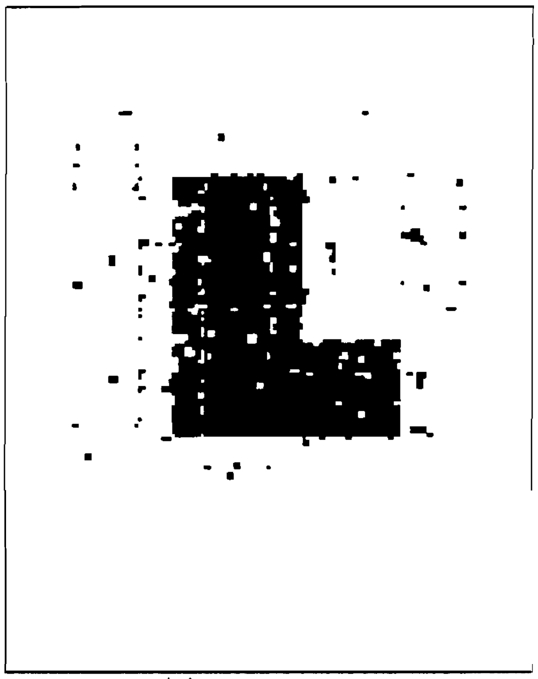


Figure (e)

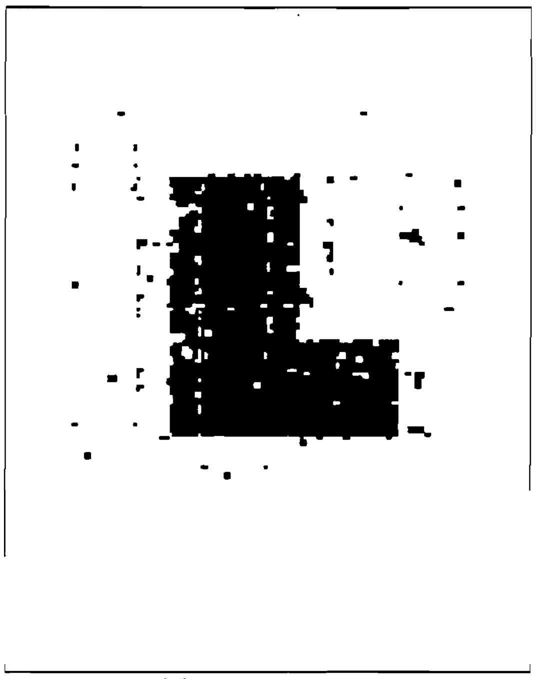


Figure (f)

Case2:λ = 3 ;4-pixel neighborhood regularization matrix; r = 0.5

Figure(a): Noisy Image

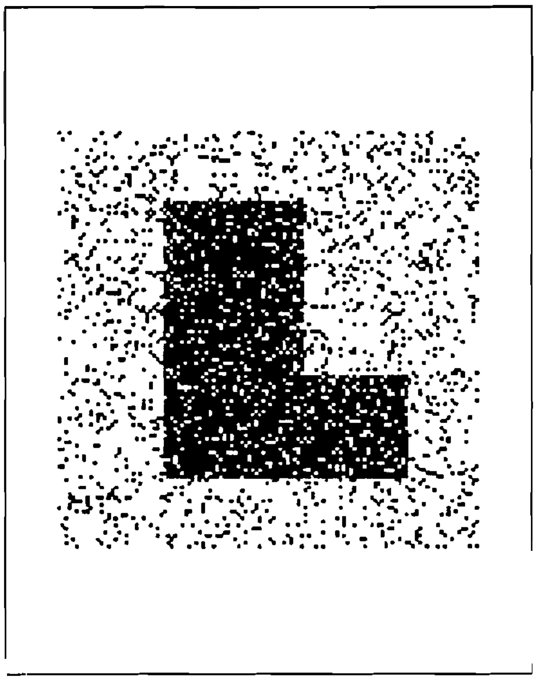
Figure(b): Single Stage Recovery

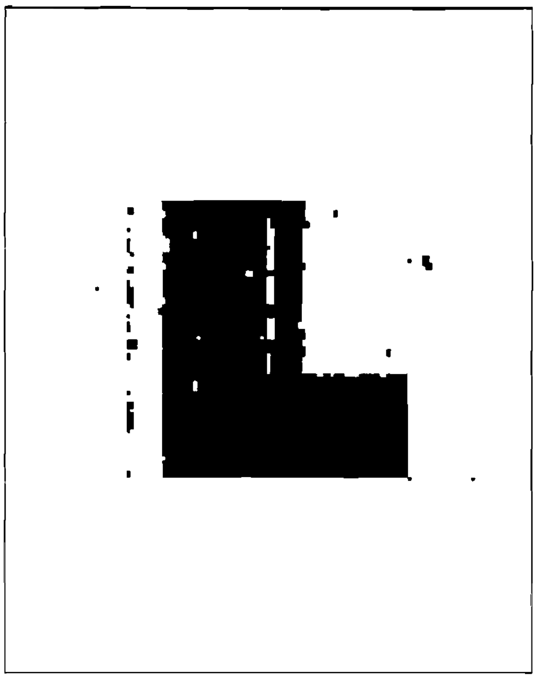
Figure(c): Double Stage Recovery

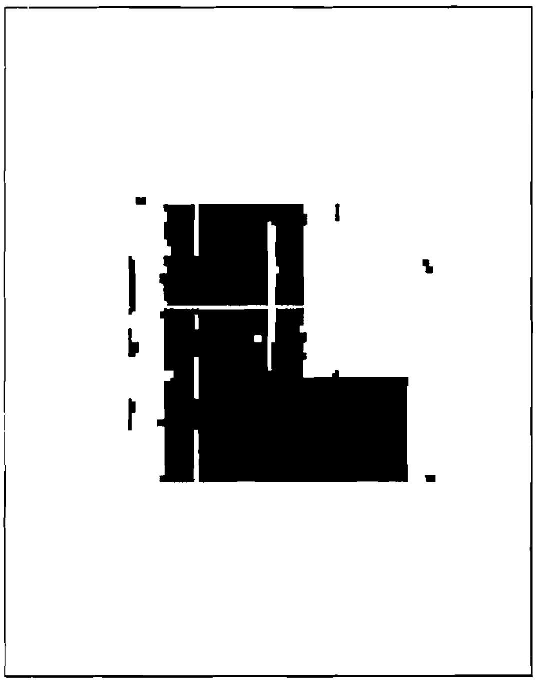
Figure(d): Single followed by Double Stage Recovery

Figure(e): Four Stage Recovery

Figure(f): Single followed by Four Stage Recovery


figure(a)


figure(b)


Figure(c)

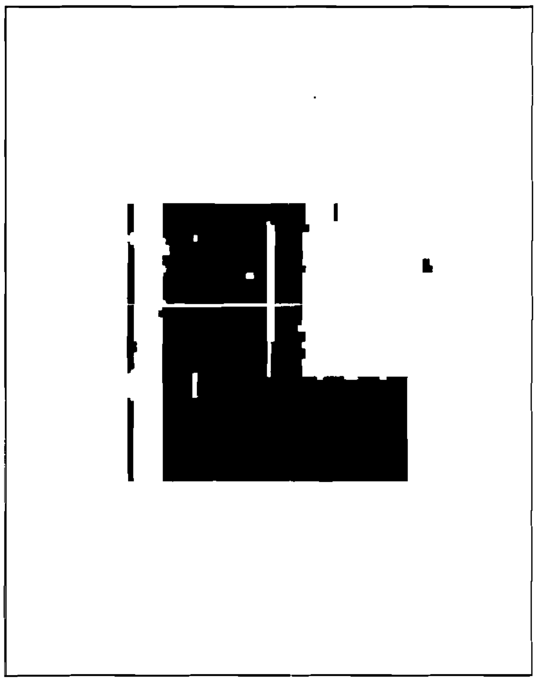


Figure (d)

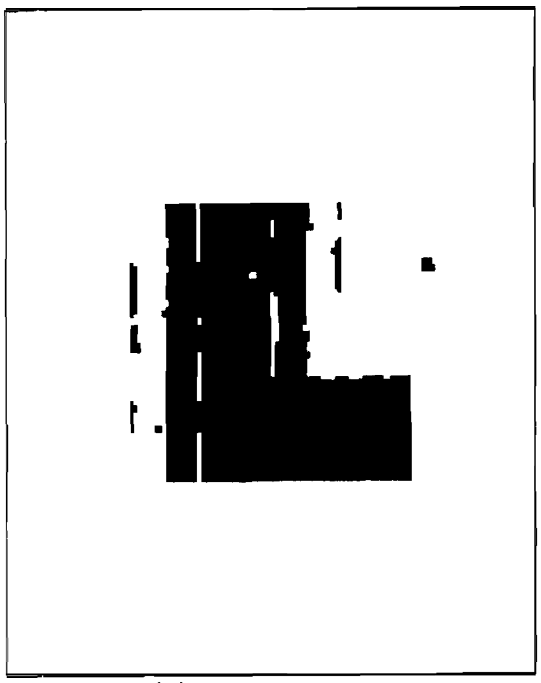


Figure (e)

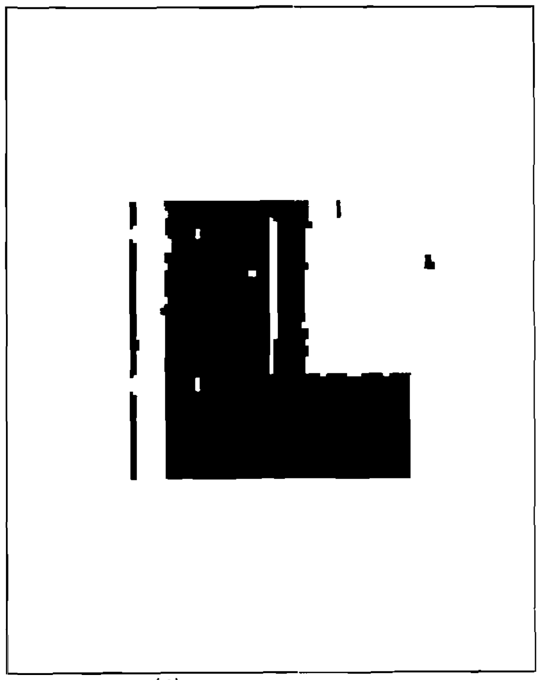


Figure (f)

Case 3: $\lambda=1$;8-pixel neighborhood regularization matrix;r=0.3

Figure(a): Noisy Image

Figure(b): Single Stage Recovery

Figure(c): Double Stage Recovery

Figure(d): Single followed by Double Stage Recovery

Figure(e): Four Stage Recovery

Figure(f): Single followed by Four Stage Recovery

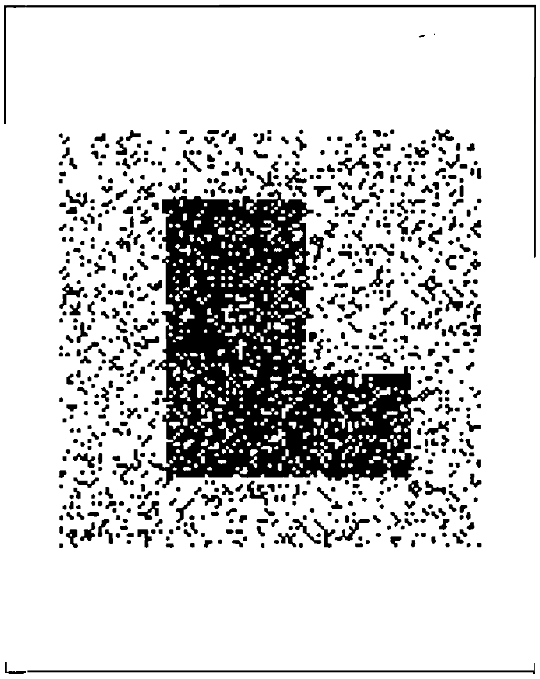


Figure (a)

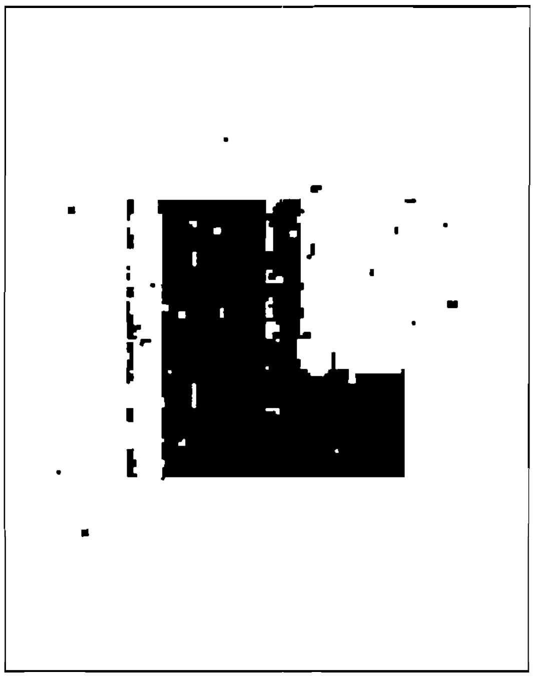


Figure (b)

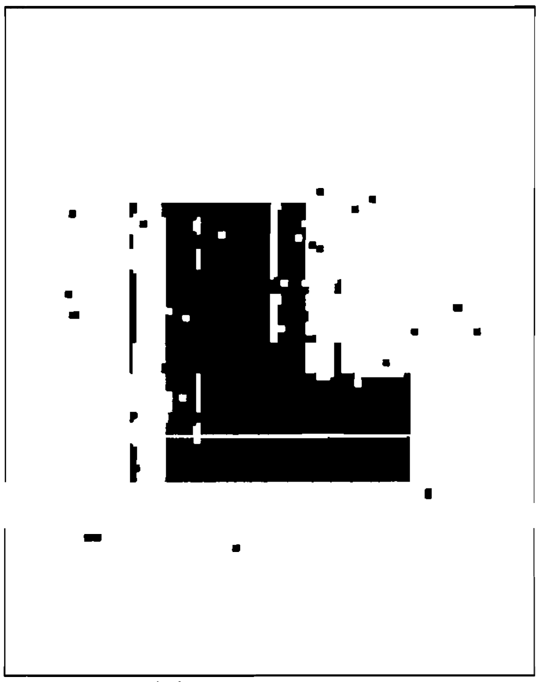


Figure (c)

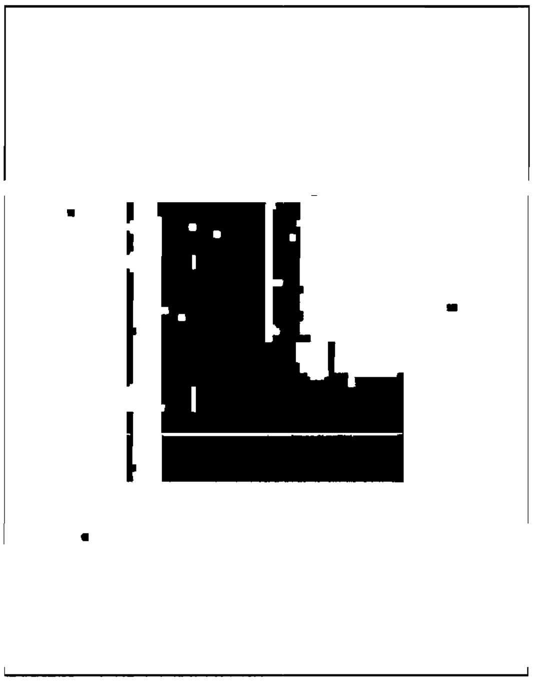


Figure (d)

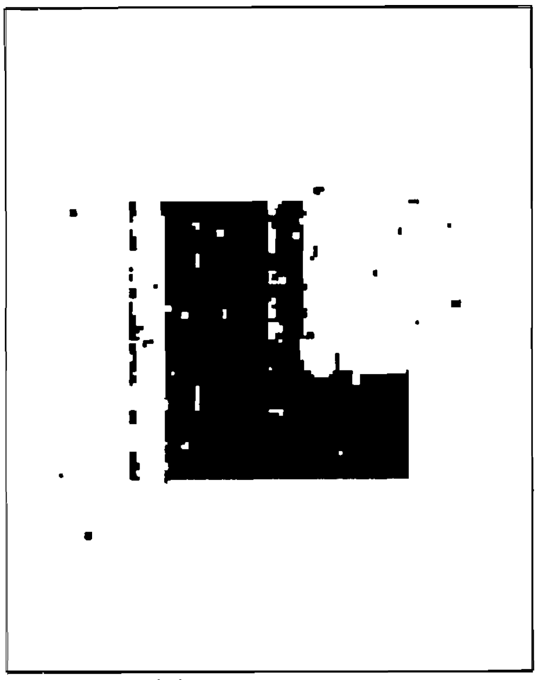


Figure (e)

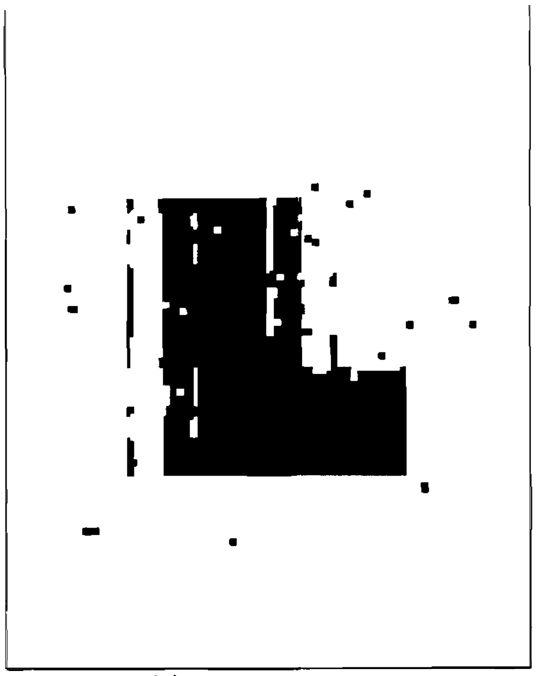


Figure (f)

Case 3: λ=1;8-pixel neighborhood regularization matrix;r=0.4

Figure(a): Noisy Image

Figure(b): Single Stage Recovery

Figure(c): Double Stage Recovery

Figure(d): Single followed by Double Stage Recovery

Figure(e): Four Stage Recovery

Figure(f): Single followed by Four Stage Recovery

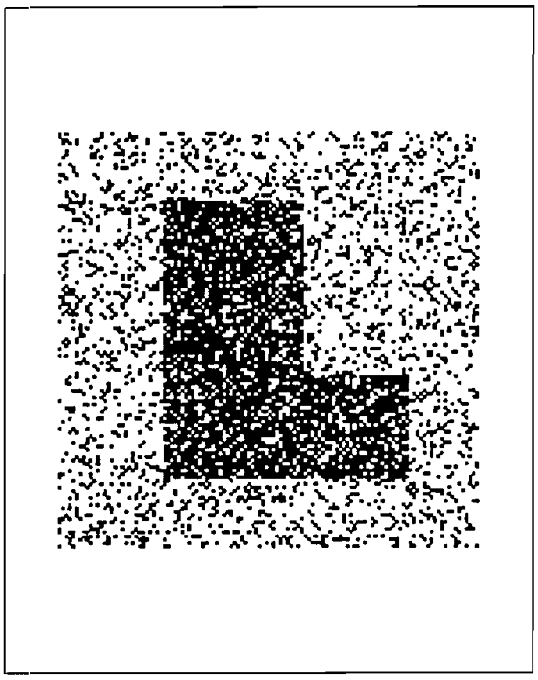
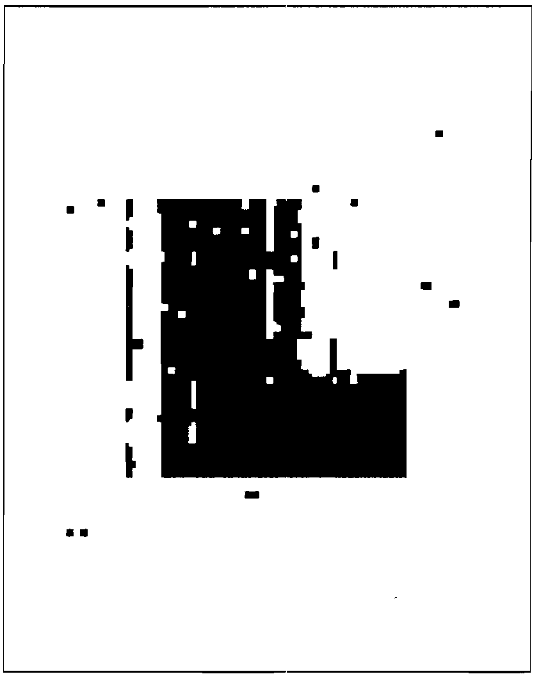


Figure (a)


Figure(b)

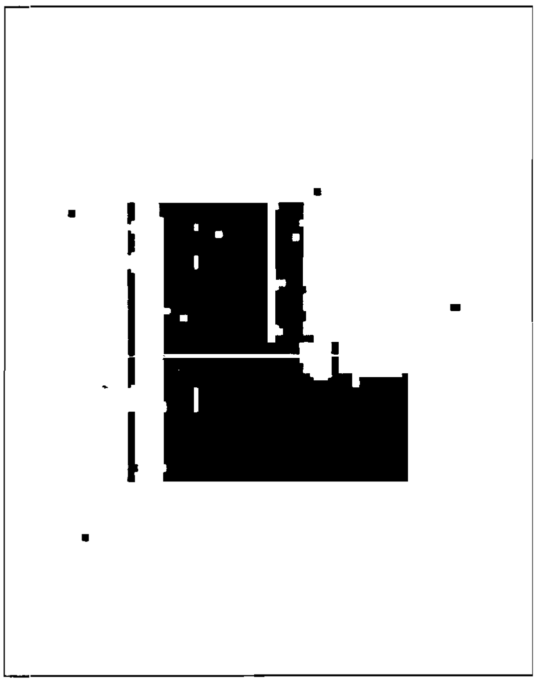


Figure (c)

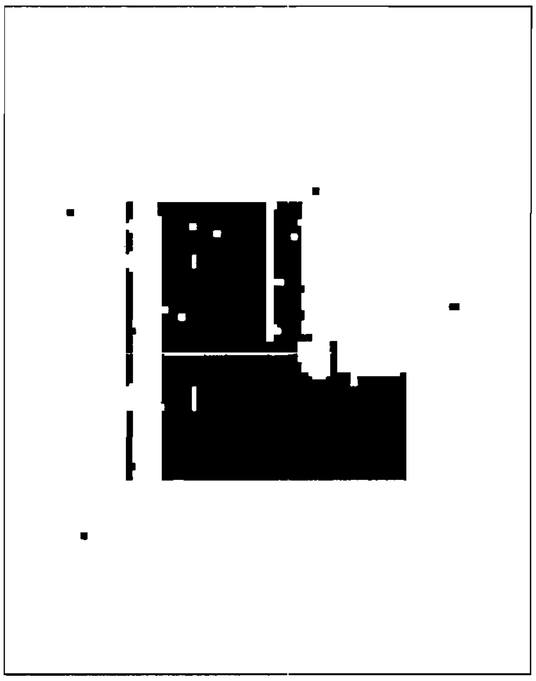


Figure (d)

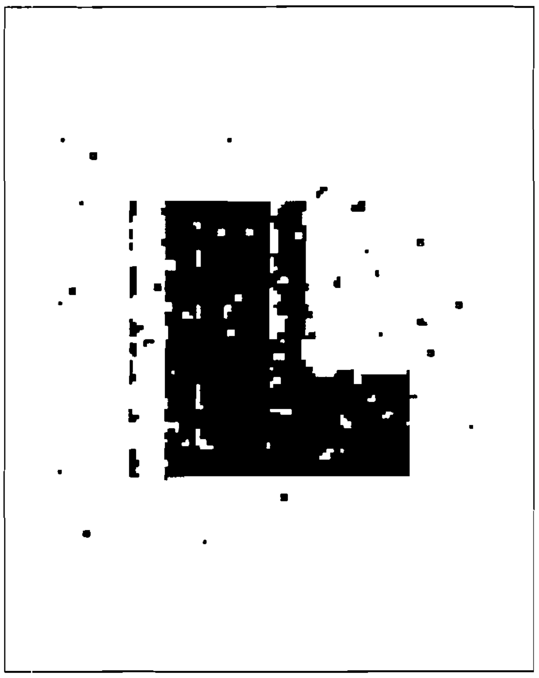


Figure (e)

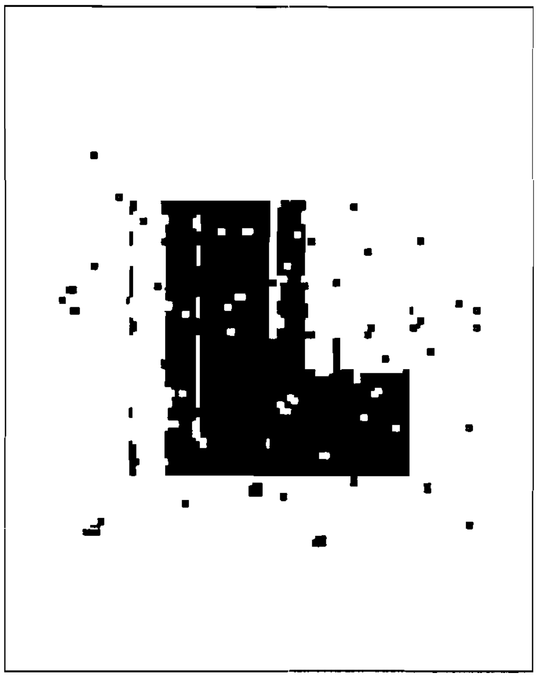


Figure (f)

Case 3: $\lambda=1$;8-pixel neighborhood regularization matrix;r=0.5

Figure(a): Noisy Image

Figure(b): Single Stage Recovery

Figure(c): Double Stage Recovery

Figure(d): Single followed by Double Stage Recovery

Figure(e): Four Stage Recovery

Figure(f): Single followed by Four Stage Recovery

REFERENCES

1.Walter E. Lillo, Stefen Hui, and Stanislaw H. Zak. " Neural Networks for constrained optimization problems." International Journal of Circuit Theory and Applications, Vol.21, pp.385-399, (1993).

2. Osama A. Mohammed, Riaz S. Merchant and Fuat G. Uler. " Utilizing Hopfield Neural Networks and an Improved simulated Annealing Procedure for Design Optimization of Electromagnetic Devices. IEEE gnetics, vol. 29, No.6, November 1993.

3. Wong, K.Y.M and Sherrington, D. "Adaptive Optimization in Neural Networks ". Physica A., Vol.185, pp. 466-470

4. Abe, Shigeo;Kawakami, Junzo and Hirasawa, Kotaroo. " Solving Inequality Constrained Combinatorial Optimization Problems by the Hopfield Neural Networks." Neural Networks, Vol.5, No.4, pp.663-670

5. Janssen, A.J.E.M." An Optimization Problem Related to Neural Networks". Information Processing Letters. Vol.37, No.3, PP.155-157

6. Jayadeva; Bhaumik, B. " Optimization with Neural Networks". Biological Cybernetics, Vol.67, No.5, pp.445-449

7. Takagi, Hideyuki; Sakaue, Shigeo; Togawa, Hayato. " Evaluation of Nonlinear Optimization Methods for the Learning Algorithm of Aritificial Neural Networks.", Vol. 23, No.1, p.101-111

8. Chao, Jinhui; Ratanaswan, Wiak; Tsujii, Shigeo. " New Global Optimization Method and Supervised Learning of Multilayer Neural Networks". Trans. of the IEICE, Vol.73, No.11, pp.1796-1799

9. Narendra, K.S.; Parthasarathy, K. " Gradient Methods for the Optimization of Dynamical Systems containing Neural Networks". IEEE Transactions on Neural Networks. Vol.2, No.2, pp.252-262

10. Maa, C.Y; Shanblatt, M.A. " A Two-Phase Optimization Neural Network". Vol.3, No.6, pp.1003-1009.

11. Hopfield JJ, Tank DW (1985) "'Neural' computation of decisions optimization problems," Biological Cybern, vol. 52, pp. 141-152, 1985.

12. Dahl Ed (1987) "Neural network algorithm for an NP complete problem:map and graph coloring", in: Caudill M, Butler C (eds) Proc IEEE Int. Conference Neural Networks, vol III, p.113-120

13. Athanasios G. Tsirukis, Gintaras V. Reklaitis and Manoel F. Tenorio. " Nonlinear Optimization Using Generalized Hopfield Networks." Neural Computation 1, 511-521 (1989)

14. Jeffrey, W., and Rosner, R, 1986. " Neural Network Processing as a Tool for Function Optimization." in "Neural Networks for Computing", J.S. Denker, ed., pp.241-246. American Institute of Physcis, New York, NY.

15. Kennedy, M.P., and Chua, L.O. "Neural Networks for nonlinear programming". IEEE Trans. Circuits and Systems 35(5), 554-562. (1988)

16. Anil K. Jain. "Fundamentals of Digital Image Processing ". Prentice Hall, Englewood Cliffs, NJ 07632

17. H. Makino and Y. Akada. " On bilevel Rendition for a Document Image Including Half-tone Pictures", IEICE, Vol. J65-D, No.1, pp307-314 (1982).

18. D. Wang, M. Minoh and T. Sakai. " An Area Division Method of Document Images based on the Grey-level Patterns inside a 3x3 Unit Mesh". IIEEJ, vol.13, No.1, pp29-37 (1984)

19. T. Semasa et al., " Bi-level Rendition for Image Containing Text, Screened-Halftone and Continous-c one". IIEEJ, vol.20, No.5, pp476-483 (1991)

20. N. Otsu. " An Automatic Threshold Selection Method Based on Discriminant and Least Squares Criteria, IEICE, Vol.J63-D, No.4, pp349-356 (1980)

21. Naoki Kuwata, Yasuhiko Murayama and Schoichi IIno. " A New Bi-level Quantizing Method For Document Images ". IEEE Transactions on Consumer Electronics, Vol.38, No.3,pp 718-724

22. William K. Pratt. " Digital Image Processing ". eds. John Wiley & sons

23. J.W. Tukey. " Exploratory Data Analysis ", Addison-Wesley, Reading, Mass., 1971

24. John Hertz, Anders Krogh and Richard G. Palmer. " Introduction to the Theory of Neural Computation". eds Addison-Wesley Publishing Company

25. David W. Tank and John J. Hopfield " Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit." IEEE Transactions on circuits and systems, vol. cas-33, No.5, 1986.

26. A. K. Katsaggelos " Digital Image Restoration ". p73-78

27. Yi Sun and Song-Yu Yu. " An Eliminating Highest Error (EHE) Criterion in Hopfield Neural Networks for bilevel image restoration". Pattern Recognition letters 14 (1993) 471-474, North Holland.

28. Htong Zhou, Rama Chellappa, Aseem Vaid and B. Keith Jenkins. " Image Restoration using a Neural Network ". IEEE Transactions on Accoustics, Speech and Signal Processing Vol.36, No.7, July 1988

29. R.P. Lippmann. " An Introduction to computing with Neural Networks ". IEEE ASSP Mag., April 1987

30. P. Carnevali, L. Coletti, S. Patarnello. " Image processing by simulalted annealing". IBM J. Res. Develop. vol. 29 no. 5 , November 1985.

31. K.A. Narayanan, D.O'Learly, and A. Rosenfeld.. " Image Smoothing and Segmentation by Cost Minimization". IEEE Trans. Syst. Man. Cyber. SMC-12, 91 (1982)

32. O.K. Ersoy, J.Y. Zhuang, J. Brede, " Iterative Interlacing Approach to the Synthesis of Computer-Generated Holograms ". Applied Optics, Vol.31, No.32, pp.6894-6901, November 10, 1992

33. Steve G. Romaniuk and Lawrence O. Hall. " Divide and Conquer Neural Networks ". Neural Networks, Vol. 6, pp. 1105-1116, 1993.

34. Sundaram R. and Ersoy O.K. " Partitioning of Hopfield Networks and its Application to Image Restoration ". To appear in Applied Optics

35. B.M. Forrest. " Restoration of Binary images Using Networks of Analogue Neurons." in " Parallel Architectures and Computer Vision". eds. Ian Page