Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

New Jersey's Science & Technology University

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

ABSTRACT

Blur Identification and Restoration of Images of Coronary Microvessel

by

Kuen Tak Wong

The objective of this research was to identify the blur characteristics of the blurred images of the rat coronary microvessel, and the information of the blur characteristics was used to restore the blurred images. The blur characteristics were analyzed by using the image power cepstrum. The Wiener filter was implemented to restore the images. There were two types of point spread functions proposed and studied for the restoration. They were: defocus blur PSF and motion blur PSF. The images were transferred from HP A900 system to an AVS workstation. The images were processed and manipulated by the AVS, and showed significant improvement in quality. Blur characteristics which were similar to the motion blur were found in all the images. The motion blur PSF did not show much effectiveness in the restoration process. No defocus blur or motion blur characteristics appeared on the cepstrum of the microvessel images, suggesting that the strobe technique was capable of acquiring stationary coronary microvessel images.

BLUR IDENTIFICATION AND RESTORATION OF IMAGES OF CORONARY MICROVESSEL

by
Kuen Tak Wong

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Biomedical Engineering

Biomedical Engineering Committee

May 1993

APPROVAL PAGE

Blur Identification and Restoration of Images of Coronary Microvessel

Kuen Tak Wong

57-/93

Dr. Arthur B. Ritter, Thesis Advisor date
Professor of Physiology, Division of Microcirculation Research,
University of Medicine and Dentistry of New Jersey

5/11/93

Dr. Yun-Qing Shi, Thesis Advisor date
Assistant Professor of Electrical Engineering, NJIT

5ˉ/11/93

Dr. David Kristol, Committee Member date
Professor of Chemistry, Director and Graduate Advisor of Biomedical
Engineering and Director of the Center for Biomedical Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Kuen Tak Wong

Degree: Master of Science in Biomedical Engineering

Date: May 1993

Date of Birth:

Place of Birth:

Graduate and Undergraduate Education:

- Master of Science in Biomedical Engineering
  New Jersey Institute of Technology, Newark, NJ, 1993
- Bachelor of Science in Electrical and Computer Engineering
  State University of New York at Buffalo, Buffalo, NY, 1991

Major: Biomedical Engineering

This thesis is dedicated to my dear parents,
and to Wai-lin Cheung and her family

ACKNOWLEDGMENT

The author wishes to express his sincere gratitude to his thesis advisors, Dr. Arthur B. Ritter and Dr. Yun Q. Shi, for their guidance, friendship, and moral support throughout this research. I would like to express my appreciation to Dr. Ritter for his patience and sparing his precious time to help me to acquire the research data. I would like to express my appreciation to Dr. Yun Q. Shi for his encouragement and providing me valuable information and advice.

I would like to express my special thanks to Dr. David Kristol for his guidance in my graduate program and serving as a member of the committee.

The author appreciates the timely help from Feng You and the computer consultants at UMDNJ who provided me important computer system information .

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

  1.1 Research Background 1

  1.2 Objective 2

2 MICROSCOPE IMAGING SYSTEM MODELING 4

  2.1 System Configuration 4

  2.2 System Modeling 7

3 BLUR IDENTIFICATION 10

  3.1 Introduction to Blur Identification 10

  3.2 Blur Identification Method 11

  3.3 Blur Modeling 12

    3.3.1 Motion Blur 13

    3.3.2 Defocus Blur 15

  3.4 Choice of PSF Models 16

  3.5 Power Cepstrum Blur Identification 17

4 IMAGE RESTORATION 21

  4.1 Introduction to Image Restoration 21

  4.2 Wiener Filter 23

5 RESULTS AND DISCUSSION 26

  5.1 Testing of Blur Identification and Restoration Algorithm 26

  5.2 Blur Identification and Restoration of Microscope Images 26

  5.3 Blur Identification and Restoration of Coronary Microvessel Images 28

6 CONCLUSION 42

APPENDIX PROGRAM LISTING 44

REFERENCES 62

LIST OF FIGURES

Figure **Page**

2.1 The image acquisition system schematic diagram 6

2.2 Linear image acquisition system model 8

3.1 Blur identification algorithm 20

4.1 Wiener filter algorithm 25

5.1 Blur identification and restoration of test images 30

5.2 Restored test images 31

5.3 Micrometer images 32

5.4 Enhanced version of restored the micrometer image 33

5.5 Micrometer images restored with different R value 33

5.6 Micrometer images restored with different K value 34

5.7 Microvessel image displayed on TV monitor 35

5.8 Microvessel image displayed on AVS workstation monitor 36

5.9 Blur identification and restoration of the microvessel image...........37

5.10 Microvessel images with lower quality 38

5.11 Enhanced microvessel images 39

5.12 Blur identification of the microvessel images with lower quality 40

5.13 Blur identification of the microvessel images with lower quality (continuous) 41

CHAPTER 1

INTRODUCTION

1.1 Research Background

A microscope image acquisition system has been developed for direct visualization of surface vessels of coronary microcirculation of rat [1,2,4]. The aim for developing this system is to study the coronary microcirculation, which is an essential component in the understanding of the transportation of macromolecules such as proteins between blood and heart tissue [1-4]. A beating rat heart was used in these studies because rats maintain normal blood pressure under anesthesia and in the open chest condition. The rats which are subjected to these experiments were anesthetized, trachesostomized, ventilated with a small animal respirator, and sternotomized to exposed the heart which is isolated from the lung by placing it in a pericardial cradle. The coronary surface vessels are exposed by removing the pericardial sac. Fluorescein isothiocyanate (FITC) labeled dextrans are introduced into the femoral vein as a tracer for macromolecules. A detailed description of the experiments are contained in reference [1-8]. The macromolecules in the coronary microvessel are tagged with the fluorochrom FITC, observed through intravital microscopy, and the experiments are recorded on 3/4 inch analog video tape by the image acquisition system.

The imaging system uses a computer controlled EKG triggered strobe which drives a frame grabber digitizer to arrest the image of the beating rat heart once each heart beat. The stroboscopic technique allows observing only a small region of the cardiac cycle at the same interval of each beat, making the observation appear to be more stationary in order to visualize the surface microvessel of the epimyocardium. A video camera is adapted to the microscope to monitor the whole experimental process through a monitor, and records it on video tape. The digitized data is transferred to a minicomputer system for storage, and subjected to further process and analysis by a software package installed on the minicomputer.

1.2 Objective

The software package of the system, constructed for direct visualization of the surface coronary microvessel, provides substantial different image processing utilities. It can interactively select different portions and sizes of image from the image data displayed on the monitor. The image can be compressed or uncompressed to save storage space, processed by different image enhancement techniques such as filtering and edge detection, and transferred back to the monitor to analyze the result. However, the image enhancement techniques are inadequate to recover the blurred image data obtained from the experiments [3]. Therefore, image restoration is proposed to deblur the image in order to obtain a more focused image and more precise information [5,6].

In previous works [5,6], different image restoration techniques were applied and the nature of the blurring process is suggested, with significant improvement in image restoration. In Wang's paper [6], motion blurring on the image is assumed to be due to the heart motion. This assumption is valid because the frame grabber cannot freeze the moving heart completely to obtain the image. However, the motion direction and blur extent were entered into the restoration filter by trial and error methods to search for an appropriate result. This method is very time consuming and tedious, and the result may not be the optimized one.

Therefore, a more practical approach is appropriated to recover the blurred image in a more accurate and reliable way.

In this thesis, the blur nature of the image data is first identified before the restoration is performed. There is no constraint on the beating heart when the image data is acquired so that the physiology of the coronary microcirculation is not impaired. This means that not only their blurring can be due to motion but also the defocus blurring can occur because the microvessels are not in the focal plane of the microscope. Therefore, two types of blur models are proposed: defocus blur and motion blur. The blur parameters are extracted by the blur identification algorithm, and those parameters will be incorporated into a Wiener Filter to restore the image. The image data is transferred from the local computer network in the laboratory to the academic computer system at University of Medicine and Dentistry of New Jersey (UMDNJ). The data is converted into a format that is compatible with the software called Application Visualization System (AVS) [11]. The purpose of using AVS is that its computing platform (HP workstation) has a high resolution monitor and AVS can use these capabilities for sophisticated image processing and visualizing techniques. The image is first subjected to blur identification analysis, and the blur information is then used in image restoration. The blur characteristics analysis and further enhancement of the restored image are all manipulated by the AVS software.

CHAPTER 2

MICROSCOPE IMAGING SYSTEM MODELING

2.1 System Configuration

A microscope imaging system has been developed to study the coronary microcirculation [8]. The system can record images observed through the microscope, on video tape, digitize it, and process or enhance the image by the computer. The image acquisition can also be controlled by a computer triggered strobe. The system configuration is as follows [8]:

1. Nikon OPTIPHOT microscope with both transillumination and equilluminantion.
2. Model COHU 4410 S17 video camera equipped with separate-mesh vidicon with silicon intensifier target (SIT).
3. Quantex DS20F digital video processor.
4. HP2240A measurement and control processor for digital and analog input and output.
5. Panasonic CT-2010M video monitor.
6. Sony 5600 video recorder.
7. EKG signal amplifier.
8. HP1000 RTE-A900 processor.

The system diagram is shown in figure 2.1.

The image obtained through the microscope is scanned by the vidicon which produces an analog electronic image signal. The image signal is recorded on 3/4 inch VHS video tape. The scene either from the microscope or the tape can be digitized into video frames by the digitizer, with the digitizing time of one frame corresponding to standard video rates of 33 msec. The size and region of interest on each video frame can be selected by the user and transmitted through an HP interface bus (IEEE 488) to HP1000 series A-900 minicomputer for further image processing. The stored image is compressed to save storage space and is represented by 8 bit per pixel (256 grey level). There is a software package installed on the minicomputer system, which provides different image processing and enhancement techniques. The processed image can be sent back and viewed on the monitor.

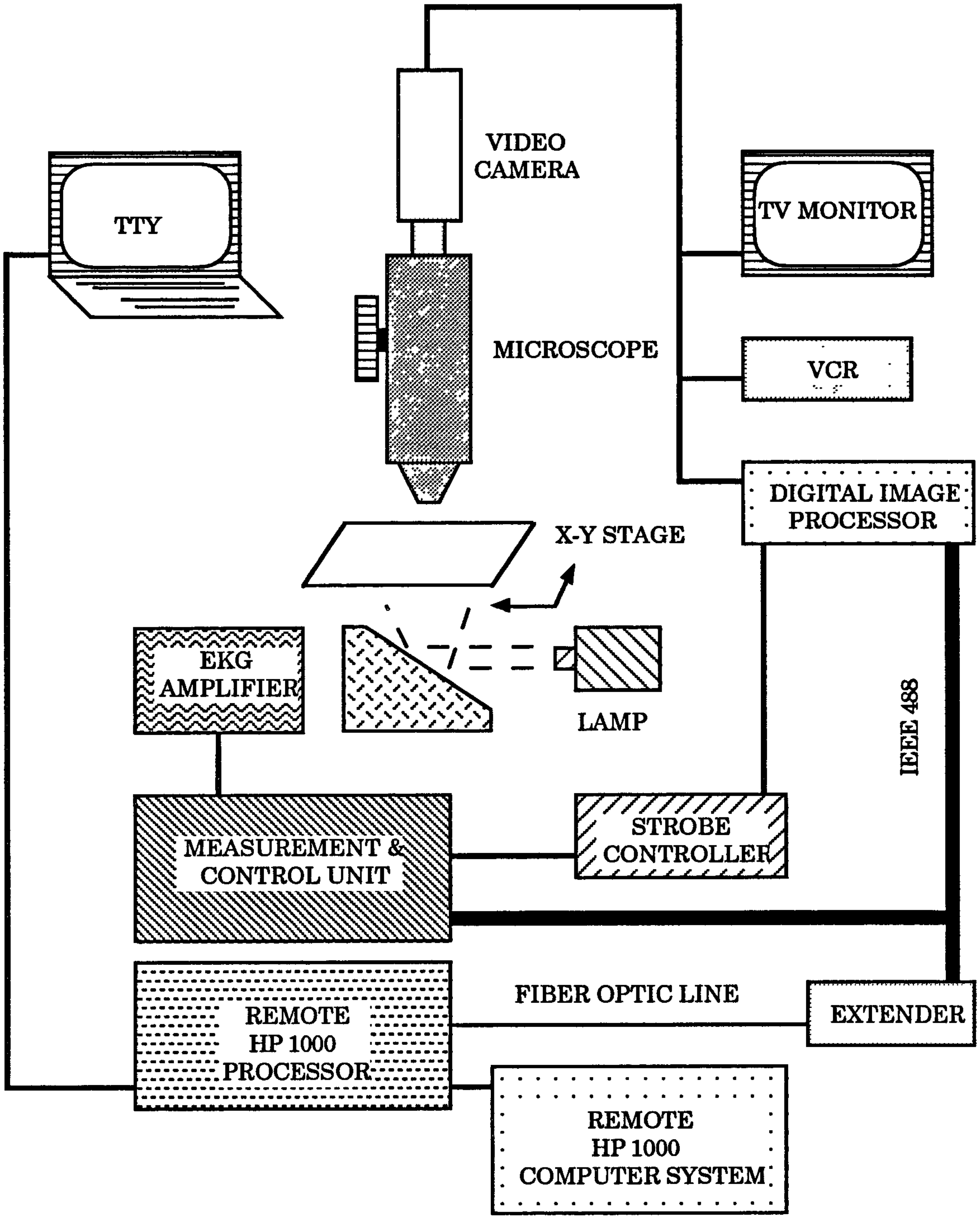

Figure 2.1 The image acquisition system schematic diagram.

2.2 System Modeling

The modeling of the system can be very complicated since many components are involved. Some of the components perhaps behave nonlinearly, or some may be linear. Some of them perhaps have space-variant characteristics or some may be space-invariant. Furthermore, the noise introduced by the components can be additive or multiplication or both. A detailed modeling of each component individually is extremely tedious and complicated since the nonlinear and space-variant problems are difficult to solve. Obviously this kind of approach does not yield a simple solution to the problem.

A different approach, which is usually applied in real practice, is to assume the components are linear, or almost linear. Therefore, by the additivity property of linear systems, all the transfer functions of different components can be lumped together into a single degradation model. When the image acquisition is modeled as a linear system, it benefits from the well established linear system mathematical techniques, and hence can be solved in a more simple way than with a nonlinear systems approach [9].

The noise can be approximated by a zero-mean, white Gaussian random field that is additive and signal independent [10]. There are two main sources of noise in a digital imaging system. They are sensor noise and quantization noise. Although the sensor noise is usually signal dependent and possibly multiplicative, the assumption of white Gaussian noise is still practical because almost all restoration algorithms follow this noise assumption [10]. In fact, a more sophisticated noise model does not lead to significant improvement in restoration.

The image acquisition system can be described as

$$g(x,y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(\alpha,\beta)h(x,y,\alpha,\beta)d\alpha d\beta + n(x,y,\alpha,\beta), \tag{2.1}$$

where g(x,y) is the observed image; f(α,β) is the original scene; h(x,y,α,β) is the impulse response of the blurring system, which is called the point spread function (PSF), and n(x,y,α,β) is the additive noise.

The system equation with noise ignored is graphically depicted below.

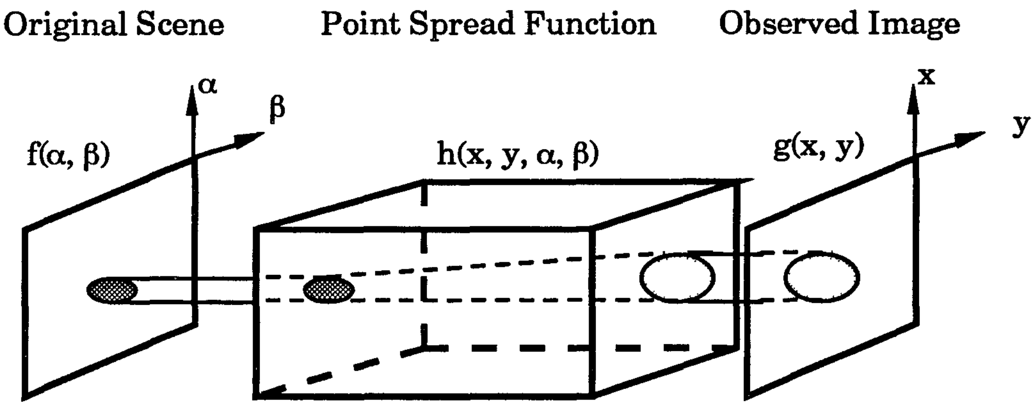

Figure 2.2 Linear image acquisition system model.

When the PSF changes only position but maintains the same function form as the point source explores the object plane, the imaging system becomes space-invariant, and the PSF becomes

$$h(x,y,\alpha,\beta) = h(x-\alpha,y-\beta). \tag{2.2}$$

Hence,

$$g(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha d\beta + n(x,y,\alpha,\beta)$$

$$= f(x,y) \otimes h(x,y) + n(x,y), \qquad (2.3)$$

where $\otimes$ is the convolution operator. As a result, the blurred image g(x,y) can be written as the convolution of the original image f(x,y) and the PSF h(x,y) plus the observed noise. The original image is degraded by the PSF and the observed noise.

In Fourier transform domain, equation (2.3) takes the form

$$\begin{aligned} \mathcal{F}\{g(x,y)\} &= \mathcal{F}\{f(x,y) \otimes h(x,y) + n(x,y)\} \\ &= \mathcal{F}\{f(x,y)\}\mathcal{F}\{h(x,y)\} + \mathcal{F}\{n(x,y)\} \\ \Rightarrow \quad G(u,v) &= F(u,v)H(u,v) + N(u,v), \qquad (2.4) \end{aligned}$$

where $\mathcal{F}\{\bullet\}$ is the Fourier transform, G(u,v), F(u,v), H(u,v) and N(u,v) are the Fourier transform of the observed image, original image, PSF and noise.

CHAPTER 3

BLUR IDENTIFICATION

3.1 Introduction to Bur Identification

In the last chapter, we described the image acquisition system as linear space-invariant image system. The original scene is degraded by the point spread function (PSF) h(x,y) and additive noise n(x,y). The image can be restored when the information about the PSF and noise are known. However, the noise affects only the details of an image and usually is not very severe enough to overwhelm the original scene, but the presence of the PSF can greatly degenerate the intelligibility of an image, even though the PSF is insignificant because it convolves over the whole image. Therefore the availability of known or estimated information about the PSF is crucial to the success of the degraded image recovery.

In general, blur identification refers to the identification of the PSF h(x,y). In some situations, the source or reason of blurring can be known exactly. For instance, when a picture of a moving object is taken, a blur picture will be obtained if the shutter speed is not fast enough to freeze the moving object. Since we know the object was moving while the picture was being taken, we can deduce that the picture is degraded by motion blur. This *a priori* information of the blur nature can assist in modeling the input/output relation of imaging system or the PSF, and supporting restoration algorithm development. However, in practice, the nature of blur is always either unknown or very complicated. In addition, even if the nature of the blur can mostly be realized *a priori*, the parameters of the degradation must be determined and measured for PSF modeling. Therefore blur identification is an essential step in solving image restoration problems.

3.2 Blur Identification Method

The nature and extent of blurring in an image is determined by the PSF. An impluse response function of a linear system reflects the output characteristics from an impluse input [12,13]. Since the PSF is the impluse response of an imaging system, it represents the output spreading pattern from an ideal point source. Therefore, if an ideal point source is present on the original scene, the spread of that ideal point source on the observed image can be used to determine the PSF [13]. Furthermore , an ideal line source can be also used to determine the PSF [13]. In fact, some computer searching algorithm have been developed to estimate the PSF from specific features, such as an edge or a point, in the original image [14]. However, the original scene, in general, may not contain any ideal point or line source. Even though a point source is present, the point spread measurement on the observed image is obscured by observation noise. And in severely blurred images, the images are almost unintelligible and that makes the measurement of the PSF almost impossible. Therefore, these methods are of limited use in practice.

Another technique is to try to identify the PSF parameters in the frequency domain [15-19]. Certain types of PSF can be characterized by the location of zero crossings in the Fourier transform or power cepstrum (Fourier transform of the logarithm of the power spectrum) of the observed image. This method is sensitive to the presence of noise which can make the zero crossings disappear. And this method cannot be applied to the PSFs (such as Gaussian PSF modeling of atmospheric turbulence) that do not contain zero crossings in the frequency domain.

There is a new approach which uses the bispectrum of the observed image to identify the frequency domain zero crossings of the PSF [10]. The bispectral technique is more robust because the observation noise does not affect the bispectrum computation. The bispectrum technique is still under research and it requires much more data than the method based on power spectrum. There is another approach which incorporates the identification procedure into the restoration algorithm [10]. This method expresses the observed image as an autoregressive moving average (ARMA) field. The image and blur are identified when the ARMA parameters are identified. This technique requires statistical modeling and is extremely computationally intense.

In this paper, power cepstrum method is employed for blur identification because this technique has been successfully applied in many cases [15-17]. This method is quite simple and can provide adequate information for our PSFs modeling. A detail description of this method will be presented in a later section of this chapter.

3.3 Blur Modeling

The blur considered here is space invariant and isoplanatic, which means each portion of the image is affected by the same blurring equally. There are three major blurs that are always encountered in real practice, they are: defocus blur, motion blur and atmospheric turbulence. Atmospheric turbulence usually affect remote imaging, and it can be excluded in microscope imaging since the distance between the observing subject and microscope lens is very short, and the environment is indoors. Motion and defocus blur can possibly occur in our acquired image because the heart is moving and cannot be properly focused. Before we discuss the models, we should notice that

$$\sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} h(x,y) = 1 \tag{3.1}$$

to preserve the mean value of the blurred image from the original image.

3.3.1 Motion Blur

Motion blur can be caused either by motion of the object being imaged, or motion of the camera or both. However in our model both cannot occur because the blur becomes space variant when both happen. In microscope imaging, the microscope and video camera stand still, therefore the image obtained does satisfy the assumption above. The output spreading pattern of motion blur PSF from an ideal point source is a short line (elongated point) with a blur length of d. The blur length, d, is proportional to the relative velocity between the camera and the object and to the film exposure time. Therefore for horizontal motion blur [18],

$$h_m(x,y) = \begin{cases} \frac{1}{d} & -\frac{d}{2} \leq x \leq \frac{d}{2}\,;\, y = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

The Fourier Transform is

$$\begin{aligned} H_m(u,v) &= \mathcal{F}\{h_m(x,y)\} \\ &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} h_m(x,y) \exp[-j2\pi\,(ux + vy)]\, dxdy \\ &= \int_{0}^{0} \int_{-d/2}^{d/2} \frac{1}{d} \exp[-j2\pi\,(ux + vy)]\, dxdy \\ &= \int_{-d/2}^{d/2} \frac{1}{d} \exp[-j2\pi ux]\, dx \\ &= \frac{-1}{j2\pi du} \left[e^{-j2\pi ux}\right]_{-d/2}^{d/2} \\ &= \frac{-1}{j2\pi du} \left[e^{-j2\pi u(\frac{d}{2})} - e^{-j2\pi u(-\frac{d}{2})}\right] \end{aligned}$$

$$= \frac{1}{j2\pi du}\left[e^{j\pi du} - e^{-j\pi du}\right]$$

$$= \frac{\sin(\pi du)}{\pi du}$$

$$= \mathrm{sinc}(\pi du). \tag{3.3}$$

The amplitude of $H_m(u,v)$ is characterized by periodic zeros on the u axis, which occur at

$$u = \pm\frac{1}{d}, \frac{2}{d}, \frac{3}{d}, \dots . \tag{3.4}$$

When the motion is at angle θ with horizontal axis, then the PSF is obtained by rotating $h_m(x,y)$ with angle θ above the origin. Hence,

$$h_m(x,y) = \begin{cases} \frac{1}{d} & -\frac{d}{2} \le w \le \frac{d}{2} \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

Where

$$w = x\cos\theta + y\sin\theta.$$

The Fourier Transform of $h_m(x,y)$ is

$$H_m(u,v) = \frac{\sin(\pi d\omega)}{\pi d\omega}, \tag{3.6}$$

where

$$\omega = u\cos\theta + v\sin\theta.$$

The zeros appears along the axis of ω, the direction of motion, having the interval of zeros as in (3.4).

3.3.2 Defocus Blur

Defocus blur occurs when the lens does not focus correctly on the object. Assumes the lens system is perfect. Defocus PSF degrades an ideal point source into a spreading circle with blur radius R. Hence, the defocus model is [18],

$$h_d(x,y) = \begin{cases} \dfrac{1}{\pi R^2} & x^2 + y^2 \leq R^2 \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

The blur radius is proportional to the extent of defocussing. The Fourier Transform of $h_d(x,y)$ is

$$\begin{aligned} H_d(u,v) &= \mathcal{F}\{h_d(x,y)\} \\ &= \frac{J_1(\pi R r)}{\pi R r}, \end{aligned} \tag{3.8}$$

where $r^2 = u^2 + v^2$.

In (3.8), $J_1(\bullet)$ is the first-order Bessel function of the first kind. The amplitude of $H_d(u,v)$ is characterized by almost periodic circles of zero when r satisfies

$$2\pi R r = 3.83, 7.02, 10.2, 13,3, 16.5, \ldots \tag{3.9}$$

$$\Rightarrow \qquad r = \frac{3.83}{2\pi R}, \frac{7.02}{2\pi R}, \frac{10.2}{2\pi R}, \frac{13.3}{2\pi R}, \frac{16.5}{2\pi R}, \ldots$$

$$r \approx \frac{1}{2R}, \frac{1}{R}, \frac{3}{2R}, \frac{2}{R}, \frac{5}{2R}, \ldots . \tag{3.10}$$

3.4 Choice of PSF Models

Both the motion blur model and the defocus blur model require only a small number of parameters to determine them completely. PSFs like the two discussed above which can be characterized by a few parameters are called deterministic point spread functions [9].

The simplicity of the deterministic PSF model greatly reduces the information demand in PSF measurement. These models are applied by assuming the PSF of the imaging system, whether the models are linear or nonlinear, space invariant or space variant, can be known deterministically. However, the analyst of PSF is often faced with imprecise knowledge because of the an imperfect imaging system and noise. An alternative approach to imprecise knowledge is through statistical description (stochastic PSF) [9].

Stochastic PSF can improve the performance of restoration since it describes the degradation more accurately, however, this approach involves intensive estimating work and is much more difficult to model than a deterministic approach. Stochastic PSFs are computationally intensive, which obscured its use in restoration algorithms. Although the stochastic model is more accurate than the deterministic model, the deterministic model can provide the basic idea and perception of the nature of the system blurring. It is computationally economic, and gives good result in many cases [9,12,13,17]. Once the deterministic model is established, we can refine it to a stochastic model to improve the performance of restoration.

3.5 Power Cepstrum Blur Identification

In the Fourier transform domain of the imaging system described in equation (2.4), the observation of zero crossings in H(u,v) is obscured because of the extreme randomness of the scene F(u,v) and noise N(u,v). These make the zeros hard to observe or they disappear completely [18]. The effect of F(u,v) and N(u,v) on the PSF can be attenuated by using the power spectrum of the observed image. The power spectrum from equation (2.4) is

$$P_g(u,v) = P_f(u,v)\,|H(u,v)|^2 + P_n(u,v), \tag{3.11}$$

where $P_g(u,v)$, $P_f(u,v)$ and $P_n(u,v)$ represent the power spectra of g(x,y), f(x,y) and n(x,y) respectively. The power spectrum $P_g(u,v)$ can be estimated by using Welch's algorithm [20]. The whole image is divided into multiple overlapping subimages. The subimages are multiplied by a 2D Hamming window to avoid wrapping error in the Fourier transform. The square magnitude of the Fourier transform is computed, and averaged over all subimages. The process is written as

$$\begin{aligned}\overline{P_g(u,v)} &= \frac{1}{N}\sum_{i}^{N} |G_i(u,v)|^2 \\ &= \frac{1}{N}\sum_{i}^{N} P_{fi}(u,v)\,|H(u,v)|^2 + P_{ni}(u,v) \\ &= \overline{P_f(u,v)}\,|H(u,v)|^2 + \overline{P_n(u,v)}.\end{aligned} \tag{3.12}$$

The image and the noise are assumed to be samples of a stationary random process, therefore $\overline{P_n(u,v)}$, assuming to be white, converges to a constant equal to noise variance. The term $|H(u,v)|^2$ contains the strong periodicities of the zeros which are not averaged out because of the assumed space invariance of H(u,v). The power spectrum of $P_f(u,v)$ is not periodic because of the averaging process of the individual image segment spectra.

After the image power spectrum is computed, it is transformed into the cepstral domain to locate the zeros. The cepstrum of a function g(x,y) is defined to be:

$$\begin{aligned} C_g(p,q) &= \mathcal{F}^{-1}\{\log|\mathcal{F}\{g(x,y)\}|\} \\ &= \mathcal{F}^{-1}\{\log|G(u,v)|\}. \end{aligned} \tag{3.13}$$

where $\mathcal{F}\{\bullet\}$ is the Fourier transform; $\mathcal{F}^{-1}\{\bullet\}$ is the inverse Fourier transform; $C_g(p,q)$ is the cepstrum of function g(x,y).

The cepstrum of the power spectrum is called power cepstrum. From equation (3.12)

$$\begin{aligned} C_{\bar{g}}(p,q) &= \mathcal{F}^{-1}\{\log|\overline{P_g(u,v)}|\} \\ &= \mathcal{F}^{-1}\{\log|\overline{P_f(u,v)}\,|H(u,v)|^2 + \overline{P_n(u,v)}\,|\}. \end{aligned} \tag{3.14}$$

The averaged noise is assumed to be small and neglected, hence, we have

$$\begin{aligned} C_{\bar{g}}(p,q) &= \mathcal{F}^{-1}\{\log|\overline{P_f(u,v)}\,|H(u,v)|^2|\} \\ &= \mathcal{F}^{-1}\{\log|\overline{P_f(u,v)}| + \log|H(u,v)|^2\} \\ &= \mathcal{F}^{-1}\{\log|\overline{P_f(u,v)}|\} + \mathcal{F}^{-1}\{\log|H(u,v)|^2\} \\ &= C_{\overline{P_f}}(p,q) + 2C_h(p,q). \end{aligned} \tag{3.15}$$

The logarithm of the zeros locating in $|H(u,v)|^2$ produces strong negative spikes. Therefore, the first term in (3.15) on the right hand side does not retain any special characteristics since the image undergoes an averaging process; the second term $C_h(p,q)$ retains the strongly periodic characteristics of the blur function.

The independent variables (u,v) in a Fourier analysis are called frequencies and have the physical dimensions of 1/x and 1/y respectively, where 1/x and 1/y are the physical dimension of the spatial independent variables x and y. The independent variables in the cepstrum domain (p,q) are called quefrencies and have the physical dimensions of 1/u=p and 1/v=q [21].

For the motion blur in (3.3)

$$\log|H(u,v)| = \log\left|\frac{\sin(\pi du)}{\pi du}\right|$$
$$= \log|\sin(\pi du)| - \log|\pi du|. \tag{3.16}$$

The term $\log|\pi du|$ contributes a hyperbolic type of curve in the cepstral domain. The function $\log|\sin(\pi du)|$ has a period the same as (3.4), therefore, there are negative spikes at d, d/2, d/3..., along the direction of motion in the cepstral domain.

The defocus blur is derived in a manner similar to the motion blur except the period is circular and occurs at 2R, R, 2R/3... interval because of the periodicity in (3.10).

The power spectrum is real and positive, therefore the inverse Fourier transform of its log value is also real but contains negative spikes coming from the PSF and partial image data. Thus, the real part of the cepstrum is clipped at zero amplitude (The positive value has no contribution to the identification procedure), inverted and scaled for display and analysis.

When both defocus and motion blur are present in the same image, the two sets of spikes are simply added together because

$$g(x,y) = f(x,y) \otimes h_m(x,y) \otimes h_d(x,y), \tag{3.17}$$

then in cepstrum domain

$$C_g(p,q) = C_f(p,q) + C_{h_m}(p,q) + C_{h_d}(p,q). \tag{3.18}$$

Finally, we should notice that in (3.15), the noise spectra is neglected to derive the equation. When additive noise is introduced, the zeros start to disappear. This happens especially in the high frequency regions of the power spectrum where the noise spectra is highly dominant [9]. This is the reason that this technique performs poorly in very noisy images which have high noise power spectra, dominating a large region of the image power spectrum, even in the low frequency regions. Therefore in practice, only the first period of spikes or maybe the second can be visualized since the higher order of periods are corrupted by noise spectra. In case of motion blur, two strong intensive spikes are located with distance d from the origin are displayed. The motion direction is along the axis which passes through these two spikes. In case of defocus blur, a ring with radius 2R from the origin is displayed.

The blur parameters are measured and are used to construct the PSF for image restoration. This blur identification algorithm can identify the blur with zeros in the frequency domain such as motion blur and defocus blur. The main drawback of this technique is its high sensitivity to additive noise.

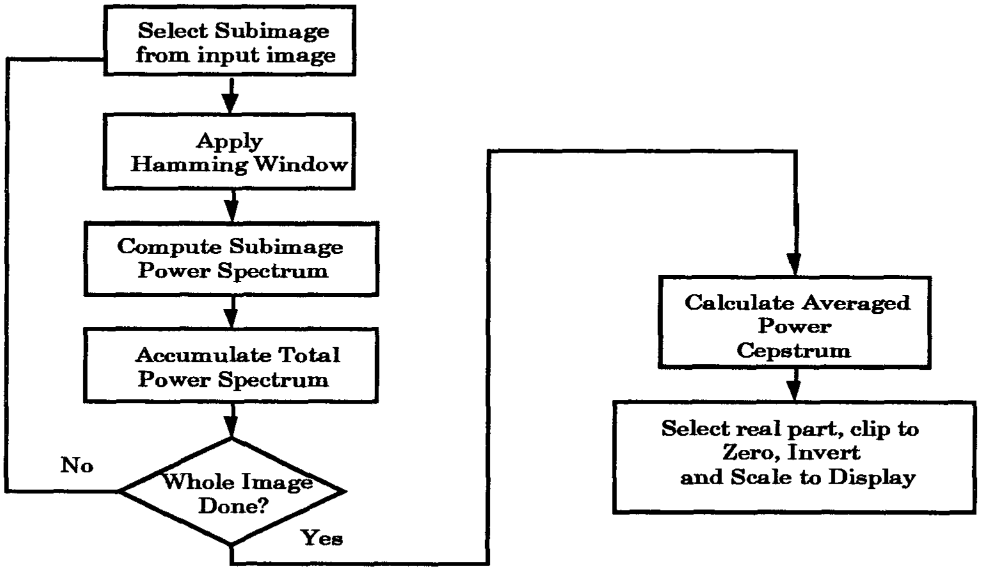
Select Subimage from input image
Apply Hamming Window
Compute Subimage Power Spectrum
Accumulate Total Power Spectrum
Whole Image Done?
No
Yes
Calculate Averaged Power Cepstrum
Select real part, clip to Zero, Invert and Scale to Display

Figure 3.1 Blur identification algorithm.

CHAPTER 4

IMAGE RESTORATION

4.1 Introduction to Image Restoration

The purpose of image restoration is to reconstruct an image that has been degraded by blur and noise. Thus, the goal of restoration techniques is to model the degradation and apply the inverse process in order to recover the original image. However, image restoration is always an ill-posed inverse problem so that a unique solution may not exists, and therefore it is essential to have some *a priori* information about the ideal solution [10]. The performance of an image restoration depends largely on how good the assumed mathematical model fits the degradation characteristics of the image system. We have discussed the degradation models used in our restoration in a previous chapter; they are a motion blur model and a defocus blur model. The decision of which model to use and the appropriate value of parameters to apply to the restoration procedure is made by analyzing the image power cepstrum created by the blur identification process.

Image restoration is different from image enhancement. Basically, image enhancement is procedures designed to manipulate an image in order to take advantage of the psychophysical aspects of the human visual system. Therefore, image restoration can produce dramatic improvement in an image that the image enhancement cannot achieve. However, a restored image may not be pleasant for human observation, and enhancement is also required. Hence, usually both of these techniques should work together to optimize the information recovery in an image [12].

There are hundreds of image restoration algorithms reported in the literature [10]. Each procedure approaches the problem in a different way, e.g. linear, nonlinear, iterative, noniterative, deterministic, stochastic, etc. Although there exist abundant algorithms for image restoration, generally, the restoration problem is solved by defining certain criteria or constraints which allow one to estimate the ideal image [9,10]. The power spectrum equalization filter and Wiener filter were applied in the previous works [5,6]. The power spectrum equalization filter (PSEF) is a homomorphic filter whose restored image is pleasant to the human nonlinear visual system [17]. The Wiener filter is based on a linear minimum mean square error (LMMSE) criterion. The linear minimum mean square error is a much stronger criterion than power spectrum equalization in estimation, and the Wiener filter has been applied successfully in many cases [9,12,13,19]. In addition, the PSEF is a phaseless filter that requires extra phase estimation work when the PSF is not phaseless. The models we used are not phaseless and in fact have phase alternation because of their zero crossings. The PSEF and the Wiener filter have similar performance characteristics in low signal-to-noise ratio images, whereas the Wiener filter is superior to the PSEF in noisy images [22]. Therefore, the Wiener filter has been employed to restore the degraded image because of its strong criterion and effectiveness in practice. Furthermore, the Wiener filter is computationally economic compared to most of the other restoration algorithms [10] because it can be simply implemented in the frequency domain, increasing the computation efficiency significantly.

4.2 Wiener Filter

Recall from (2.3), that when the image acquisition system is a linear, signal-independent-noise model, the matrix vector equation of (2.3) can be written as

$$\mathbf{g} = \mathbf{Hf} + \mathbf{n}, \tag{4.1}$$

where **g**, **f** and **n** represent the vector matrix of observed image, original image and noise, and **H** is the block Toeplitz matrix of the PSF [9].

The criterion used by the Wiener filter is the minimum mean square error. Assume there is an estimate **f~** of the original intensity distribution. The error of the estimate is

$$\varepsilon = \mathbf{f} - \mathbf{f}^{\sim}. \tag{4.2}$$

The MMSE requires the error of the estimation to be minimized, that is

$$\min_{\mathbf{f}^{\sim}} E\{\varepsilon\varepsilon^T\}, \tag{4.3}$$

where E{•} is the expectation value operation. Since **H** is linear, the estimation is linear. And **f~** can be derived by a linear operation **L** on the observed image,

$$\mathbf{f}^{\sim} = \mathbf{Lg}. \tag{4.4}$$

Hence,

$$\begin{aligned}\min_{\mathbf{f}^{\sim}} E\{\varepsilon\varepsilon^T\} &= E\{(\mathbf{f} - \mathbf{Lg})(\mathbf{f} - \mathbf{Lg})^T\} \\ &= E\{(\mathbf{f} - \mathbf{L}(\mathbf{Hf} + \mathbf{n}))(\mathbf{f} - \mathbf{L}(\mathbf{Hf} + \mathbf{n}))^T\} \\ &= \mathbf{R_f} - \mathbf{LHR_f} - \mathbf{R_f HL}^T + \mathbf{LHR_f H}^T\mathbf{L}^T + \mathbf{LR_n L}^T, \end{aligned} \tag{4.5}$$

since signal independent noise is assumed,

$$E\{\mathbf{fn}^T\} = E\{\mathbf{n}^T\mathbf{f}\} = 0, \tag{4.6}$$

and the autocorrection matrix of **f** and **n** are

$$E\{\mathbf{ff}^T\} = \mathbf{R_f}; \qquad E\{\mathbf{nn}^T\} = \mathbf{R_n}. \tag{4.7}$$

Differentiating equation (4.5) with respect to $\mathbf{L}$ and setting the result equal to zero to get the minimum value, the solution is

$$\mathbf{L} = \mathbf{R_f H^T (H R_f H^T + R_n)^{-1}}. \tag{4.8}$$

The matrixes $\mathbf{H}$, $\mathbf{R_f}$ and $\mathbf{R_n}$ are approximated by block circulants to utilize the rapid Fourier transform computation. Let $\mathbf{Q_g}$ be a circulant matrix [9], then

$$\Lambda_{\mathbf{g}} = [\mathcal{F}]\mathbf{Q_g}[\mathcal{F}^{-1}] \tag{4.9}$$

and

$$\Lambda_{\mathbf{g}}^* = [\mathcal{F}]\mathbf{Q_g}^T[\mathcal{F}^{-1}], \tag{4.10}$$

where $[\mathcal{F}]$ and $[\mathcal{F}^{-1}]$ is the discrete Fourier transform matrix and its inverse, that

$$[\mathcal{F}^{-1}]\mathbf{g} = \mathcal{F}\{\mathbf{g}\}, \tag{4.11}$$

and $\Lambda_{\mathbf{g}}$ and $\Lambda_{\mathbf{g}}^*$ is the eigenvalue matrix of $\mathbf{Q_g}$ and its conjugate. By applying the circulant matrix and diagonalization, equation (4.8) becomes

$$\mathbf{L} \approx [\mathcal{F}]\Lambda_{\mathbf{f}}\Lambda_{\mathbf{h}}^*(\Lambda_{\mathbf{h}}\Lambda_{\mathbf{f}}\Lambda_{\mathbf{h}}^* + \Lambda_{\mathbf{n}})^{-1}[\mathcal{F}^{-1}]. \tag{4.12}$$

Substituting (4.11) into (4.4), we have

$$[\mathcal{F}^{-1}]\mathbf{f}^{\sim} = \Lambda_{\mathbf{f}}\Lambda_{\mathbf{h}}^*(\Lambda_{\mathbf{h}}\Lambda_{\mathbf{f}}\Lambda_{\mathbf{h}}^* + \Lambda_{\mathbf{n}})^{-1}[\mathcal{F}^{-1}]\mathbf{g}. \tag{4.13}$$

Therefore the Wiener filter in Fourier domain is represented by

$$F^{\sim}(u,v) = \frac{P_f(u,v)H^*(u,v)G(u,v)}{H^*(u,v)H(u,v)P_f(u,v) + P_n(u,v)}$$

$$= \left[\frac{H^*(u,v)}{|H(u,v)|^2 + P_n(u,v)/P_f(u,v)} \right] G(u,v), \tag{4.14}$$

where $P_f(u,v)$ and $P_n(u,v)$ are the power spectra of the original image and the noise. The term $P_n(u,v)/P_f(u,v)$ is inversely relates to the signal-to-noise ratio because

$$\text{SNR} = 10 \log \left(\frac{\text{variance of signal}}{\text{variance of noise}}\right) \tag{4.15}$$

In practice, this information is usually unavailable and are approximated by a constant K, so that

$$F^{\sim}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2 + K} \right] G(u,v), \tag{4.16}$$

The constant K is a factor that trades the visual sharpness off against ringing, artifacts and/or noise in the restored image [19].

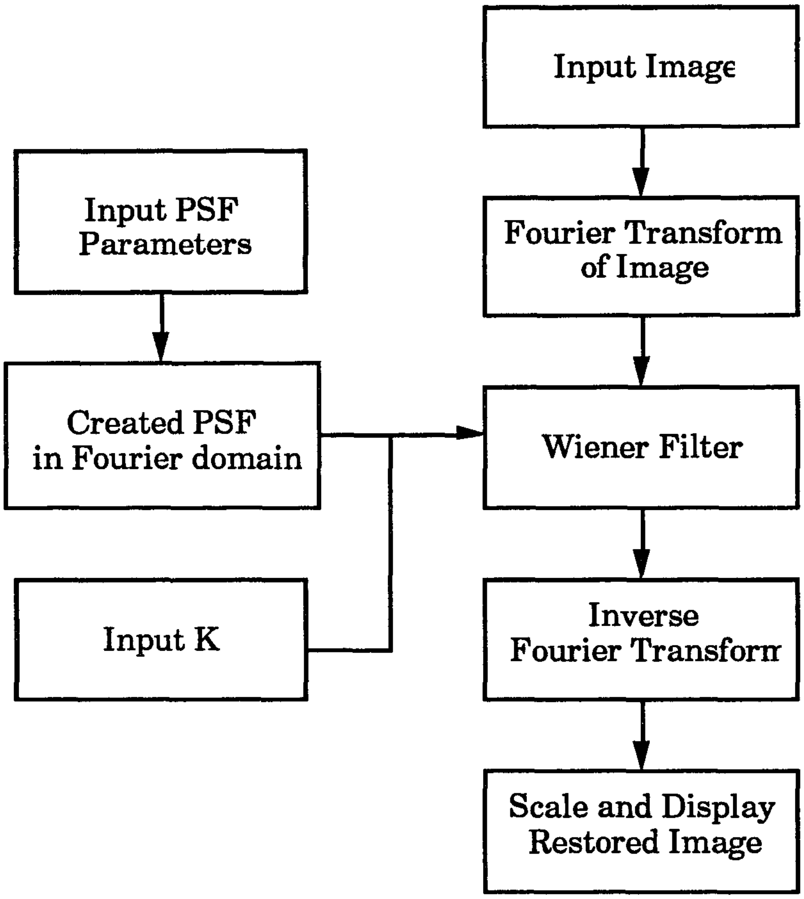
Input Image
Fourier Transform of Image
Input PSF Parameters
Created PSF in Fourier domain
Wiener Filter
Input K
Inverse Fourier Transform
Scale and Display Restored Image

Figure 4.1 Wiener filter algorithm.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Testing of Blur Identification and Restoration Algorithm

The blur identification program and Wiener filter were tested before being applied to microscope images. The original test image, along with images created using horizontal motion blur with d=18, and defocus blur with R=8 are shown in figure 5.1. Their corresponding cepstral domain images, created by the blur identification program, are displayed beside them. Notice that in the cepstral domain images, each tick interval on the horizontal and vertical scales is five pixels. For the defocus blur, a ring with radius of 16 pixels (twice the defocus radius R) clearly appears. For the motion blur, two bright spikes lie along the horizontal axis. These spikes indicate that the motion direction is horizontal. These points are located 18 pixels away from the origin, which is the motion blur distance d. These blur parameters are used to produce the corresponding PSF for image restoration by the Wiener filter with the results shown in figure 5.2. The images are restored successfully. Although noise and some artifacts are introduced by the Wiener filter, most fine detail in the restored image reappears.

5.2 Blur Identification and Restoration of Microscope Images

Images of a micrometer scale were recorded on videotape through the microscope. The micrometer scale was deliberately defocussed to obtain a severely blurred image. The micrometer scale image, its blurred version, restored version, and their corresponding cepstrum are shown in figure 5.3. Notice that the spikes on the in-focus image cepstrum come from the periodic pattern of the micrometer. An ellipse-shaped ring is present on the defocussed image cepstrum. The defocus blur characteristic is ellipse-shaped because the imaging system is imperfect. By analyzing the defocussed image cepstrum, a defocus blur radius of R=5 was applied to restore the blurred image, using K=0.04. Most of the fine detail of the micrometer scale image was recovered. Since the defocus blurred PSF could not exactly match the actual blur characteristics of the defocussed image, there was some ring residue on the restored image cepstrum, which was introduced by the defocus blur PSF. The difference between the PSF and the real blur characteristics explains why the blurred image could not be restored completely. Furthermore, the noise and the unknown information about the original image power spectrum also play a role. In figure 5.4, the restored image was enhanced by contrast stretching, and it became much more observable and started to resemble the in-focus image.

Different defocus blur radii were tried on the blurred image. These are shown in figure 5.5. Neither R=4 nor R=6 produced a recovered image as completely as R=5 did. The effect of changing R resembles that of changing focal plane on the blurred image because different portions of the blurred image were recovered with different R value. The value of K=0.04 was used on the images shown in figure 5.5. In figure 5.6, K values of 0.5, 0.005 and 0.0005 are shown. The smaller the value of K, the stronger the influence of artifacts and noise were on the restored image. The larger the value of K, the smoother the restored image became. The range of K from 0.08 to 0.01 produced restored images with similar quality. This illustrates that a wide range of K values can be acceptable in restoration. On the contrary, image recovery is highly sensitive to R, suggesting that the matching between the PSF and the actual blur characteristics are very critical to the success of image recovery.

5.3 Blur Identification and Restoration of Coronary Microvessel Images

Images of coronary microvessel obtained from a previous work [6] were transferred and converted into Application Visualization System (AVS) compatible format. Figure 5.7 shows the original image displayed on the TV monitor of the imaging system. Figure 5.8 shows the same image displayed on the AVS workstation monitor, which shows a significant improvement in quality because of the higher resolution of the AVS workstation monitor. There are two spikes on the image cepstrum in figure 5.9 top, with distance of two pixels from the origin on the horizontal axis, suggesting that there is horizontal motion. Therefore the image was restored by the horizontal motion algorithm with d=2 and K=0.001. The restored image is shown in figure 5.9 middle. Since the motion blur is very slight, the restored image did not show much difference from the original one. The restored image was enhanced. This is shown in figure 5.9 bottom, and the contour of the vessels shows clearly.

A set of lower quality microvessel images were obtained for blur identification. The images and their enhanced version are shown on figure 5.10 and 5.11. Their cepstrum appears in figures 5.12 and 5.13. It is very surprising that all the cepstrum were identical, with the same blur characteristics. Horizontal motion blur PSF with d=2 was applied to restore the images, but there was no significant improvement. In addition, the blur characteristics of the restored image remained the same. Therefore this characteristic is quite different from the motion blur PSF even though it presents itself as motion blur. Figure 5.9 top shows an image pattern appearing on the cepstrum of a higher quality image, which indicates that the blur characteristics are not very dominant and are similar to the motion blur PSF. In that kind of image, the blur characteristics were attenuated when the motion PSF is applied to restore the blurred image. However, in lower quality images, the blur characteristics are dominant since no image pattern appears on their cepstrum. Severe blur characteristics in the images are much different than the motion blur PSF, and therefore the motion blur PSF is not effective on the restoration process. Furthermore, since the images come from different scenes, it is not possible that their motion blur would have the same direction and equal extent. Therefore, the blur characteristics may come from the imaging system, which shows a similar pattern to motion blur on the cepstrum. It is also possible that the blur is space variant or the actual motion blur characteristics are very different from the proposed motion blur PSF. Since a motion blur microscope image is unavailable, the actual motion blur characteristics cannot be evaluated and compared with the proposed motion blur PSF. Defocus blur or other motion blur characteristics were not found on the cepstrum, suggesting that the strobe technique does a good job in acquiring the coronary microvessel images.

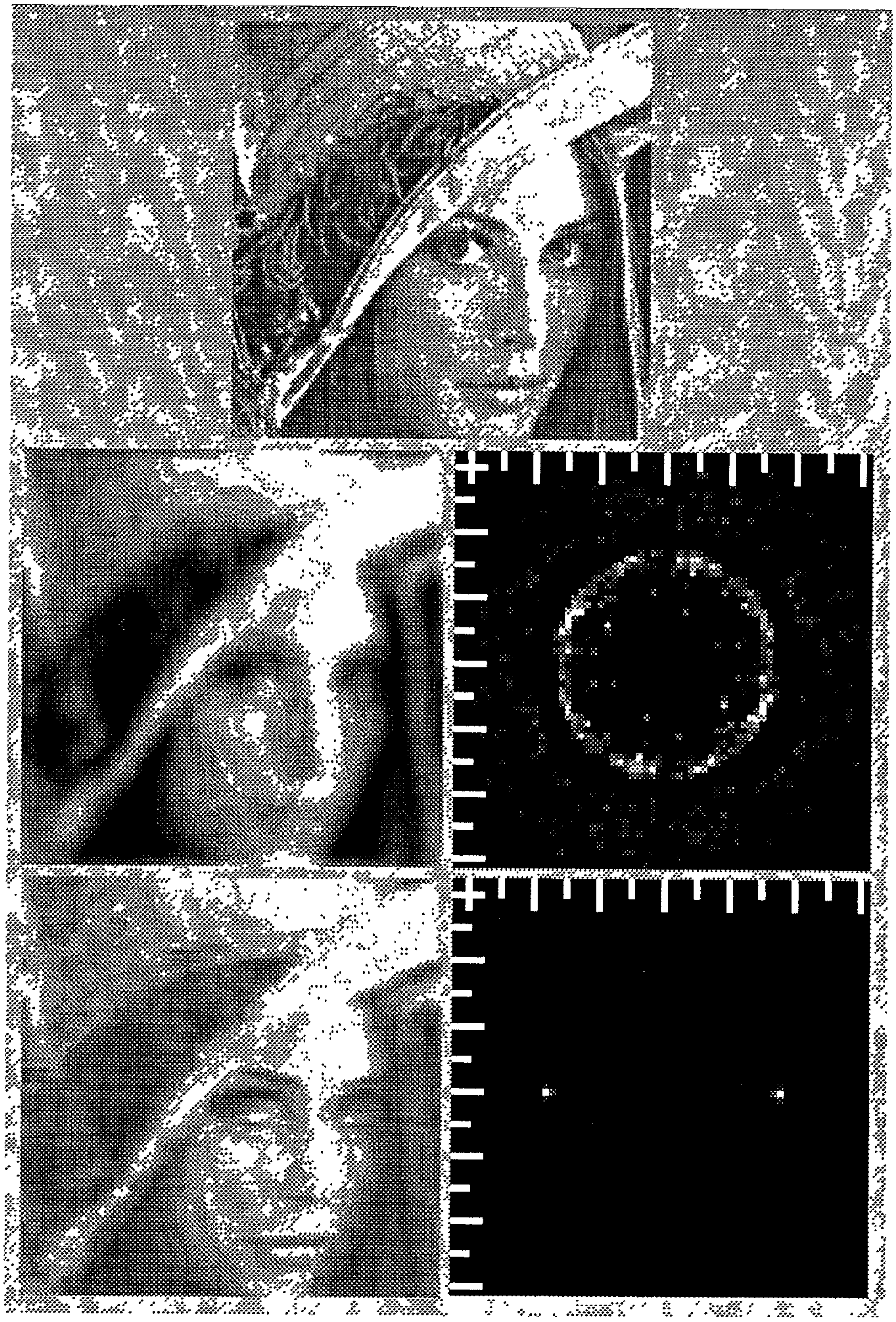

Figure 5.1 Blur identification and restoration of test images.
Top: original test image.
Middle: defocus blur with R=8 and its cepstrum.
Bottom: motion blur with d=18 and its cepstrum.

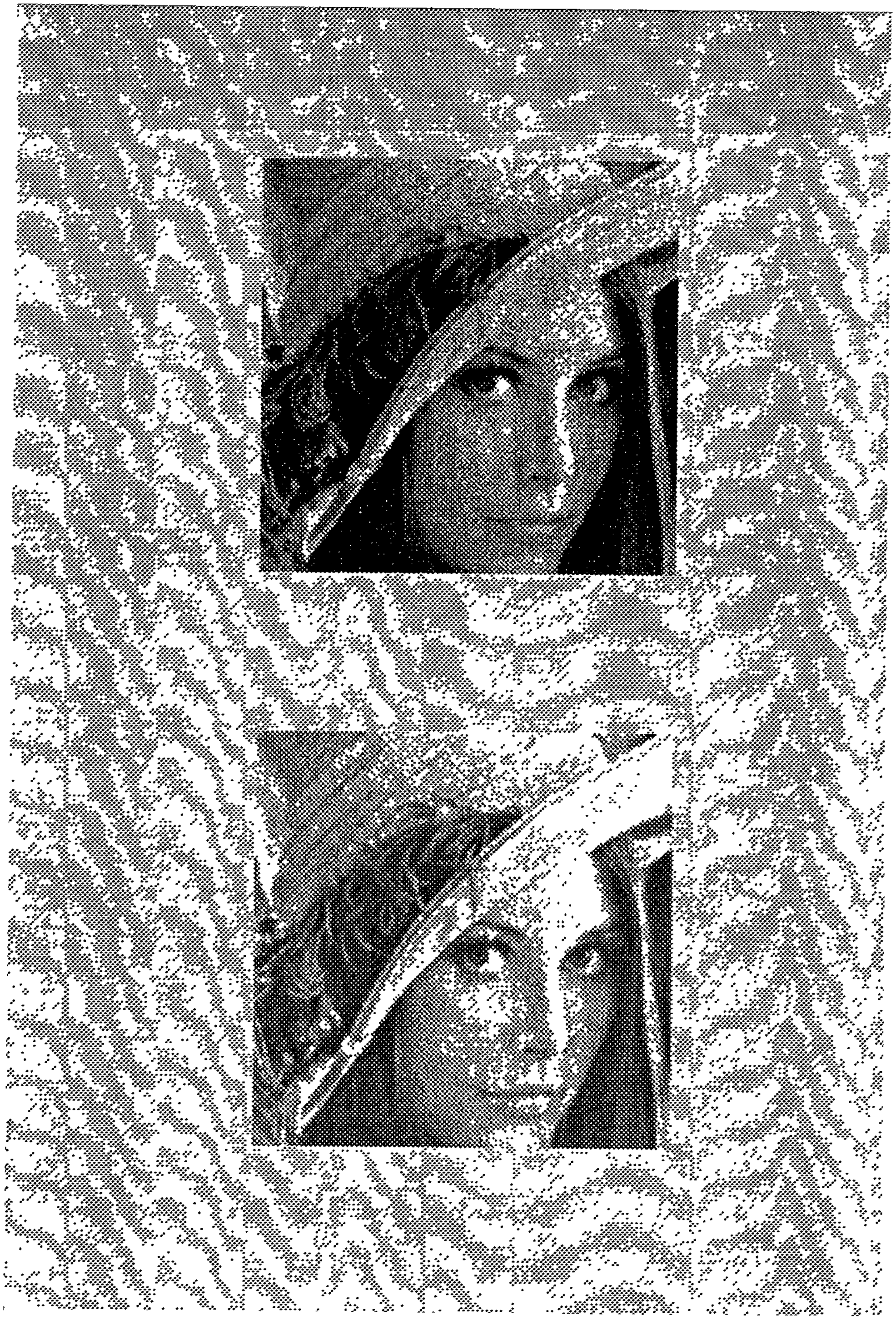

Figure 5.2 Restored test images.
Top: restored defocus blurred image.
Bottom: restored motion blurred image.

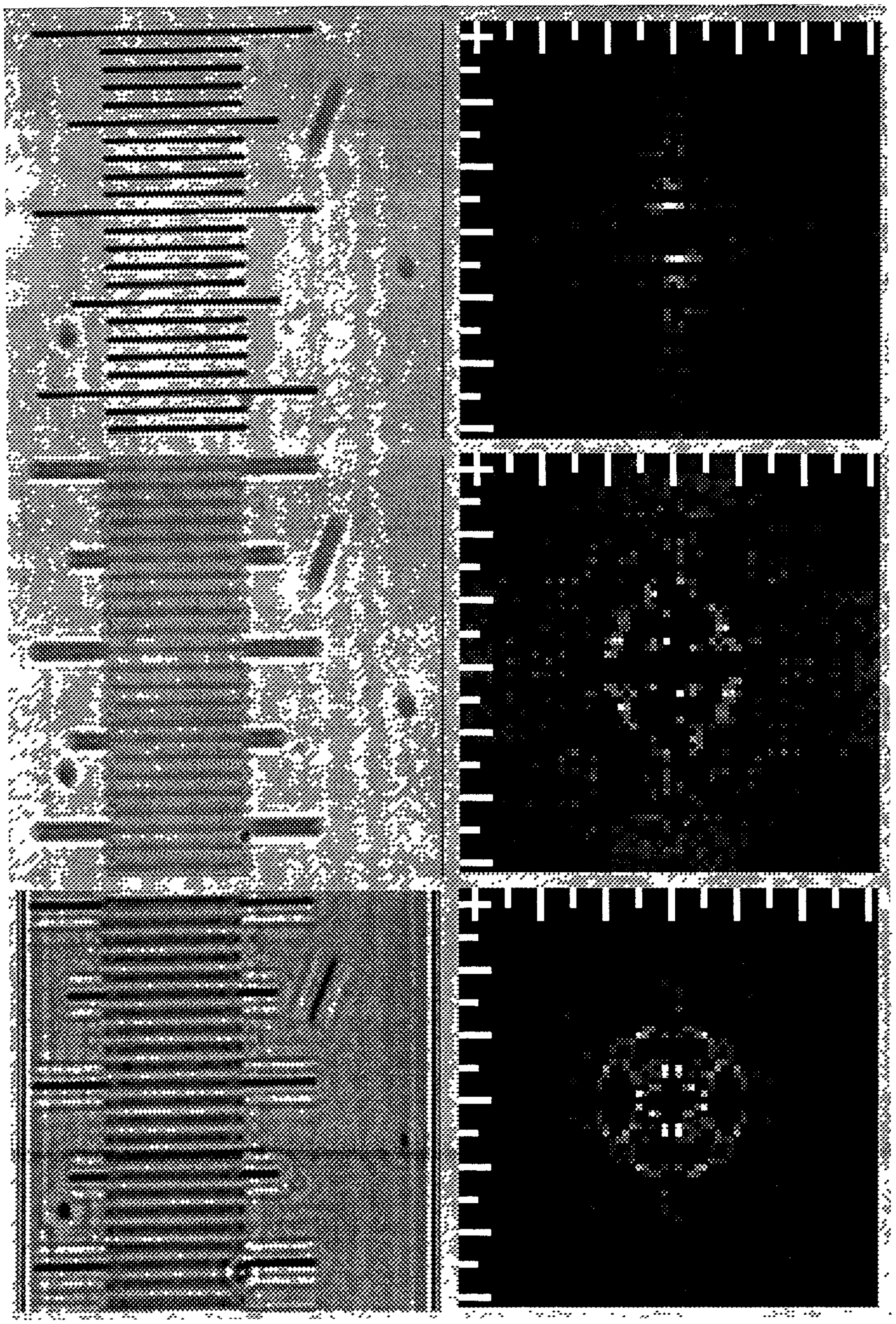

Figure 5.3 Micrometer images.
Top: on-focussed image and its cepstrum.
Middle: defocussed image and its cepstrum.
Bottom: restored image and its cepstrum.

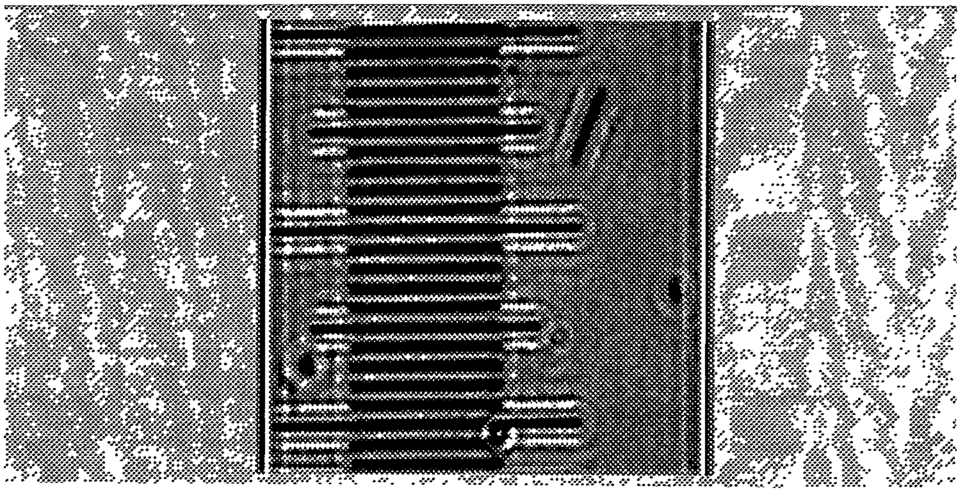

Figure 5.4 Enhanced version of the restored micrometer image.

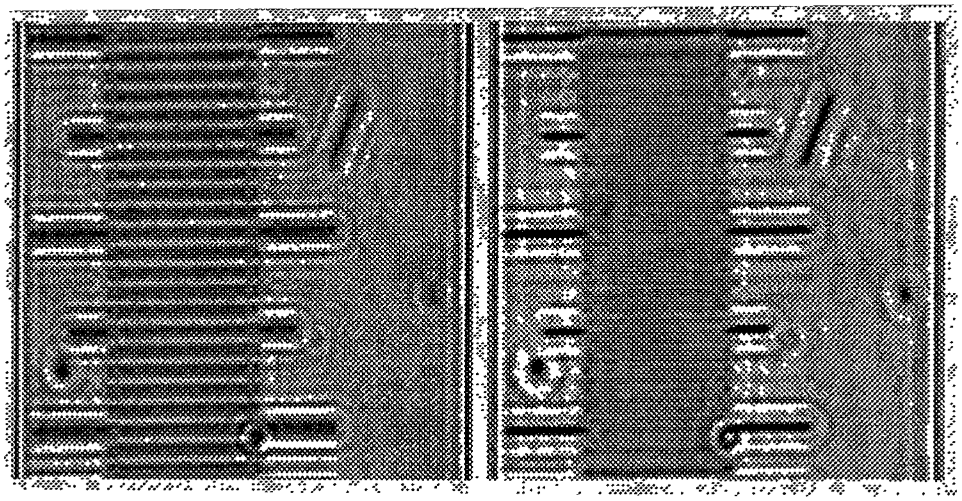

Figure 5.5 Micrometer images restored with different R value.
Left: R=4.
Right: R=6.

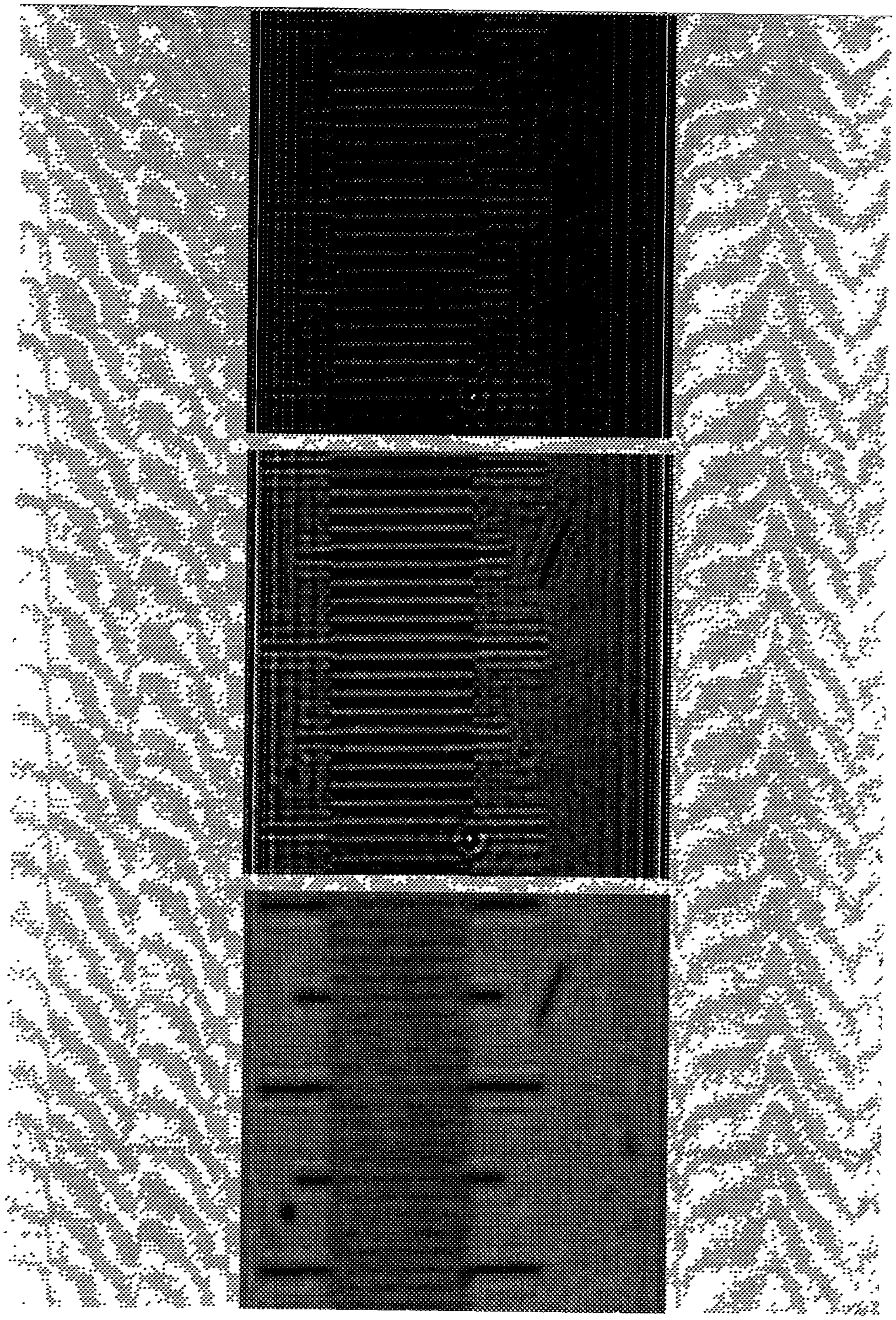

Figure 5.6 Micrometer images restored with different K value.
Top: K=0.0005
Middle: K=0.005
Bottom: K=0.5

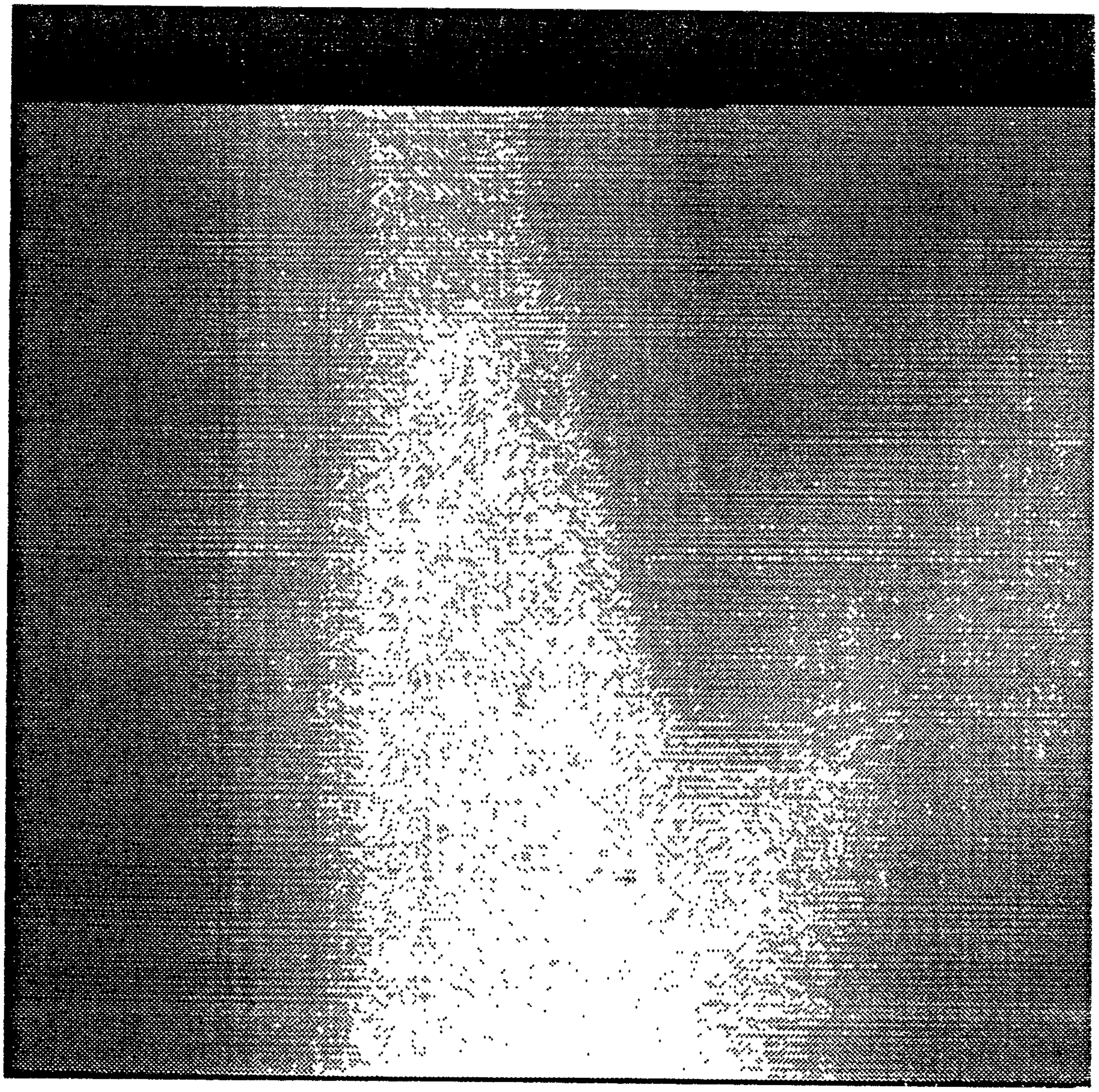

Figure 5.7 Microvessel image displayed on TV monitor.

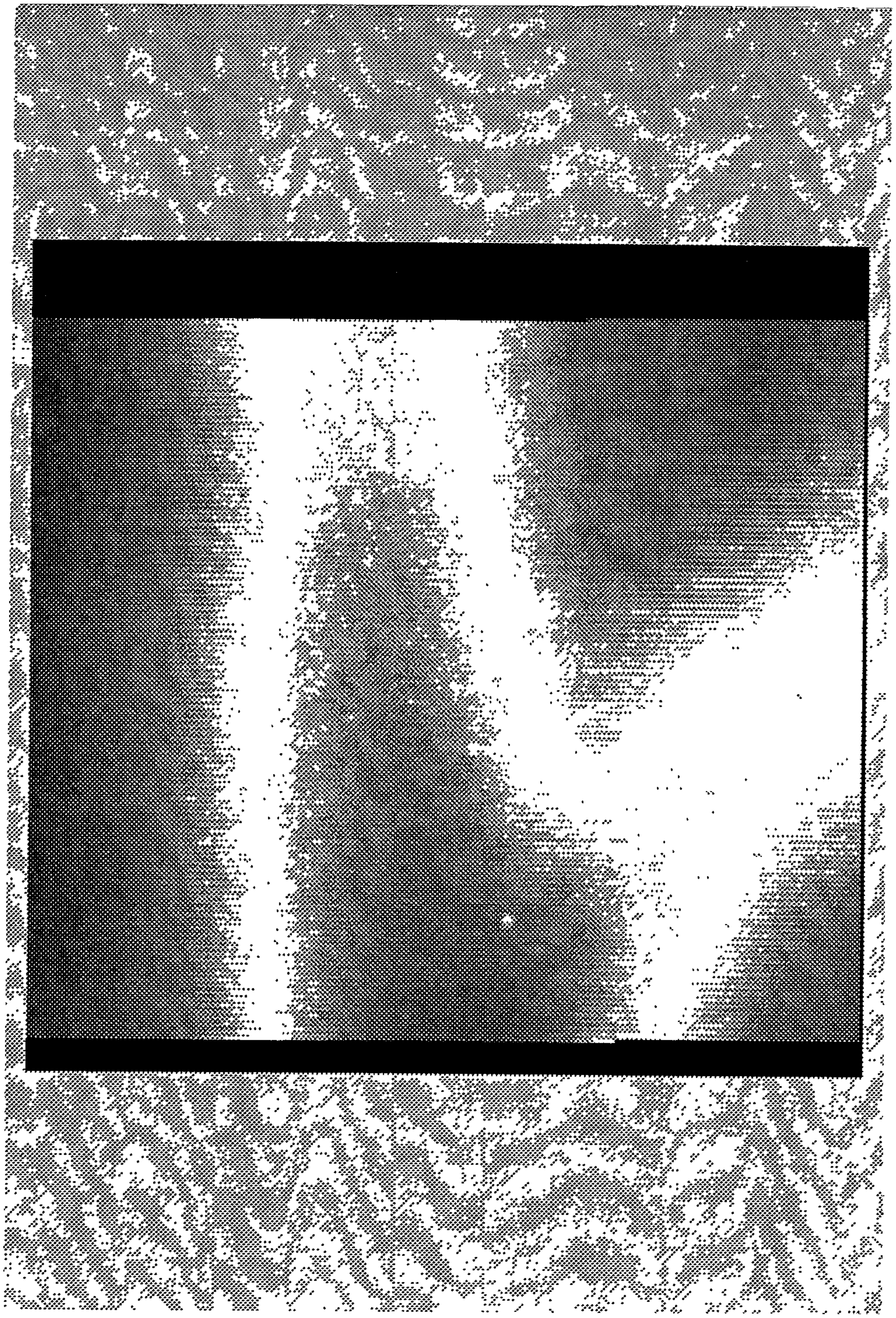

Figure 5.8 Microvessel image displayed on AVS workstation monitor.

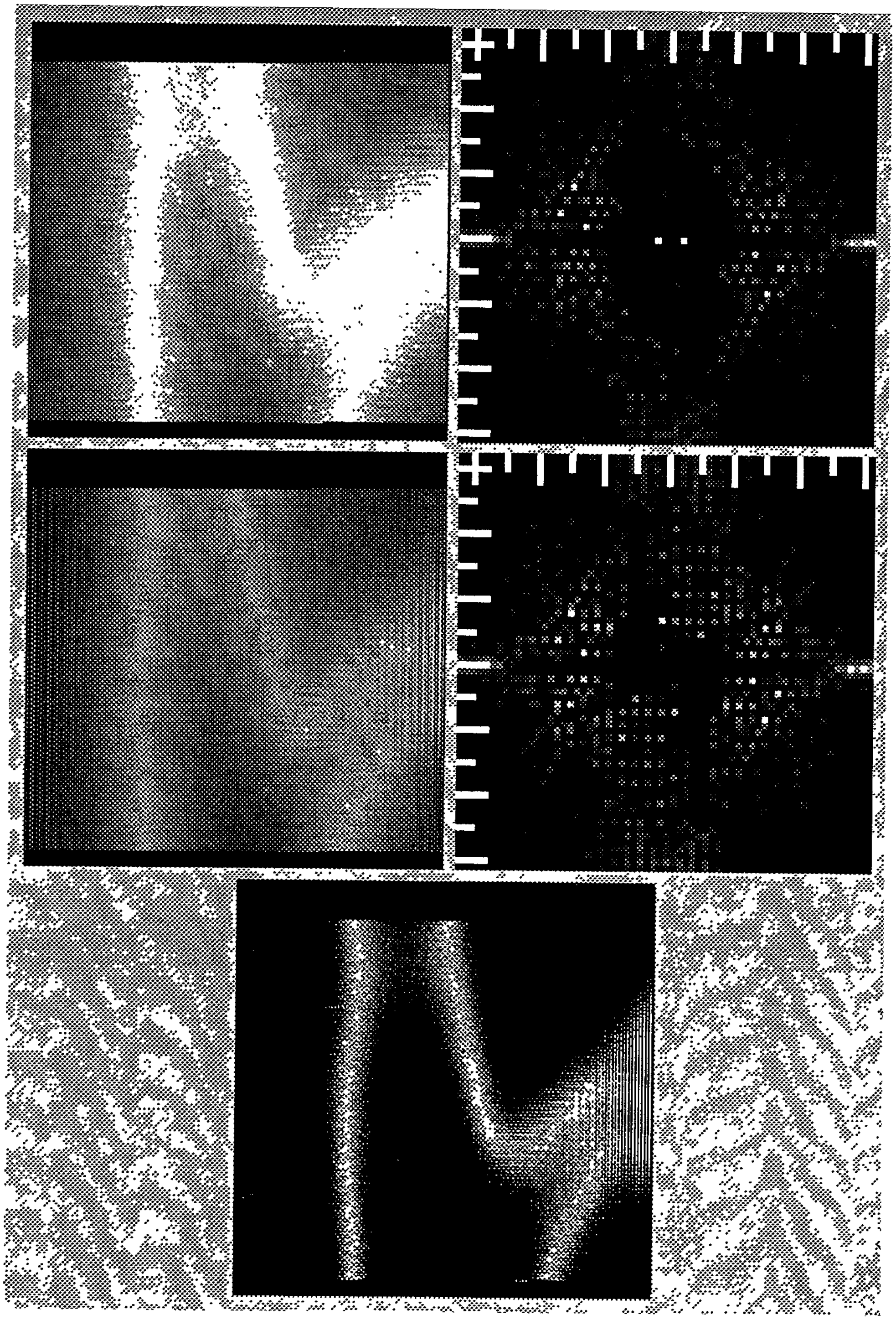

Figure 5.9 Blur identification and restoration of the microvessel image.
Top: original image and its cepstrum.
Middle: restored image and its cepstrum.
Bottom: enhanced image.

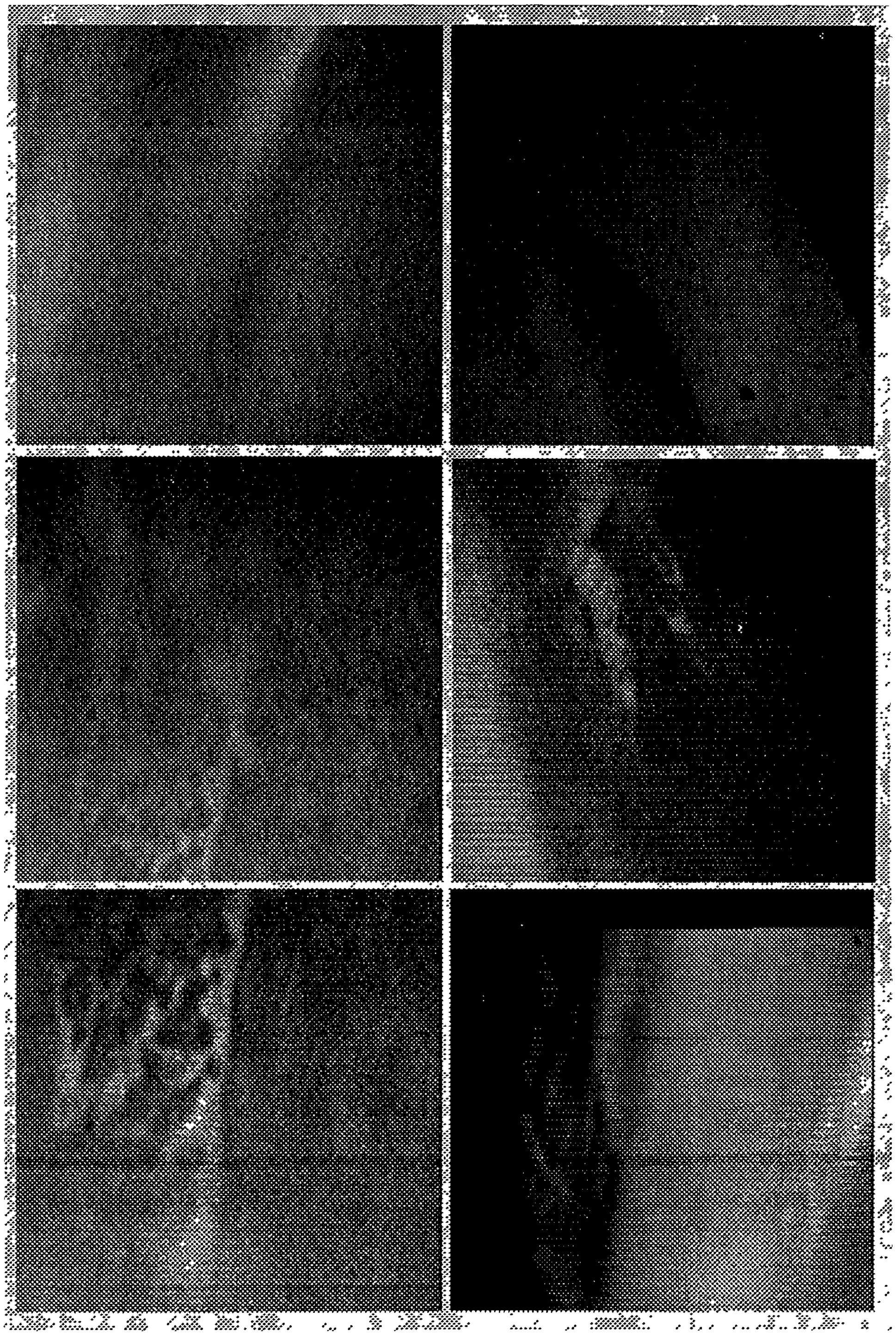

Figure 5.10 Microvessel images with lower quality.

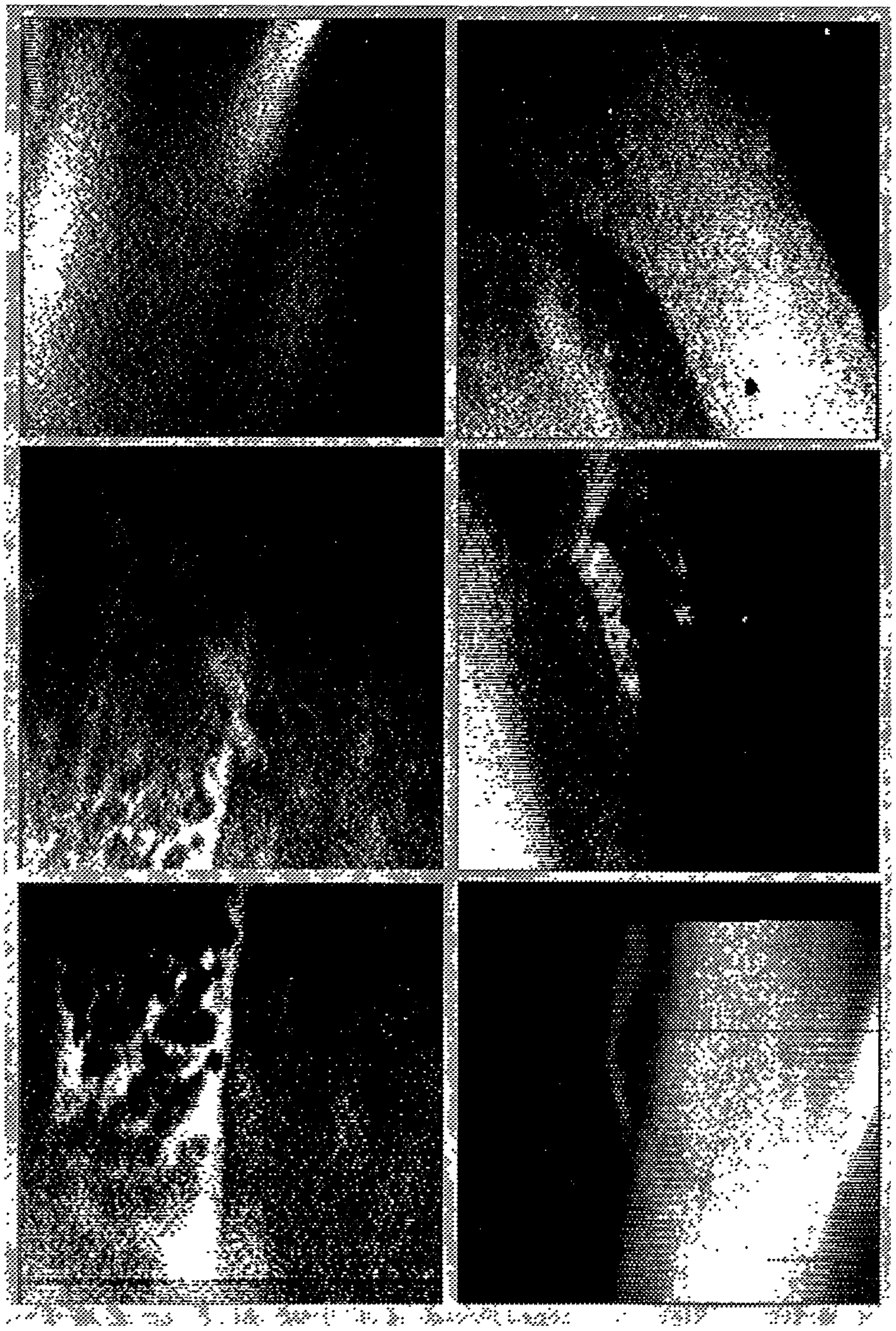

Figure 5.11 Enhanced microvessel images.

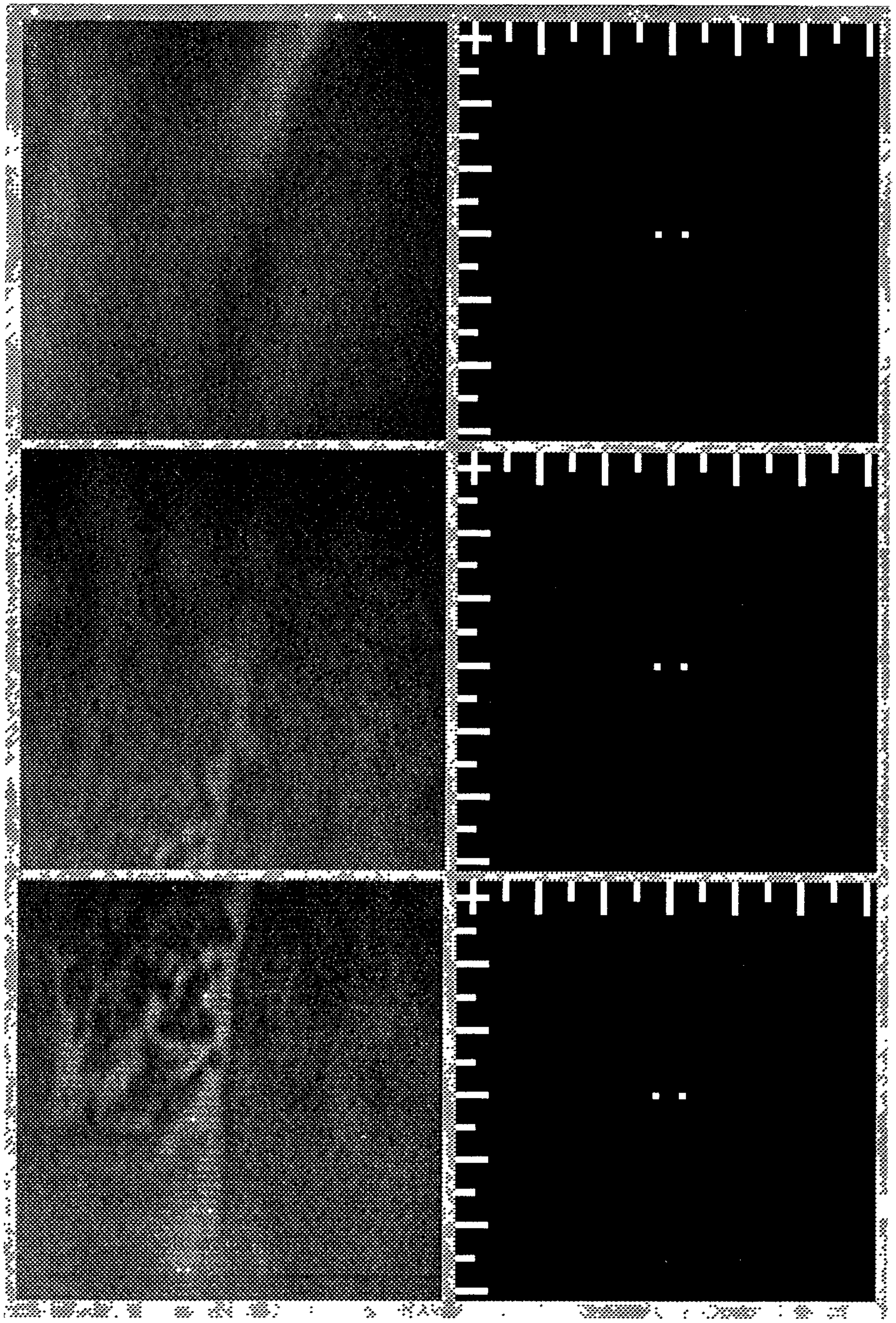

Figure 5.12 Blur identification of the microvessel images with lower quality.

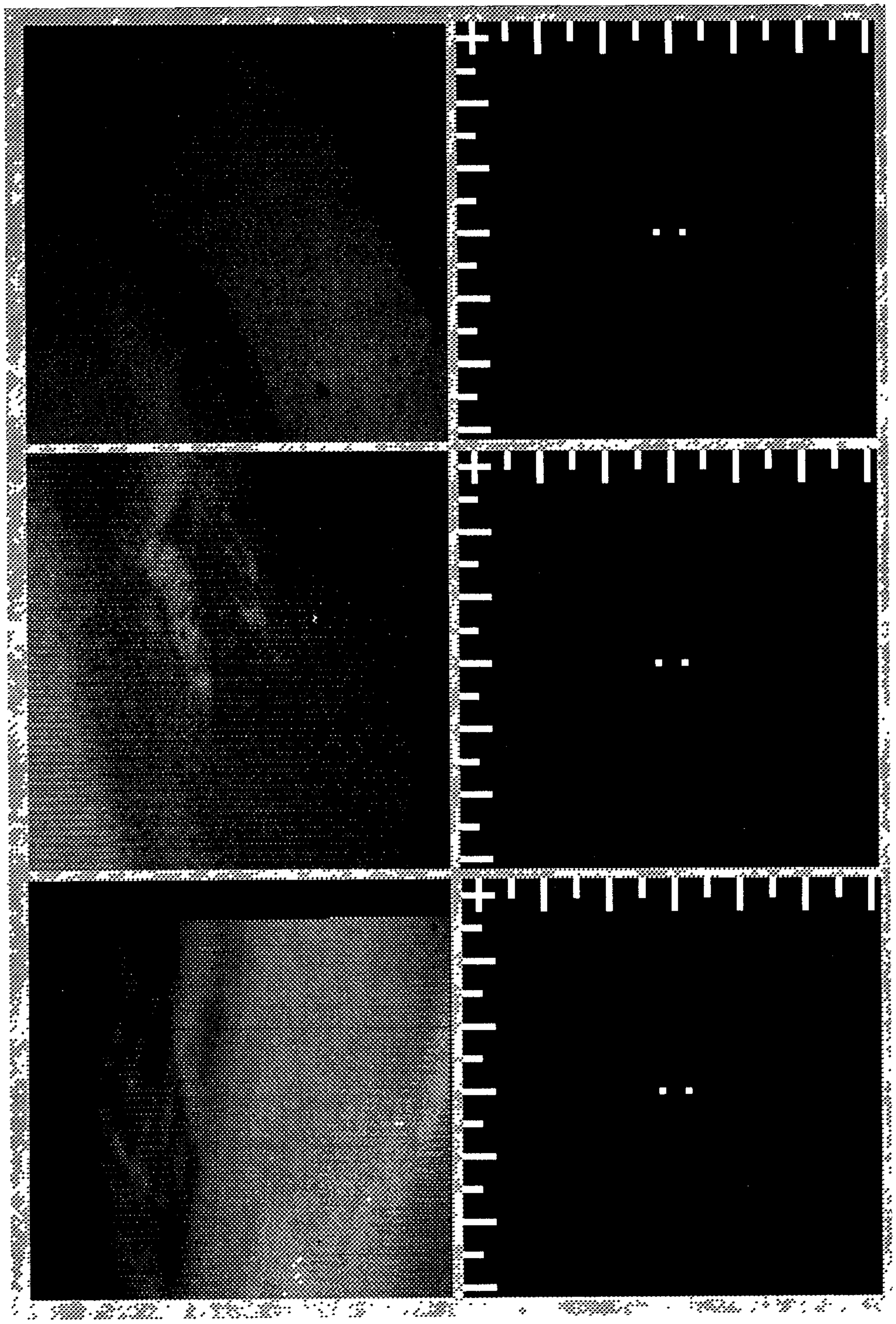

Figure 5.13 Blur identification of the microvessel images with lower quality (continuous).

CONCLUSION

Coronary microvessel images obtained from a beating rat heart were transferred from an HP A900 system to a Unix based system and converted into the software called Application Visualization System (AVS) compatible format, which could utilize its image processing techniques and the high resolution display of the AVS workstation. Two blur models: defocus blur PSF and motion blur PSF were proposed for image restoration. The images underwent blur identification by the power cepstrum method to estimate the blur nature and extract the blur parameters. When defocus blur occurs on the image, a ring appears on its cepstrum with radius twice the defocus blur radius R. If the blur is motion, two spikes with blur distance d away from the origin appear symmetrically on the cepstrum, and the axis passing through these two spikes indicates the direction of the motion. This information was used in conjunction with Wiener filter to restore the blurred images.

Images of a stage micrometer were defocussed and used to evaluate the blur identification and restoration algorithm. The defocus blur characteristics of the imaging system was ellipse-shaped because of imperfections in the imaging system. However, the defocus PSF could mostly recover the blurred images, although the PSF characteristics did not match the image blur characteristics completely. In addition, image recovery was very sensitive to the defocus blur radius R, since only a particular value of R could be used to recover a given image. On the other hand, values of K had a much greater range of tolerance. The K value determined the smoothness or the amount of artifact and noise introduced into the restored image. A small value of K produces a smoother restored image. A larger value of K produces a sharper restored image but containing more artifacts and noise. A microscope image with a known motion blur was not available for this work, and therefore the motion blur characteristics of the image system could not be evaluated and compared with the proposed PSF.

Different coronary microvessel images were transferred from a HP A900 system to the Unix system and converted to AVS compatible images. The images were manipulated by the AVS image processing techniques and their enhanced images became more observable and informative. They all behaved the same in their blur identification cepstrum in that they were very similar to images which exhibited horizontal motion blur with d=2. However, the motion blur PSF could not remove the blur characteristics from the processed images, and therefore the images did not show much improvement. These blur characteristics may come from the imaging system because all the images contained these blur characteristics. No defocus or other motion blur characteristics appeared in the image cepstrum. Therefore, the strobe technique was capable of acquiring stationary coronary microvessel images. Further research is required to investigate the source of the blur characteristics which appeared in all the strobed images, and to verify the actual blur characteristics in order to refine the defocus and motion blur PSF or to propose other more appropriate PSFs to improve image restoration.

APPENDIX

PROGRAM LISTING

The programs written in this research are listed below,

Program 1: raw2avs.c

It converts the image file acquired from the microscope imaging system to AVS compatible format.

Program 2: biden.c

This is the blur identification program. It outputs the power cepstrum of an input image. The size of the cepstrum is 64x64.

Program 3: wiener.c

This is the Wiener Filter used to restore the blurred image.

The program 2 and 3 are AVS format compatible. It requires the input image in AVS format and its output image is also in AVS format.

Program 1: raw2avs.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

main(argc,argv)
    int argc;
    char *argv[];
{
   int i,j,idx,xsize,ysize,gsize,avsize,nln;
   char buf[51],*gimg,*avsimg,*sptr,pix;
  FILE *gfp,*avsfp;

  if(argc!=2)
   {
     printf("Convert raw file to AVS format\n");
     printf("Usage: raw2avs [input-file]\n");
    exit(1);
   }
  gfp = fopen(argv[1],"rb");
  if (!gfp)

{
    printf("Error in reading file %s \n",argv[1]);
   exit(1);
 }
for (i=1;i<=2;i++)
 fgets(buf,50,gfp);
 fscanf(gfp,"%s",buf);
 fscanf(gfp,"%s",buf);
xsize=atoi(buf);
 fscanf(gfp,"%s",buf);
ysize=atoi(buf);
fgets(buf,50,gfp);
gsize = xsize*ysize;
avsize =4*gsize;
 gimg = (char *)calloc(gsize,sizeof(char));
 avsimg = (char*)calloc(avsize,sizeof(char));
 if((!avsimg)| |(!gimg))
 {
    printf("memory request failed\n");
   exit(1);
 }
if (xsize<=170)
 {
   nln = (xsize+1)/2;
   for (j=0,idx=0; j<=(ysize-1); j++)
       for (i=0; i<=(xsize-1);i++,idx+=4)
        {
          pix=getc(gfp);
          avsimg[idx]=255;
           avsimg[idx+1]=pix;
           avsimg[idx+2]=pix;
           avsimg[idx+3]=pix;
          if (i==(nln-1))
            pix=getc(gfp);
        }
 }
else
 {
    if(fread (gimg,sizeof(char),gsize,gfp)!=gsize)
      {
        printf("Input file error! \n");
        exit(1);
      }
   for (j=0,idx=0; j<=(ysize-1); j++)
       for (i=0; i<=(xsize-1);i++,idx+=4)
        {
          avsimg[idx]=255;
           avsimg[idx+1]=gimg[j*xsize+i];
           avsimg[idx+2]=gimg[j*xsize+i];
           avsimg[idx+3]=gimg[j*xsize+i];
        }
  }
  sptr = strchr(argv[1],'.');
 if (sptr)
   *sptr = '\0';
  strcat(argv[1],".x");
  printf("Output file name : %s\n",argv[1]);
  avsfp = fopen(argv[1],"wb");
 if (!avsfp)
  {
     printf("Error in writing file %s \n",argv[1]);
    exit(1);
  }
 if((fwrite (&xsize, sizeof(int),1,avsfp)!=1)||
    (fwrite (&ysize, sizeof(int),1,avsfp)!=1))
  {
     printf("Output file error \n");
    exit(1);
  }
  if(fwrite (avsimg, sizeof(char),avsize,avsfp)!=avsize)
  {
     printf("Output file error \n");
    exit(1);
  }
  printf("Program completed!\n");
 fclose(gfp);
 fclose(avsfp);
}
```

Program 2: biden.c

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>

#define IMG_SIZE 256
#define SPM_SIZE 64
#define STX 16
#define PI 3.141592654
#define MAGSQ(a,b) ((double)(a)*(double)(a)+(double)(b)*(double)(b))
#define WINDOW(a) 0.5*(1.0-cos(2.0*PI*(double)(a)/(SPM_SIZE-1)))
/*Hanning*/

#define SWAP(a,b) tempr=(a);(a)=(b);(b)=tempr

 /*****************************************************/
 /* Function : fourn(data,nn,ndim,isign)              */
/*          N-dimension Fourier Transform                  */
/* data : input complex number data which stored in        */
/*      row major format.                                  */
/*      Data started with index at 1, stored with          */
/*      real number follow by imagine number               */
/* nn   : number of dimension                              */
/* ndim : store the length of each dimension, started      */
/*      with index 1                                       */
/* isign: -1 for forward transform                         */
/*       1 for inverse transform                           */
/*********************************************************/

void fourn(data,nn,ndim,isign)
   float data[];
    int nn[],ndim,isign;

{
 int i1,i2,i3,i2rev,i3rev,ip1,ip2,ip3,ifp1,ifp2;
 int ibit,idim,k1,k2,n,nprev,nrem,ntot;
 float tempi,tempr;

  double theta,wi,wpi,wpr,wr,wtemp;

 ntot=1;
  for (idim=1;idim<=ndim;idim++)
   ntot *= nn[idim];
 nprev=1;
  for (idim=ndim;idim>=1;idim--)
   {
     n=nn[idim];
     nrem=ntot/(n*nprev);
    ip1=nprev << 1;
    ip2=ip1*n;
    ip3=ip2*nrem;
    i2rev=1;
    for (i2=1;i2<=ip2;i2+=ip1)
        {
         if (i2 < i2rev)
          {
           for (i1=i2;i1<=i2+ip1-2;i1+=2)
               {
                for (i3=i1;i3<=ip3;i3+=ip2)
                 {
                  i3rev=i2rev+i3-i2;
                  SWAP(data[i3],data[i3rev]);
                  SWAP(data[i3+1],data[i3rev+1]);
                 }
               }
          }
          ibit=ip2 >> 1;
          while (ibit >= ip1 && i2rev > ibit)
            {
             i2rev -= ibit;
             ibit >>= 1;
            }
          i2rev += ibit;
        }
    ifp1=ip1;
    while (ifp1 < ip2)
        {
          ifp2=ifp1 << 1;
          theta=isign*6.28318530717959/(ifp2/ip1);
           wtemp=sin(0.5*theta);
           wpr = -2.0*wtemp*wtemp;
           wpi=sin(theta);
           wr=1.0;
           wi=0.0;
          for (i3=1;i3<=ifp1;i3+=ip1)
            {
              for (i1=i3;i1<=i3+ip1-2;i1+=2)
                  {
                    for (i2=i1;i2<=ip3;i2+=ifp2)
                     {
                       k1=i2;
                       k2=k1+ifp1;
                        tempr=wr*data[k2]-wi*data[k2+1];
                        tempi=wr*data[k2+1]+wi*data[k2];
                       data[k2]=data[k1]-tempr;
                       data[k2+1]=data[k1+1]-tempi;
                       data[k1] += tempr;
                       data[k1+1] += tempi;
                     }
                  }
                wr=(wtemp=wr)*wpr-wi*wpi+wr;
                wi=wi*wpr+wtemp*wpi+wi;
            }
           ifp1=ifp2;
         }
     nprev *= n;
   }
  if (isign == 1)
    for (idim=1;idim<=(2*ntot);idim+=2)
     {
         data[idim]=(double)data[idim]/ntot;
         data[idim+1]=(double)data[idim+1]/ntot;
     }
}
```

```
/****************************************************/
/* Function : ucps(inimg,oimg)                      */
/*          uncompress image                        */
/* inimg : input image with character type          */
/* oimg  : output image with integer type           */
/****************************************************/

void  ucps(inimg,oimg)
     char *inimg;
      int oimg[IMG_SIZE][IMG_SIZE];
{
  int i,j,idx,t,pix[4];

  for (j=0,idx=0;j<=(IMG_SIZE-1);j++)
    for (i=0;i<=(IMG_SIZE-1);i++,idx+=4)
     {
         for (t=1;t<=3;t++)
          {
            pix[t]=(int)inimg[idx+t];
           if (pix[t]<0)
             pix[t]=256+pix[t];
          }
         oimg[j][i]=(pix[1]+pix[2]+pix[3])/3;
     }
}

/****************************************************/
/* Function : cps(inimg,oimg)                       */
/*          compress image                          */
/* inimg : input image with float type              */
/* oimg  : output image with character type         */
/****************************************************/

void  cps(inimg,oimg)
      char oimg[SPM_SIZE][SPM_SIZE];
     float *inimg;
{
  int i,j,idx;
  float max,min;

  for (j=0,max=0.0,min=999.9,idx=1;j<=(SPM_SIZE-1);j++)
    for (i=0;i<=(SPM_SIZE-1);i++,idx+=2)
     {
         if (inimg[idx]>0)
           inimg[idx] = 0.0;
         else
           inimg[idx] = -inimg[idx];
         if (inimg[idx]>max)
            max=inimg[idx];
```

```
        if (inimg[idx]<min)
          min=inimg[idx];
    }
  max-=min;
  for (j=0,idx=1;j<=(SPM_SIZE-1);j++)
    for (i=0;i<=(SPM_SIZE-1);i++,idx+=2)
      oimg[j][i]=(char)(255*(inimg[idx]-min)/max);
  /* Draw scales on output image*/
  for (i=2;i<=(SPM_SIZE-1);i+=5)
    for (j=0;j<=2;j++)
    {
        oimg[j][i]=255;
        oimg[i][j]=255;
    }
  for (i=2;i<=(SPM_SIZE-1);i+=10)
    for (j=3;j<=4;j++)
    {
        oimg[j][i]=255;
        oimg[i][j]=255;
    }
}

/*****************************************************/
/* Blur identification program                      */
/*****************************************************/

main(argc,argv)
    int argc;
    char *argv[];
{
  int i, j, x, y,idn,idx,s2size,dim[3],stn,i2size;
   int gimg[IMG_SIZE][IMG_SIZE],spnum,avsize,xsize,ysize;
   char filename[50],pname[50],chtr,*sptr,pimg[SPM_SIZE][SPM_SIZE];
  char *avsimg;
  double *totspm;
  float *ftspm;
  FILE *gfp,*ffp;

  if(argc!=2)
   {
     printf("Blur identification program\n");
     printf("Usage: biden [input-file]\n");
    exit(1);
   }
  i2size = IMG_SIZE*IMG_SIZE;
  s2size = SPM_SIZE*SPM_SIZE;
  avsize = 4*i2size;
  idn = s2size*2;
  ftspm = (float *)calloc(idn+1,sizeof(float));
```

```
totspm = (double *)calloc(idn+1,sizeof(double));
 avsimg = (char*)calloc(avsize,sizeof(char));
 if((!ftspm)||(!totspm)||(!avsimg))
  {
    printf("memory request failed\n");
   exit(1);
  }
 gfp = fopen(argv[1],"rb");
if (!gfp)
  {
    printf("Error in reading file %s \n",argv[1]);
   exit(1);
  }
if((fread (&xsize, sizeof(int),1,gfp)!=1)||
   (fread (&ysize, sizeof(int),1,gfp)!=1))
  {
    printf("Output file error \n");
   exit(1);
  }
 if ((xsize!=IMG_SIZE)||(ysize!=IMG_SIZE))
  {
    printf("Image size should be 256x256\n");
   exit(1);
  }
 if(fread (avsimg, sizeof(char),avsize,gfp)!=avsize)
   {
     printf("Input file error! \n");
    exit(1);
   }
  ucps(avsimg,gimg);             /* image uncompress */
/* Initialize power spectrum */
for (idx=1;idx<=idn;idx+=2)
  {
   totspm[idx] = 0.0;
   totspm[idx+1] = 0.0;
  }
dim[1] = SPM_SIZE;
dim[2] = SPM_SIZE;
stn = IMG_SIZE - SPM_SIZE;
spnum = stn/STX + 1;
 spnum *= spnum;
 printf("Working...! \n");
for (y=0;y<=stn;y+=STX)
  for (x=0;x<=stn;x+=STX)
   {
      for (j=0,idx=1;j<=(SPM_SIZE-1); j++)
        for (i=0; i<=(SPM_SIZE-1); i++,idx+=2)
         {
            ftspm[idx]=WINDOW(i)*WINDOW(j)*gimg[y+j][x+i];
```

```
              ftspm[idx+1] = 0.0;
            }
        fourn(ftspm,dim,2,-1);
        for (idx=1; idx<=idn; idx+=2)
           totspm[idx]+=(MAGSQ(ftspm[idx],ftspm[idx+1])/(double)spnum);
   }
for (y=0,idx = 1;y<=(SPM_SIZE-1);y++)
  for (x=0;x<=(SPM_SIZE-1);x++,idx+=2)
   {
        ftspm[idx] = log(totspm[idx]+1);
        if (((x+y)%2)!=0)
          ftspm[idx]*=-1.0;
        ftspm[idx+1] = 0.0;
   }
 fourn(ftspm,dim,2,1);
 printf("\n");
 sptr = strchr(argv[1],'.');
if (sptr)
  *sptr = '\0';
 strcat(argv[1],"_id.x");
 printf("Output blur signature file name : %s\n",argv[1]);
 cps(ftspm,pimg);
 ffp = fopen(argv[1],"wb");
 if (!ffp)
  {
    printf("Error in writing file %s \n",argv[1]);
    exit(1);
  }
 free(avsimg);
avsize = 4*s2size;
 avsimg = (char*)calloc(avsize,sizeof(char));
 if(!avsimg)
  {
    printf("memory request failed\n");
    exit(1);
  }
for (j=0,idx=0; j<=(SPM_SIZE-1); j++)
   for (i=0; i<=(SPM_SIZE-1);i++,idx+=4)
    {
         avsimg[idx]=255;
         avsimg[idx+1]=pimg[j][i];
         avsimg[idx+2]=pimg[j][i];
         avsimg[idx+3]=pimg[j][i];
    }
xsize=SPM_SIZE;
ysize=SPM_SIZE;
if((fwrite (&xsize, sizeof(int),1,ffp)!=1)||
  (fwrite (&ysize, sizeof(int),1,ffp)!=1))
 {
    printf("Output file error \n");
    exit(1);
  }
 if(fwrite (avsimg, sizeof(char),avsize,ffp)!=avsize)
  {
     printf("Output file error \n");
     exit(1);
   }
 printf("Program completed!\n");
 free(totspm);
 free(ftspm);
 free(avsimg);
 fclose(gfp);
 fclose(ffp);
}
```

Program 3: wiener.c

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>

#define IMG_SIZE 256
#define PI 3.141592654
#define MAGSQ(a,b) ((double)(a)*(double)(a)+(double)(b)*(double)(b))
#define HANWIN(a) 0.5*(1.0-cos(2*PI*(double)(a)/(IMG_SIZE-1)))
/*Hanning*/
#define HAMWIN(a) (54-46*cos(2*PI*(double)(a)/(IMG_SIZE-1)))/100
/*Hamming*/

#define SWAP(a,b) tempr=(a);(a)=(b);(b)=tempr

 /*****************************************************/
 /* Function : fourn(data,nn,ndim,isign)               */
 /*          N-dimension Fourier Transform             */
 /* data : input complex number data which stored in   */
 /*       row major format.                            */
 /*       Data started with index at 1, stored with    */
 /*       real number follow by imagine number         */
 /* nn   : number of dimension                         */
 /* ndim : store the length of each dimension, started */
 /*       with index 1                                 */
 /* isign: -1 for forward transform                    */
 /*       1 for inverse transform                      */
 /*****************************************************/

void fourn(data,nn,ndim,isign)
```

```
    float data[];
     int nn[],ndim,isign;
{
  int i1,i2,i3,i2rev,i3rev,ip1,ip2,ip3,ifp1,ifp2;
  int ibit,idim,k1,k2,n,nprev,nrem,ntot;
  float tempi,tempr;

   double theta,wi,wpi,wpr,wr,wtemp;

  ntot=1;
   for (idim=1;idim<=ndim;idim++)
    ntot *= nn[idim];
  nprev=1;
   for (idim=ndim;idim>=1;idim--)
    {
      n=nn[idim];
      nrem=ntot/(n*nprev);
     ip1=nprev << 1;
     ip2=ip1*n;
     ip3=ip2*nrem;
     i2rev=1;
     for (i2=1;i2<=ip2;i2+=ip1)
          {
           if (i2 < i2rev)
            {
              for (i1=i2;i1<=i2+ip1-2;i1+=2)
                   {
                    for (i3=i1;i3<=ip3;i3+=ip2)
                      {
                       i3rev=i2rev+i3-i2;
                       SWAP(data[i3],data[i3rev]);
                       SWAP(data[i3+1],data[i3rev+1]);
                      }
                   }
            }
           ibit=ip2 >> 1;
           while (ibit >= ip1 && i2rev > ibit)
            {
              i2rev -= ibit;
              ibit >>= 1;
            }
           i2rev += ibit;
          }
     ifp1=ip1;
     while (ifp1 < ip2)
          {
           ifp2=ifp1 << 1;
           theta=isign*6.28318530717959/(ifp2/ip1);
            wtemp=sin(0.5*theta);
          wpr = -2.0*wtemp*wtemp;
          wpi=sin(theta);
          wr=1.0;
          wi=0.0;
          for (i3=1;i3<=ifp1;i3+=ip1)
           {
             for (i1=i3;i1<=i3+ip1-2;i1+=2)
                 {
                   for (i2=i1;i2<=ip3;i2+=ifp2)
                     {
                       k1=i2;
                       k2=k1+ifp1;
                       tempr=wr*data[k2]-wi*data[k2+1];
                       tempi=wr*data[k2+1]+wi*data[k2];
                       data[k2]=data[k1]-tempr;
                       data[k2+1]=data[k1+1]-tempi;
                       data[k1] += tempr;
                       data[k1+1] += tempi;
                     }
                 }
               wr=(wtemp=wr)*wpr-wi*wpi+wr;
               wi=wi*wpr+wtemp*wpi+wi;
           }
          ifp1=ifp2;
        }
     nprev *= n;
   }
  if (isign == 1)
    for (idim=1;idim<=(2*ntot);idim+=2)
     {
        data[idim]=(double)data[idim]/ntot;
        data[idim+1]=(double)data[idim+1]/ntot;
     }
}

/****************************************************/
/* Function : rnd(num)                              */
/*          return the round value of num           */
/* num : number to be rounded                       */
/****************************************************/

int  rnd(num)
    double num;
{
  int i;
  double f;

  i=(int)num;
  f=num-i;
```

```
 if (f>=0.5)

   return (i+1);
 else
   return i;
}

/*****************************************************/
/* Function : ucps(inimg,oimg)                       */
/*         uncompress image                          */
/* inimg : input image with character type           */
/* oimg  : output image with integer type            */
/*****************************************************/

void  ucps(inimg,oimg)
     char *inimg;
     int oimg[IMG_SIZE][IMG_SIZE];
{
  int i,j,idx,t,pix[4];

  for (j=0,idx=0;j<=(IMG_SIZE-1);j++)
    for (i=0;i<=(IMG_SIZE-1);i++,idx+=4)
    {
        for (t=1;t<=3;t++)
         {
           pix[t]=(int)inimg[idx+t];
          if (pix[t]<0)
            pix[t]=256+pix[t];
         }
        oimg[j][i]=(pix[1]+pix[2]+pix[3])/3;
    }
}

/*****************************************************/
/* Function : cps(inimg,oimg)                        */
/*         compress image                            */
/* inimg : input image with integer type             */
/* oimg  : output image with character type          */
/*****************************************************/

void  cps(inimg,oimg)
      char oimg[IMG_SIZE][IMG_SIZE];
      int inimg[IMG_SIZE][IMG_SIZE];
{
  int i,j,max,min;

  for (j=0,max=0,min=256;j<=(IMG_SIZE-1);j++)
    for (i=0;i<=(IMG_SIZE-1);i++)
    {
        if (inimg[j][i]<0)
          inimg[j][i]=0;
        if (inimg[j][i]>max)
          max=inimg[j][i];
        if (inimg[j][i]<min)
          min=inimg[j][i];
    }
  if (max>255)
   {
     max-=min;
     for (j=0;j<=(IMG_SIZE-1);j++)
        for (i=0;i<=(IMG_SIZE-1);i++)
          inimg[j][i]=(255*(inimg[j][i]-min))/max;
   }
  for (j=0;j<=(IMG_SIZE-1);j++)
    for (i=0;i<=(IMG_SIZE-1);i++)
      oimg[j][i]=(char)inimg[j][i];
}

/*****************************************************/
/* Function : bpsf(psft)                             */
/*         calculate the PSF                         */
/* psft : return Fourier Transform of the PSF        */
/*****************************************************/

void bpsf(psft)
    float *psft;
{
  int i,j,idx,r,diam,ndim[3],d,ang;
  char ch;
  float h,d2;
  double w;

  printf("\n");
  printf("1.Defocus restoration\n");
  printf("2.Motion restoration\n");
  printf("Enter number of selection: ");
  do
   {
     ch = getchar();
     switch(ch)
        {
        case '1':
          printf("\nPlease enter R : ");
           scanf("%d",&r);
          diam = r*r;
          h = 1.0/(PI*diam);
          for (j=0,idx = 1; j<=(IMG_SIZE-1); j++)
            for (i=0; i<=(IMG_SIZE-1); i++,idx+=2)
          {
              if ((MAGSQ(i,j)<=diam)||
                  (MAGSQ((IMG_SIZE-i-1),j)<=diam)||
                  (MAGSQ(IMG_SIZE-j-1,i)<=diam)
                   ||(MAGSQ(IMG_SIZE-i-1,IMG_SIZE-j-1)<=diam))
                psft[idx] = h;
              else
                psft[idx] = 0.0;
              psft[idx+1]=0.0;
          }
        break;
      case '2':
        printf("\nPlease enter d : ");
         scanf("%d",&d);
        printf("\nPlease enter angle : ");
         scanf("%d",&ang);
        d2=rnd(d/2.0);
        h=1.0/d;
        for (j=0,idx = 1; j<=(IMG_SIZE-1); j++)
          for (i=0; i<=(IMG_SIZE-1); i++,idx+=2)
          {
              psft[idx]=0.0;
              psft[idx+1]=0.0;
          }
        for (idx=0;idx<d2;idx++)
         {
            i=rnd(idx*cos(PI*ang/180));
            j=rnd(idx*sin(PI*ang/180));
            psft[(j*IMG_SIZE+i)*2+1]=h;

          if (ang==0)
              psft[(j*IMG_SIZE+IMG_SIZE-1-i)*2+1]=h;
          else
              psft[((IMG_SIZE-1-j)*IMG_SIZE+IMG_SIZE-1-i)*2+1]=h;
         }
        break;
      }
  }
  while (ch!='1' && ch!='2');
  ndim[1]=IMG_SIZE;
  ndim[2]=IMG_SIZE;
  fourn(psft,ndim,2,-1);
  printf("\n");
}

main(argc,argv)
    int argc;
    char *argv[];
{
 int i, j, x, y,idn,idx,dim[3],i2size,avsize,xsize,ysize;
  int gimg[IMG_SIZE][IMG_SIZE], fimg[IMG_SIZE][IMG_SIZE];
  char fnm[51],hfnm[51],*sptr,*avsimg,pimg[IMG_SIZE][IMG_SIZE],ch;
 double hpsm,hr,hi;
float *gft,*hft,*fft,k;
FILE *gfp,*hfp,*ffp;

if(argc!=2)
{
   printf("Wiener filter restoration");
   printf("Usage: wier [input-file]\n");
  exit(1);
}
 i2size = IMG_SIZE*IMG_SIZE;
idn = i2size*2;
avsize = 4*i2size;
gft = (float *)calloc(idn+1,sizeof(float));
hft = (float *)calloc(idn+1,sizeof(float));
fft = (float *)calloc(idn+1,sizeof(float));
 avsimg = (char*)calloc(avsize,sizeof(char));
 if((!gft)||(!hft)||(!fft)||(!avsimg))
  {
    printf("memory request failed\n");
   exit(1);
  }
/* read avs file format */
 gfp = fopen(argv[1],"rb");
if (!gfp)
  {
    printf("Error in reading file %s \n",argv[1]);
   exit(1);
  }
 if((fread (&xsize, sizeof(int),1,gfp)!=1)||
   (fread (&ysize, sizeof(int),1,gfp)!=1))
  {
    printf("Output file error \n");
   exit(1);
  }
  if ((xsize!=IMG_SIZE)||(ysize!=IMG_SIZE))
  {
    printf("Image size should be 256x256\n");
   exit(1);
  }
  if(fread (avsimg, sizeof(char),avsize,gfp)!=avsize)
  {
    printf("Input file error! \n");
   exit(1);
  }
  ucps(avsimg,gimg);              /* image uncompress */
 printf("Please enter K :\n");
  scanf("%f",&k);
  printf("\n");
 bpsf(hft);
  dim[1] = IMG_SIZE;
  dim[2] = IMG_SIZE;
  printf("\n");
   printf("Working...!\n");

 for (j=0,idx = 1; j<=(IMG_SIZE-1); j++)
   for (i=0; i<=(IMG_SIZE-1); i++,idx+=2)
    {
         gft[idx]=gimg[j][i];
         gft[idx+1] = 0.0;
    }
  fourn(gft,dim,2,-1);

 /* Wiener Filter */
 for (y=0,idx=1;y<=(IMG_SIZE-1);y++)
   for (x=0;x<=(IMG_SIZE-1);x++,idx+=2)
    {
         hpsm=MAGSQ(hft[idx],hft[idx+1]);
         hr=hft[idx]/(hpsm+k);
         hi=-hft[idx+1]/(hpsm+k);
         fft[idx]=gft[idx]*hr-gft[idx+1]*hi;
         fft[idx+1]=gft[idx+1]*hr+gft[idx]*hi;
    }
  fourn(fft,dim,2,1);
 for (y=0,idx=1;y<=(IMG_SIZE-1);y++)
   for (x=0;x<=(IMG_SIZE-1);x++,idx+=2)
    {
         fimg[y][x] = fft[idx];
    }
  printf("\n");
  printf("Please enter output file name:\n");
   scanf("%50s",fnm);
  ffp = fopen(fnm,"wb");
 if (!ffp)
  {
     printf("Error in writing file %s \n",fnm);
    exit(1);
 }
  cps(fimg,pimg);
 /* translate into AVS format */
 for (j=0,idx=0; j<=(IMG_SIZE-1); j++)
   for (i=0; i<=(IMG_SIZE-1);i++,idx+=4)
    {
         avsimg[idx]=255;
         avsimg[idx+1]=pimg[j][i];
```

```
          avsimg[idx+2]=pimg[j][i];
          avsimg[idx+3]=pimg[j][i];
      }
   if((fwrite (&xsize, sizeof(int),1,ffp)!=1)||
     (fwrite (&ysize, sizeof(int),1,ffp)!=1))
    {
      printf("Output file error \n");
      exit(1);
    }
   if(fwrite (avsimg, sizeof(char),avsize,ffp)!=avsize)
    {
      printf("Output file error \n");
      exit(1);
    }
   printf("Program completed!\n");
  free(fft);
  free(gft);
  free(hft);
  free(avsimg);
  fclose(gfp);
  fclose(hfp);
  fclose(ffp);
}
```

REFERENCES

1. Bekker, A. Y., A. B. Ritter and W. N. Duran. "Reduction of pressure in post-capillary venues induced by EPI-fluorescent illumination of FITC-Dextran." *Microcirc. Endoth. Lymphatics*, 3, 1978,pp. 411-423.

2. Bekker, A. Y., A. B. Ritter and W. N. Duran. "Analysis of microvascular permeability to macromolecules by video digital processing." *Microvas. Res.* 38, 1989, pp. 200-216.

3. Hellberg, K. H., H. Wayland and A. L. Bing. "Studies on the coronary microcirculation by direct visualization." *Am. J. Cardiol* 29, 1972, pp. 539-597.

4. Ritter, A. B., W. Braun, A. Stein and W. Duran. "Visualization of the coronary microcirculation using digital image processing." *Comput. Biol. Med.* 15, 1985, pp. 361-374.

5. Sen, D. "Digital image restoration using power spectrum equalization filtering technique." Thesis, New Jersey Institute of Technology, Newark, NJ, 1989.

6. Wang, Kaijun. "Image deblurring for visualization of coronary microcirculation." Thesis, New Jersey Institute of Technology, Newark, NJ, 1991.

7. Bergmann, S. R., R. E. Clark and B. E. Sobel. "An improved isolated heart preparation for external assessment of myocardial metabolism." *Am. J. Physiol.* 236, 1976, pp. 644-651.

8. Braun, W. "An experimental system for observation of the rat coronary microcirculation using digital image processing." Thesis, New Jersey Institute of Technology, Newark, NJ, 1983.

9. Andrews, H. C. and B. R. Hunt. *Digital Image Restoration*. Prentice-Hall, Inc, 1977.

10. Sezan, I. M. and A. M. Tekalp. "Survey of recent developments in digital image restoration." *Optical Engineering* 29, 1990,pp. 393-404.

11. *Application Visualization System User's Guide*. Advanced Visual System, inc., 1992.

12. Castleman, R. *Digital Image Processing*. Prentice-Hall, Inc., 1979.

13. Rosenfeld, A. and A. C. Kak. *Digital Picture Processing*. Academic Press, New York, 1982.

14. Chalmond, B. "PSF estimation for image deblurring." *CVGIP: Graphical Model and Image Processing* 53, 1991,pp. 364-372.

15. Cannon, M. "Blind deconvolution of spatially invariant image blurs with phase." *IEEE Trans. Acoust. Speech Signal Precess.* 24, 1976, pp. 58-63.

16 Stockham T. G., T. M. Cannon and R. B. Ingebresten. "Blind deconvolution through digital signal processing." *Proc. IEEE* 63, 1975, pp. 678-692.

17 Cannon, T. M. "Digital image deblurring by non-linear homomorphic filtering." Ph.D. dissertation, University of Utah, 1974.

18. Fabian, R. and D. Malah. "Robust identification of motion and out-of-focus blur parameters from blurred and noisy images." *CVGIP: Graphical Model and Image Processing* 53, 1991,pp. 403-412.

19. Ghiglia, D. C. and C. V. Jakowatz. "Some practical aspects of moving object deblurring in a perspective plane." *Applied Optics* 24, 1985, pp. 3830-3837.

20. Welch, P. D. "The use of the FFT for the estimation of power spectra." *IEEE Trans. Audio Electroacoust.* 15, 1967, pp. 70-73.

21. Rom, R. "On the cepstrum of two-dimensional functions." *IEEE Tran. of Info. Theory*, March, 1975, pp. 214-217.

22. Cannon, M., H. J. Trussell and B. R. Hunt. "Comparison of image restoration methods." *Appl. Opt.* 17, 1978, pp. 3384-3390.