AFIT/GEO/ENG/91D-3

OPTICAL IMAGE SEGMENTATION USING WAVELET CORRELATION

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

Steven D. Pinski, B.S.E.E.
Captain

December, 1991

Approved for public release; distribution unlimited

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE December 1991	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE OPTICAL IMAGE SEGMENTATION USING WAVELET CORRELATION	5. FUNDING NUMBERS
6. AUTHOR(S) Steven D. Pinski, Captain, USAF	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, WPAFB OH 45433-6583	8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GEO/ENG/91D-3
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This research introduces an optical method of segmenting potential targets using wavelet analysis. Implementation of an optical Harr wavelet is fulfilled using a magneto-optic spatial light modulator (MOSLM). Two methods of controlling wavelet dilation are explored: 1) spatial filtering of a ternary modulated MOSLM; 2). a single aperture positioned in front of a binary modulated MOSLM. Segmentation is performed through Vander Lugt correlation of a binarized image with a binarized optical wavelet. Three different image binarization methods are investigated for use in the correlation scheme: 1) average pixel value over an entire scene; 2) localized 4×4 pixel average followed by an average pixel value over the remaining scene; 3) localized 3×3 pixel average "ANDed" with an average pixel value over the entire scene. Frequency-plane masks necessary for the correlation process are generated using thermal holography. Results show image segmentation for six possible experimental methods comprised of combinations of wavelet dilation and binarization techniques. The most successful correlation design used a single aperture to control wavelet dilation and binarization based on a localized 4×4 pixel average followed by an average pixel value over the remaining scene.

14. SUBJECT TERMS Wavelets, Multiresolution Analysis, Optical Segmentation, Optical Correlation	15. NUMBER OF PAGES 113
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements.***

Block 1. Agency Use Only *(Leave blank)*

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	-	Contract	PR	-	Project
G	-	Grant	TA	-	Task
PE	-	Program Element	WU	-	Work Unit Accession No

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

DOD	-	See DoDD 5230.24, "Distribution Statements on Technical Documents."
DOE	-	See authorities.
NASA	-	See Handbook NHB 2200.2.
NTIS	-	Leave blank.

Block 12b. Distribution Code.

DOD	-	Leave blank.
DOE	-	Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
NASA	-	Leave blank.
NTIS	-	Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

Preface

I'm not sure if I was suppose to have a good time working on this research project, but I can honestly say I did have a lot of fun. All the frustrating moments caused by hardware malfunctions were quickly forgotten when successful and exciting new progress was made.

Thanks goes to my thesis committee members, Dr. Byron Welsh, Dr. Dennis Ruck, and Dr. Matthew Kabrisky (Mr. Video), for their support throughout this research project. Moreover, I am very proud to have worked with my thesis advisor, Dr. Steven Rogers, who helped me learn a great deal here at AFIT. Thanks also goes to Ken Zabel who helped me with the nasty binarization problems.

I thank my parents; they were a great inspiration for me. Most of all, my wife Linda and our children, Sondra and Kevin, made this AFIT experience bearable. Thank you Linda for typing my thesis. For all your sacrifices, I dedicate this thesis to all of you.

Steven D. Pinski

Table of Contents

List of Figures

AFIT/GEO/ENG/91D-3

Abstract

This research introduces an optical method of segmenting potential targets using wavelet analysis. Implementation of an optical Harr wavelet is fulfilled using a magneto-optic spatial light modulator (MOSLM). Two methods of controlling wavelet dilation are explored: 1) spatial filtering of a ternary modulated MOSLM; 2). a single aperture positioned in front of a binary modulated MOSLM.

Segmentation is performed through Vander Lugt correlation of a binarized image with a binarized optical wavelet. Three different image binarization methods are investigated for use in the correlation scheme: 1) average pixel value over an entire scene; 2) localized 4×4 pixel average followed by an average pixel value over the remaining scene; 3) localized 3×3 pixel average "ANDed" with an average pixel value over the entire scene. Frequency-plane masks necessary for the correlation process are generated using thermal holography.

Results show image segmentation for six possible experimental methods comprised of combinations of wavelet dilation and binarization techniques. The most successful correlation design used a single aperture to control wavelet dilation and binarization based on a localized 4×4 pixel average followed by an average pixel value over the remaining scene.

OPTICAL IMAGE SEGMENTATION USING WAVELET CORRELATION

I. Introduction

Researchers have spent many years attempting to develop an automatic machine-based pattern recognition system. Moreover, AFIT has studied the pattern recognition process in detail while attempting to recognize targets in a scene (4, 32, 33, 7, 8, 22, 24, 27). The pattern recognition process consists of three stages: image segmentation, feature extraction and classification. Image segmentation is simply removing potential targets from a background. Feature extraction refers to the important object features retained for further processing. Finally, classification is identifying segmented objects using the important features extracted earlier (21).

Optical image segmentation has the unique advantage of speed over digital processing, that is, the speed of light. An optical image segmentation scheme has recently been implemented using spatial filtering techniques (32); however, this thesis takes a different approach.

1.1 Problem Statement

This research introduces an optical wavelet processing technique for separating objects of interest (potential targets) from the background in a cluttered image.

1.2 Scope

The primary goal of this research is image segmentation. Results will reflect six basic methods used to perform this segmentation. A combination of three image binarization techniques and two wavelet implementation techniques comprise the six methods. Image binarization techniques include: 1) average pixel value over an entire scene; 2) localized 4×4 pixel average followed by an average pixel value over the remaining scene; 3)localized 3×3 pixel average "ANDed" with the average pixel value over an entire scene. Optical wavelet implementation was performed using two methods of controlling wavelet dilation: a single aperture and spatial filtering. This research does not attempt to perform feature extraction or classification of the segmented objects.

1.3 Approach

A block diagram is shown in Figure 1 detailing the various steps involved in this method of image segmentation. First, the image to be segmented was binarized using one of three techniques mentioned in the previous section. Next, the Fourier transform of the binarized image was recorded as a frequency-plane mask using thermal holography. Now, the Fourier transform of a binary or ternary wavelet is aligned with the frequency-plane mask recorded earlier. Optical correlation of the binarized image and wavelet will produce segmentation.

1.4 Outline of Thesis

Chapter II provides background on several different segmentation algorithms. The various approaches employed in these articles provide a sample of current image segmentation research. Chapter III discusses the methodology used in this research to obtain segmentation results. Necessary equipment and technologies used are outlined; moreover, schematics of correlation setups are explained. Chapter IV presents results obtained during this research. Separate discussions for the different methods of segmentation are offered while examining characteristics of each. Finally, Chapter V presents conclusions and recommendations for future research in this area.

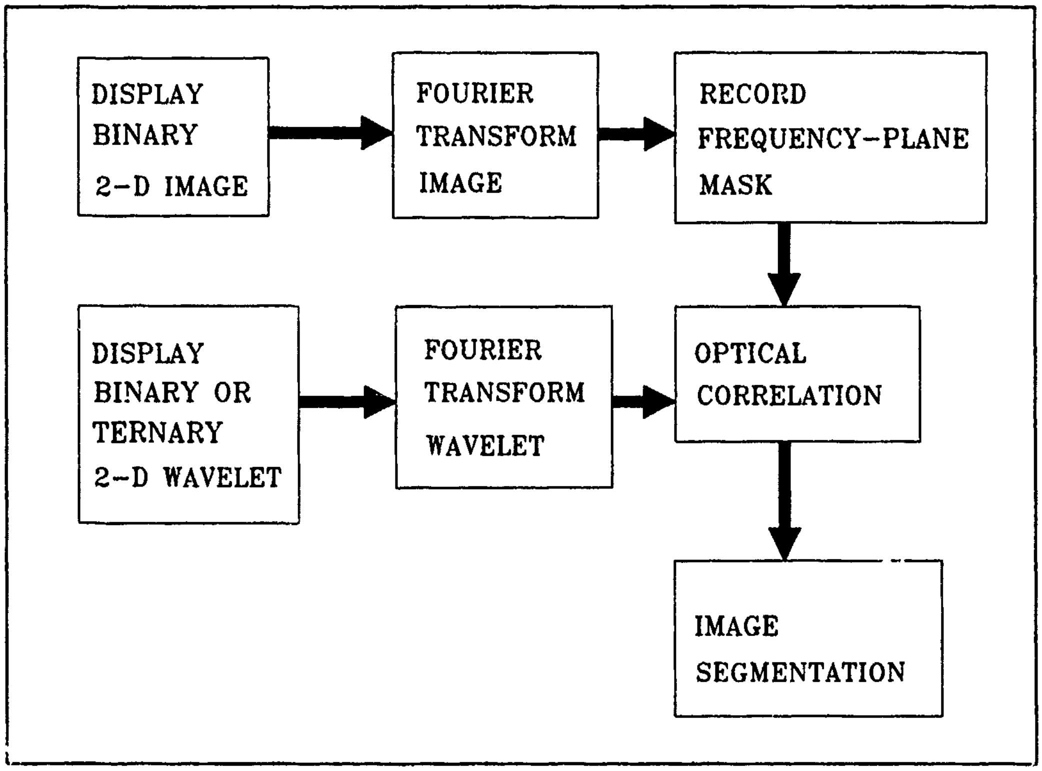

DISPLAY BINARY 2-D IMAGE
FOURIER TRANSFORM IMAGE
RECORD FREQUENCY-PLANE MASK
DISPLAY BINARY OR TERNARY 2-D WAVELET
FOURIER TRANSFORM WAVELET
OPTICAL CORRELATION
IMAGE SEGMENTATION

Figure 1. Optical wavelet segmentation approach

II. Segmentation Background

The initial problem encountered in the pattern recognition process is the extraction of objects from an image. Any method used to perform this extraction of objects (potential targets) is referred to as segmentation. A variety of filtering techniques, most of them digital, have been developed to perform image segmentation.

Segmentation of a battlefield image for target tracking is of primary concern to the Air Force; indeed, the entire Department of Defense is working on this problem. The next generation of "smart" bombs may utilize image segmentation algorithms to find targets autonomously in a complicated scene. Currently, pilots use lasers to designate targets for "smart" bombs. This laser designation precludes "launch and leave" operation by the pilot. As the pilot loiters above the battlefield waiting for his bomb to hit the designated enemy target, his plane becomes a target itself. Many pilots' lives could be saved if their "smart" weapons utilized automatic target recognition systems; in that case, pilots would no longer be required to hover over hostile areas while lasing targets.

The articles summarized in this chapter represent examples of image segmentation techniques which are real-time or near real-time solutions for the pattern recognition problem. These examples will illustrate the direction image segmentation research is heading.

2.1 Model Based

In their 1990 article, Bhanu and Holben described model-based image segmentation based on intensity and edge-based information. Forward looking infrared (FLIR) images were segmented more efficiently by using edge information combined with the intensity of the gray scale image, rather than the intensity or edge information alone. Histograms of the pixel intensity values were used to select the threshold for segmentation of target edges. The edge magnitude of pixels are compared in one small region at a time. If the majority of pixels in a particular region are low, then the region is considered to be

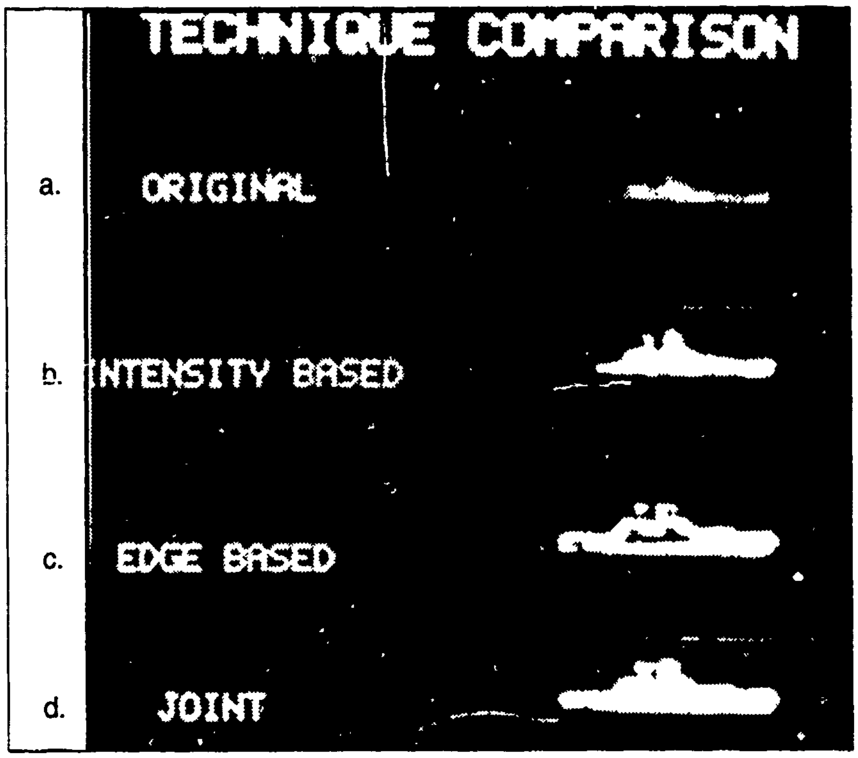


Figure 2. Segmentation technique comparison. a. Original FLIR image; b. Intensity-based segmentation; c. Edge-based segmentation; d. Joint relaxation segmentation. (5:9)

part of the background. Gray-scale and edge detection values are then used to "body fill" the target. This joint relaxation technique (Figure 2) using both intensity and edge based information segmented better than either technique used alone.

The model-based segmentation proposed is quite effective, especially in low contrast situations. Real-time segmentation of FLIR images using this approach have been demonstrated at TV frame rates.(5)

2.2 *Edge Detection and Subtraction*

In their 1990 article, Qui and Hartley, used edge detection to produce image segmentation of real-time, real-world situations. Window gradient masks co volved with a real world reference image and current image produced edge extraction. After developers completed convolution of the images with the gradient masks, a local adaptive threshold technique was used to create binary edge images. Subtracting the binary edge image of the reference scene from the binary edge image of the current scene produced the desired segmentation.

Changes in the ambient lighting of the reference and current image produce random noise pixels in the segmented image. However, contiguous edge pixels suppressed or eliminated random noise pixels through neighboring of the pixels in the subtracted image.

Qui and Hartley performed this real-time segmentation algorithm with limited computing power. The results shown in their article are useful for tracking purposes; nevertheless, this type of segmentation may not have enough resolution for post-processing classification.(20)

2.3 Digital Gabor Transforms

In his 1989 thesis, Ayer used Gabor transform computer algorithms to perform digital image segmentation of FLIR images. He performed energy normalization of the images using pixel brightness to prevent erroneous image processing. Following normalization, Ayer employed a fast Fourier transform algorithm to prepare the image for filtering. Sine and cosine Gabor functions filtered the transformed image.

Because Gabor transforms depend on frequency and orientation, to a large degree, Ayer selected spatial filters based on the target being detected. To this end, he computed orientations of 0, 45, 90 and 135 degrees separately and superimposed them on one filter.

Binarization of the FLIR images followed segmentation. Histogramming of the pixel brightness determined the binarization threshhold level for the filtered image. Ayer's simulations showed sine Gabor transforms to be "edge detectors" while cosine Gabor transforms acted as "body fillers." An example of these dramatic effects is shown in figure 3. Using superposition, Ayer utilized both sine and cosine Gabor transforms to obtain complete segmentation of FLIR images.(4)

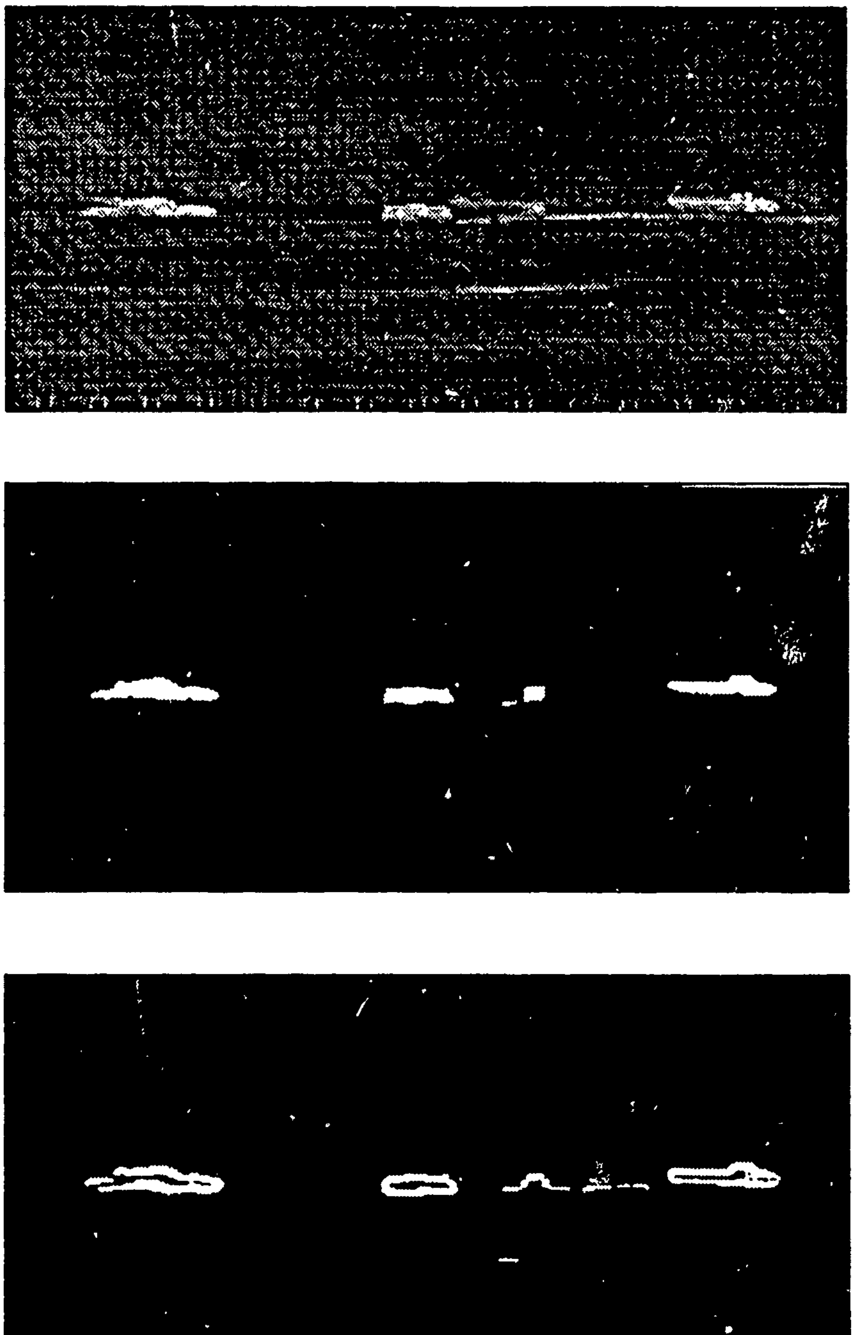


Figure 3. Examples of FLIR image segmentation using Gabor transforms. Orginal FLIR image (top); Binarized sine Gabor transform of top image showing edge detection (middle); Binarized cosine Gabor transform of top image showing body filling (bottom). (4:46,49)

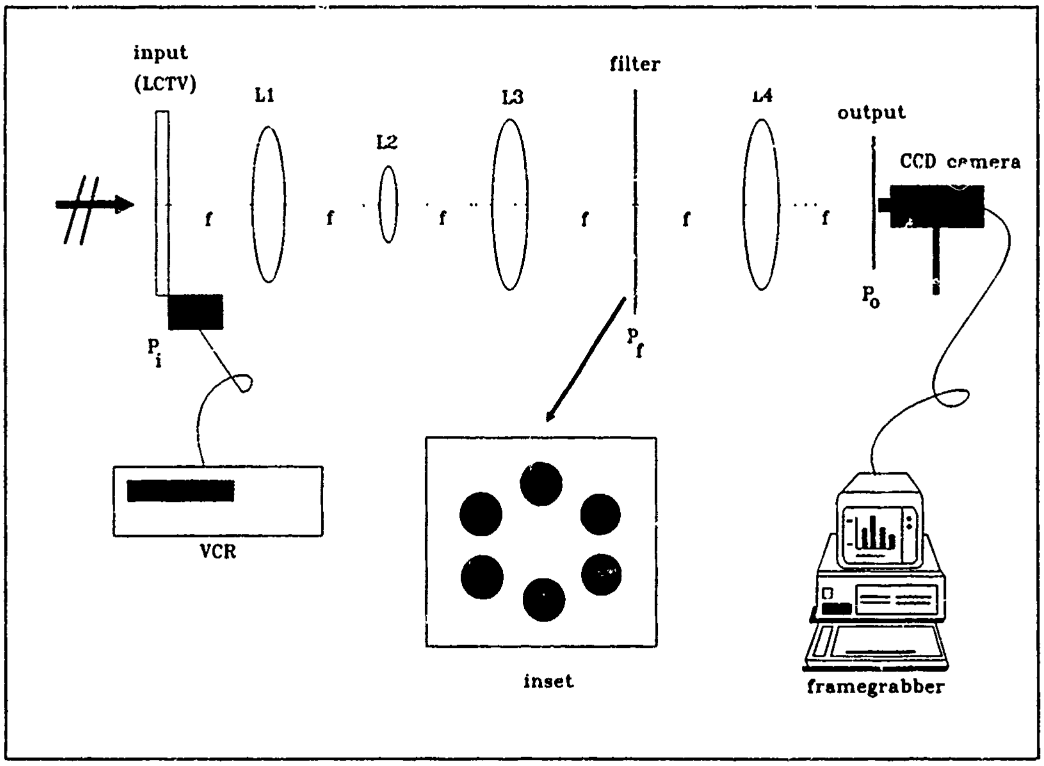


Figure 4. Set-up for optical image segmentation using spatial filtering (32:1-4).

2.4 Optical Segmentation

As a follow-on to Ayer's work, Veronin (32, 33) used coherent imaging and spatial filtering to optically segment an image. The optical set-up shown in Figure 4 produced instantaneous segmentation of real-time FLIR images. The liquid crystal television (LCTV) at input (P_i) displayed a VCR tape of real-time FLIR images. Lens L1 produces a two dimensional Fourier transform, at lens L2, of the image at P_i. Lens L2 and L3 magnify the two dimensional Fourier transform produced by lens L1. The resultant spatial filtering performed by the filter segments the image. The segmented image is again Fourier transformed by lens L4 and recorded by a charged coupled device (CCD) camera at the output (P_o).

Veronin implemented several filter designs in the frequency domain on an optical bench. These filters include: simple pinhole pairs drilled in aluminum, computer generated

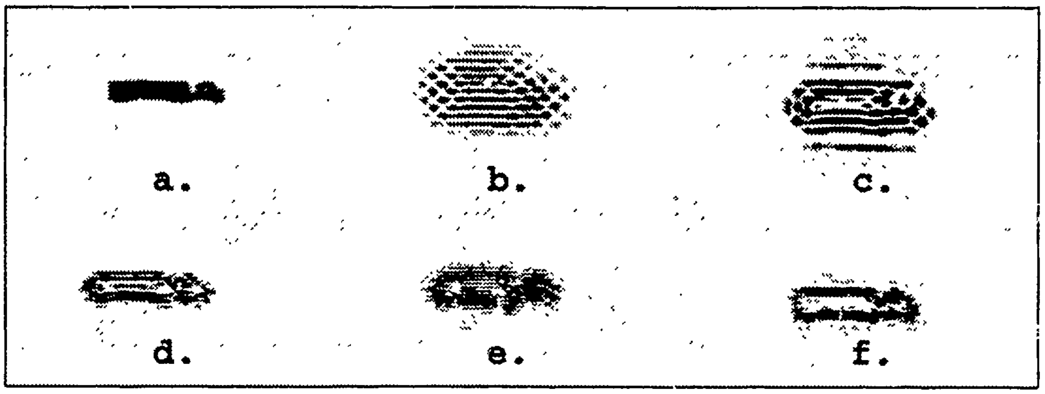


Figure 5. Truck segmentation using different pinhole filters. a. Original FLIR image; b. separations=2mm, dilations=.5mm, orientations=0 and 90°; c.s=2mm, d=1mm, o=0 and 90°; d. s=4mm, d=2mm, o=0 and 90°; e. s=6mm, d=2mm, o=0,45,90,and 135°; f. s=6mm,d-3mm,o=30,90,and 135° . (32:4-14)

holograms of sine and cosine Gabor filters, and pinhole pairs displayed on a LCTV. The pinhole pairs drilled in aluminum produced the best overall segmentation of the images. Segmentation of a FLIR image for various pinhole filters is shown in figure 5.

Optical segmentation has a distinct advantage over other methods in that the segmentation occurs at the speed of light. True real-time segmentation may require an effective optical scheme.(32)

2.5 Wavelets

Stephane Mallat, in his 1989 article describes image processing using wavelets. A wavelet can be thought of as a unique function which, through its translation and dilation, can decompose any signal. Resolution of the wavelet processed image will depend on the size of the wavelet performing the decomposition.

Normally, an image is made up of different size objects. To highlight these different size objects, Mallat used different resolutions (different size wavelets) to decompose the image.

> In some sense, the details of the image at a coarse resolution provide the "context" of the image, whereas the finer details correspond to the particular "modalities." For example, it is difficult to recognize that a small rectangle inside an image is the window of a house if we did not previously recognize the house "context." It is therefore natural to first analyze the image details at a coarse resolution and then increase the resolution. This is called a coarse-to-fine processing strategy.([illegible]

The computation time for fine detail processing is quite lengthy; therefore, Mallat recommends employing coarse details wherever possible

Some wavelets use a zero-crossing representation to locate signal edges making the wavelets ideal candidates for image segmentation. Optical implementation of these wavelet techniques may be the next step to better pattern recognition.(15)

2.6 Summary

Of the segmenting techniques described in this paper, only Veronin's optical set-up performed in true real-time. If we are to solve the preprocessing problem of image segmentation for an automatic real-time target recognition system, optical processing appears to be the best alternative. An optical image segmentation algorithm utilizing the wavelet transform explained in Mallat's article is the purpose of this thesis.

III. Methodology

This chapter presents the underlying technologies needed to implement an optical wavelet and perform segmentation of an image through correlation. Chapter organization is divided into three sections. First, wavelet analysis is discussed from the view of hardware implementation. This section briefly covers the background of wavelets, the hardware necessary to make a Harr wavelet, and the method of programming the wavelet onto the hardware. The second section covers optical correlation information. Included in this section are discussions of Vander Lugt filtering, thermoplastic holograms, and optical setups used. This section also covers the two methods of wavelet implementation used to perform the optical correlations. Finally, a brief summary will conclude the chapter. All segmentation results are discussed in Chapter IV.

3.1 Wavelet Analysis

3.1.1 Background Optical image segmentation has been explored from many different perspectives and image processing techniques. However, no one method stands out as a solution to the segmentation problem. Wavelet analysis is another method for multifrequency decomposition of an image to analyze its characteristics.

Wavelet decomposition of a signal is similar to that of a Fourier transform. A Fourier transform seeks to represent a given function as a combination of weighted sinusoidal components. These sinusoidal components are complex exponentials which form a complete orthogonal basis set. The Fourier transform can thus decompose an image from the space domain into a spatial-frequency domain. Once transformed, the image is viewed as sinusoidal components having harmonically related spatial frequencies. Unfortunately, the Fourier transform does not give any information about the location of the frequency components in a scene.

On the other hand, a wavelet decomposition does give information as to the location of frequencies. The wavelet transform decomposes an image using translations and dilations of a mother wavelet $\psi(x)$. These translations and dilations are known as the wavelet basis set. It is possible to perform multiscale edge detection optically using a Harr wavelet basis set. An orthogonal Harr wavelet is given by:

$$\psi(x) = \begin{cases} 1 & \text{if } 0 \leq x < .5 \\ -1 & \text{if } .5 \leq x < 1 \\ 0 & \text{otherwise} \end{cases}$$

Through optical correlation of a dilated and translated Harr wavelet with an image, edge detection will result. Correlation with a larger wavelet will yield coarse resolution of the scene, while correlation with a smaller wavelet will reveal details not detectable with a large wavelet. A Harr basis set was chosen due to its piecewise constant nature; a binary spatial light modulator can accommodate this type of signal. Detailed information about digital multifrequency channel decomposition of images using wavelets is available (15, 14, 26).

3.1.2 Magneto-optic Spatial Light Modulator (MOSLM) The MOSLM is an electrically addressable array of pixels which can be used in optical image processing systems. This device can randomly alter the magnetic state of individual pixels through the magneto-optic (Faraday) effect. Advantages of the MOSLM over other optical display systems include: nonvolatile display, fast switching times (100ns), and high contrasts. Array sizes of 48×48, 128×128 and 256×256 pixels are currently available, while array sizes of 512×512, 1024×1024 and 2048×2048 pixels are planned (9).

Operation of the MOSLM is shown in Figure 6. Polarization of light entering the device is magnetically rotated by electrical addressing of crossed electrodes at each pixel. Only one pixel is affected by this electrical addressing because of an ion implanted region located where the electrodes cross. This ion implanted region forces individual pixels to be highly susceptible to the Faraday effect. Once the pixels have been addressed (nucleated) they are fully rotated by applying a uniform magnetic saturation pulse from a

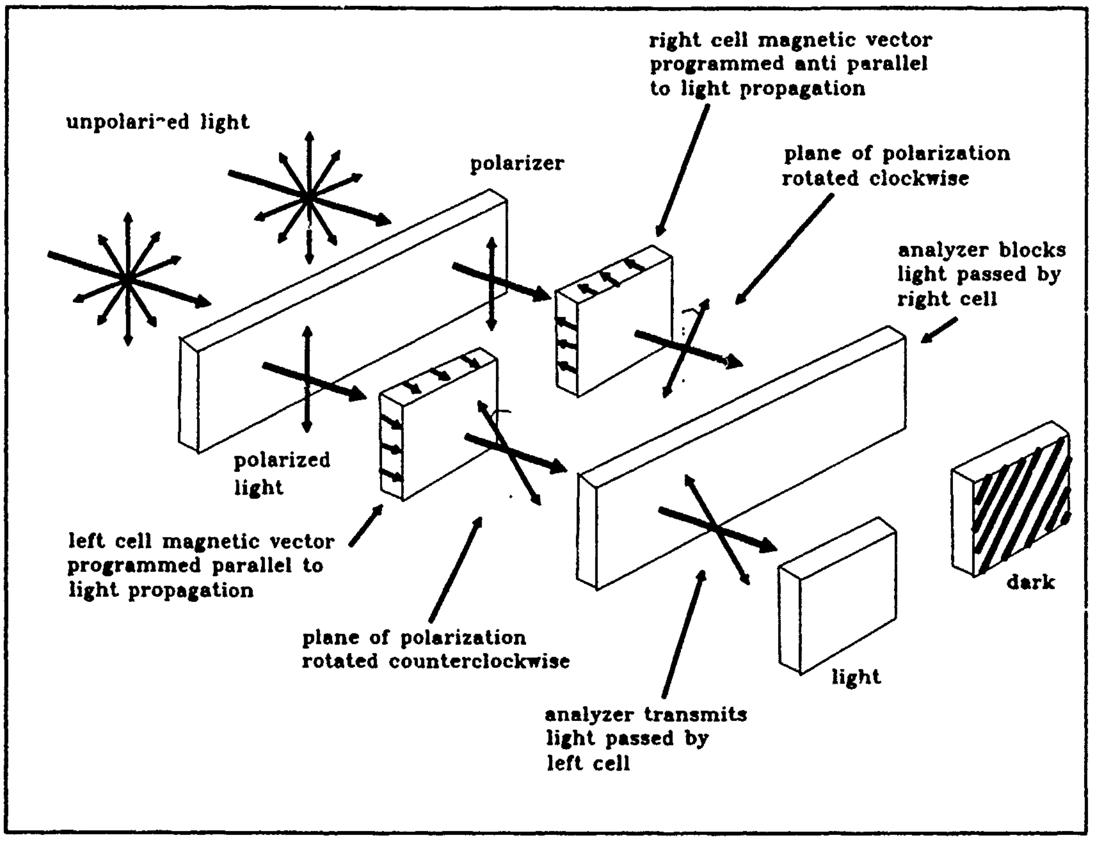

right cell magnetic vector programmed anti parallel to light propagation
unpolari~ed light
polarizer
plane of polarization rotated clockwise
analyzer blocks light passed by right cell
polarized light
left cell magnetic vector programmed parallel to light propagation
plane of polarization rotated counterclockwise
dark
light
analyzer transmits light passed by left cell

Figure 6. MOSLM theory of operation. (25)

coil which surrounds the entire array. Thus, when the pixels are viewed through an output polarizer they will appear dark or light depending on the rotation of their polarization and the position of the output polarization analyzer.

In order to clear the image displayed on the MOSLM an erase pulse, 40 to 50 times as large as the write pulse, is applied to the array. The erase pulse will magnetically rotate the polarization of all pixels back to their original state.

Although the MOSLM is normally used as a binary device, there is a third stable state the pixel magnetization will support (19, 23). This third state is known as the demagnetized or neutral state because each pixel is comprised of "worm-like" regions of the two magnetization states discussed previously.

An enlarged picture of a section of the magneto-optic chip showing the three different states is shown in Figure 7. The "on" state is shown in the top left, while the "off" state is shown in the top right portion of the picture. On the bottom of the photograph the "worm-like" neutral state can be seen.

In order to achieve this third state the MOSLM is first addressed exactly the same as in the binary case described earlier. However, after the saturation pulse is applied an additional nucleation pulse is applied to desired pixels which were erased, but not previously written to. These nucleated pixels will now be in the demagnetized state. The demagnetized pixels do not require an additional saturation pulse.

Not all Semetex Corporation MOSLMs are capable of achieving the demagnetized state. In fact, Semetex has installed permanent magnets in newer MOSLMs in order to prevent accidental selection of this state (34).

Addressing the individual pixels is accomplished via a computer interface card supplied with the MOSLM. This card has dip switches used to set the base address which the interface card will occupy in memory. For this thesis, segment hex B000 was used. Data written in bytes from B000 to B7FF will address all pixels in the 128 × 128 array. The data is written to the MOSLM from left to right then top to bottom. The result is 16 byte rows and 128 byte columns. Data files contain simple binary 1's and 0's to address individual pixels. There are no special codes which need to be programmed. For example, a byte written to memory segment B000 will nucleate the upper left most pixel as well as the seven pixels below it. An erase pulse is generated by a write to address B800; additionally, a saturation pulse is generated by a write to address B801. Detailed programming information is available in the MOSLM Operations Manual (25).

3.1.3 Wavelet Implementation Implementation of the Harr wavelet (Figure 8) is ideally suited to the MOSLM. When the output polarizer is rotated perpendicular to the input beam, polarization modulation states of +1 and -1 are achieved. These modulation states are the result of the 180 degree phase difference between the oppositely rotated

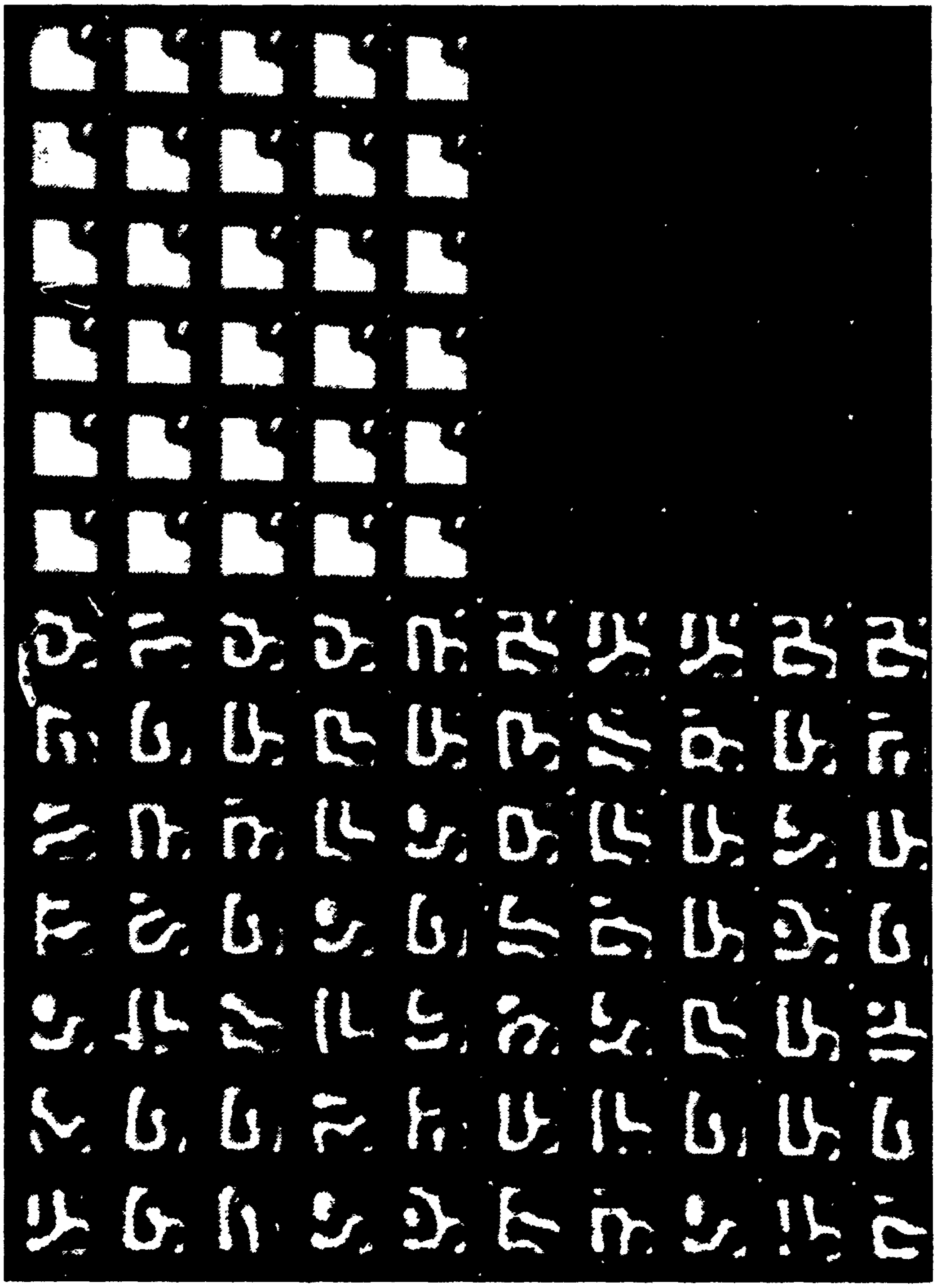


Figure 7. Section of MOSLM chip showing ternary state operation. The three states are the "on" state (top left), "off" state (top right) and neutral "worm" state (bottom).

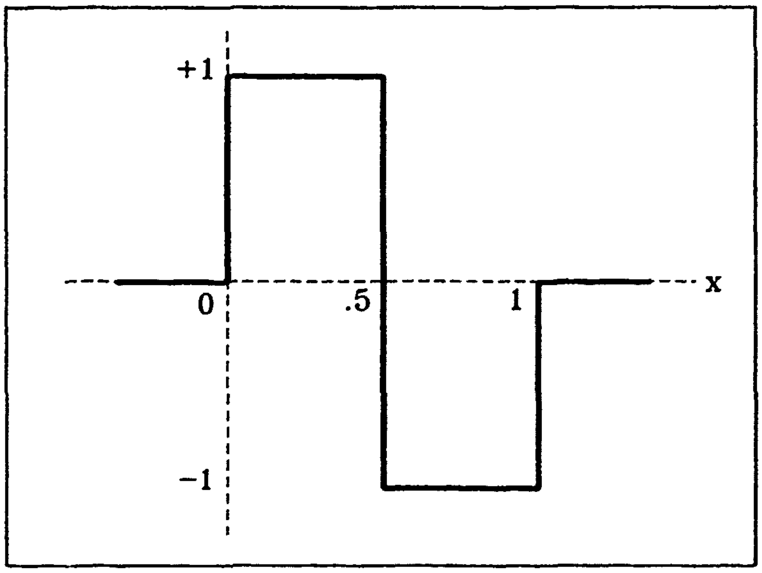
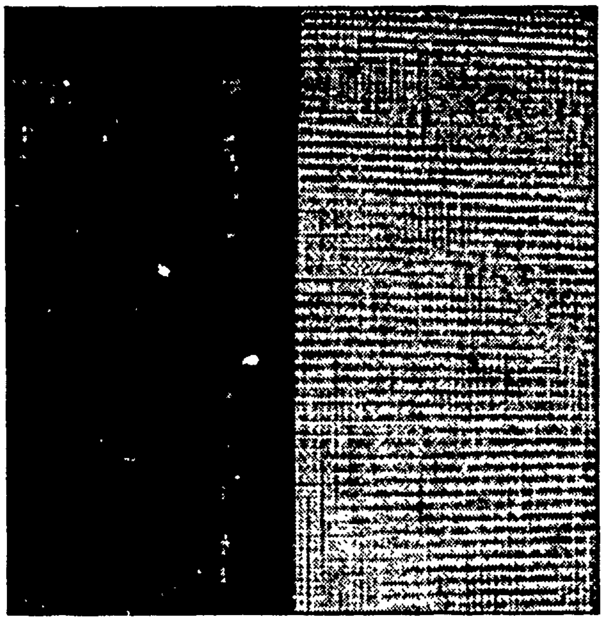


Figure 8. Harr wavelet

magneto-optic states. The zero state is obtained with an additional nucleation pulse to previously erased pixels. Consequently, the three states necessary for implementation of the Harr wavelet are accomplished.

A wavelet of 128 x 128 pixels needs no zero state because it utilized the entire array. Figure 9 shows a 128 × 128 wavelet written as rows of data (Figure 9a) and as columns of data (Figure 9b). The wavelets in Figure 9 are shown with the output polarizer adjusted for maximum contrast. Wavelet results depended on how the "C" code was written and how the MOSLM x and y current drives were adjusted. The wavelet written as rows had very few random bad pixels as opposed to the wavelet written as columns. In addition, the y current drive was adjusted higher than the x current to minimize random bad pixels.

Further adjustments to the MOSLM were accomplished using the COIL, WR and ERA controls. The ERA was increased until the entire MOSLM was completely erased with one erase pulse. The coil was increased to the point where all nucleated pixels were fully saturated with one print pulse. Finally, the WR was adjusted for minimum random


a.

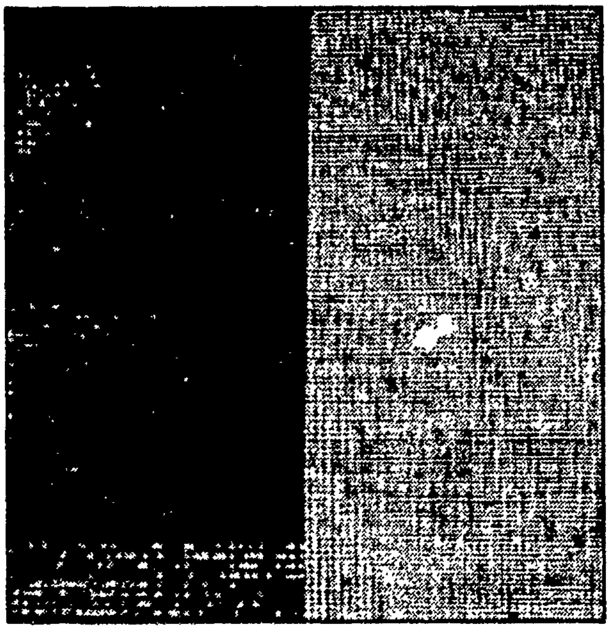

b.

Figure 9. MOSLM image displaying a 128 × 128 pixel wavelet produces fewer random bad pixels when data is written as rows. a. wavelet written as rows; b. wavelet written as columns.

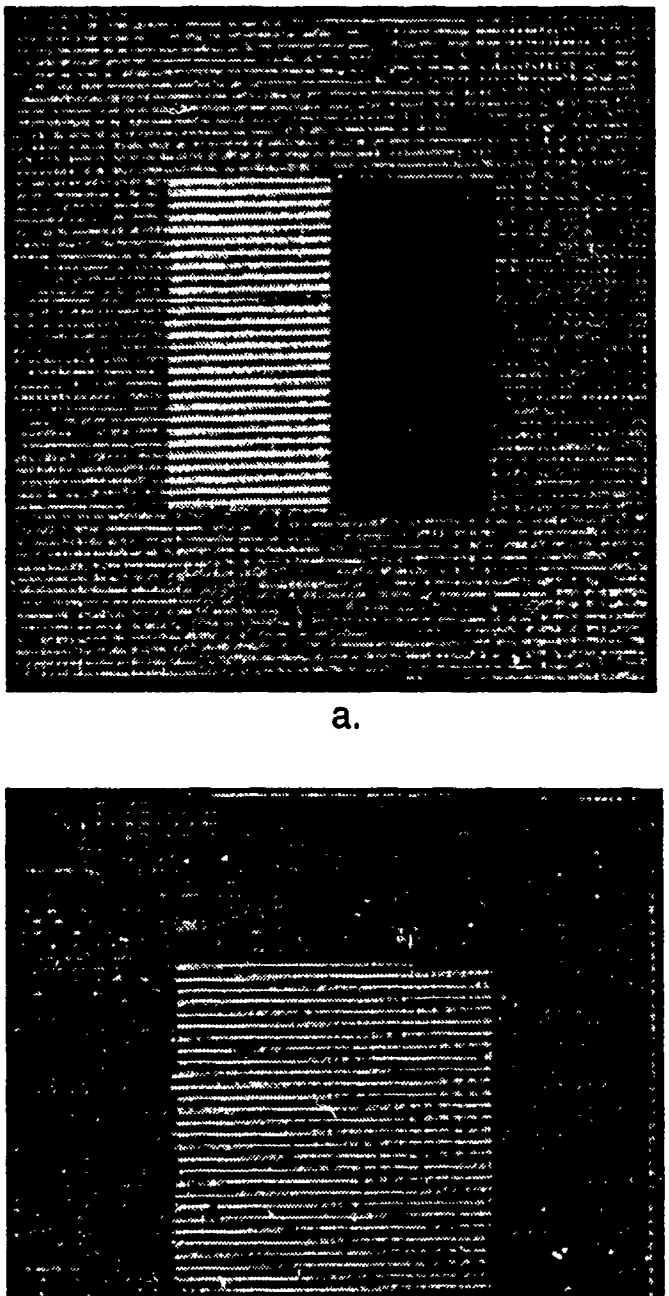
bad pixels. The WR control was very touchy and needed readjustments numerous times throughout this research.

As reported by Kast (13), the zero modulation state is produced using high-order diffraction produced by the MOSLM. This was tested using a 64 × 64 pixel wavelet as displayed in Figure 10. First, a programmed wavelet is shown with the output polarizer rotated to produce maximum contrast between the three MOSLM states (Figure 10a). Next, the output polarizer is rotated perpendicular to the input polarization producing the +1 and -1 states (Figure 10b). Notice the zero state surrounding the wavelet appears just slightly darker than the +1 and -1 states. This is due to the random worm-like nature of the demagnetized pixels. Finally, the wavelet shown in Figure 10b is viewed through the optical spatial filtering setup of Figure 11. A low-pass (pinhole) filter designed to pass spatial frequencies up to half of the Nyquist spatial frequency is positioned in the Fourier plane of the setup. The result shown in Figure 12, shows transmission of the zero state through the low-pass filter is greatly reduced. Indeed, the neutral state produces high-order diffraction. The wavelet zero crossing is clearly revealed in the filtered image (Figure 12).

Wavelet sizes used in this thesis are shown in Figure 13. The 128 × 128, 64 × 64 and 32 × 32 pixel wavelets are photographed on the entire MOSLM display. On the other hand, the 16 × 16, 8 × 8 and 4 × 4 pixel wavelets are enlarged, showing only the portion of the MOSLM necessary to view the wavelet. Although the pixels measure only 76 microns each, they can be counted from the photos of the enlarged wavelets. All wavelets were programmed in Turbo C for display on the MOSLM. Wavelet source code is provided in Appendix C.

3.2 Correlation

3.2.1 Vander Lugt Filtering In order to produce coherent optical correlation, it is necessary to record amplitude and phase information of a desired transfer function. Vander Lugt demonstrated a method for producing a single frequency-plane mask for


a.

b.

Figure 10. MOSLM image displaying a 64 × 64 pixel wavelet. a. polarizer adjusted for maximum contrast; b. polarizer adjusted for equal intensity (actual wavelet).

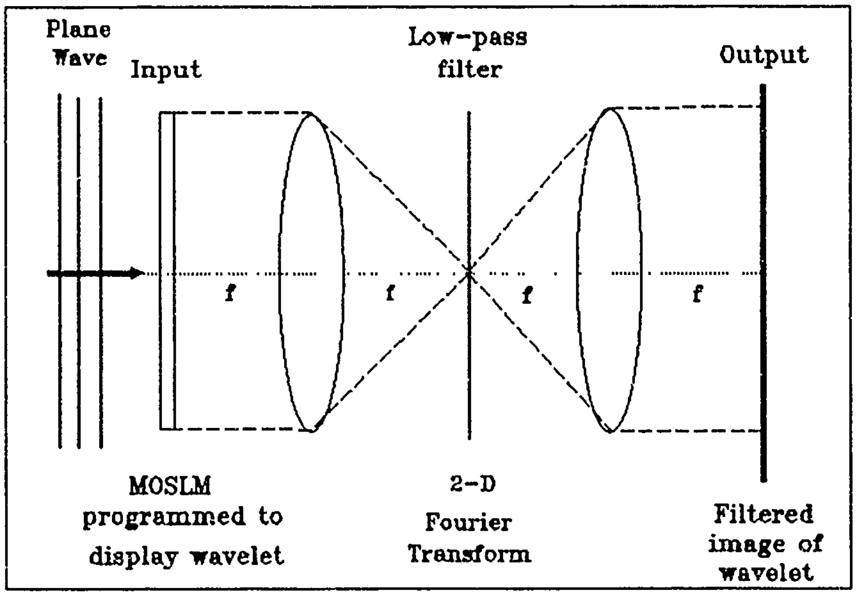


Figure 11. Spatial filtering of wavelet.

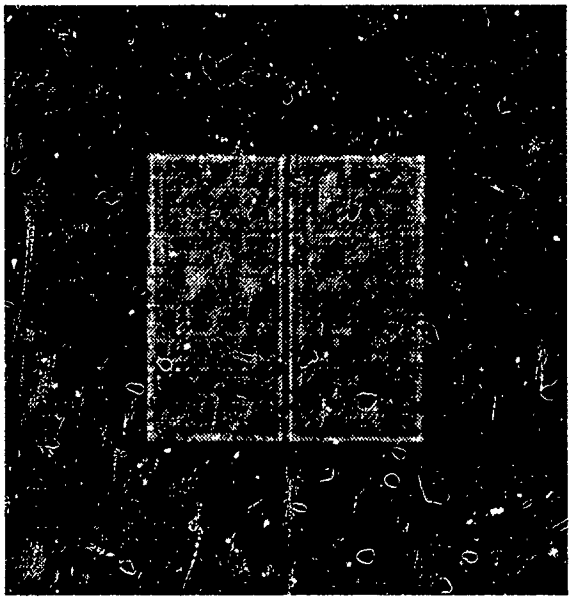


Figure 12. Wavelet in Figure 10b spatially filtered by setup in Figure 11.

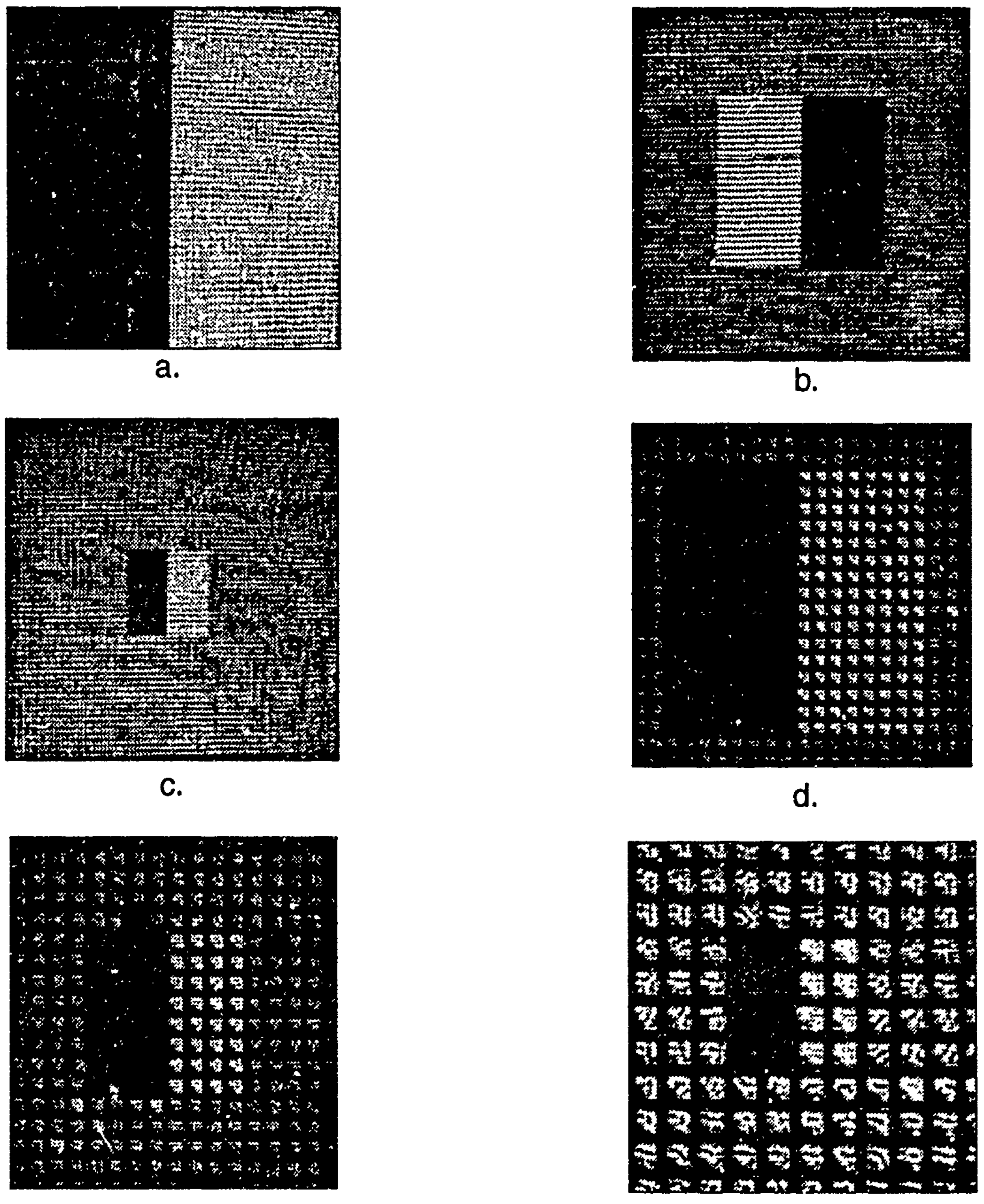


Figure 13. Wavelet sizes used in this research include: a) 128 × 128; b) 64 × 64; c) 32 × 32; d) 16 × 16; e) 8 × 8; f) 4 × 4.

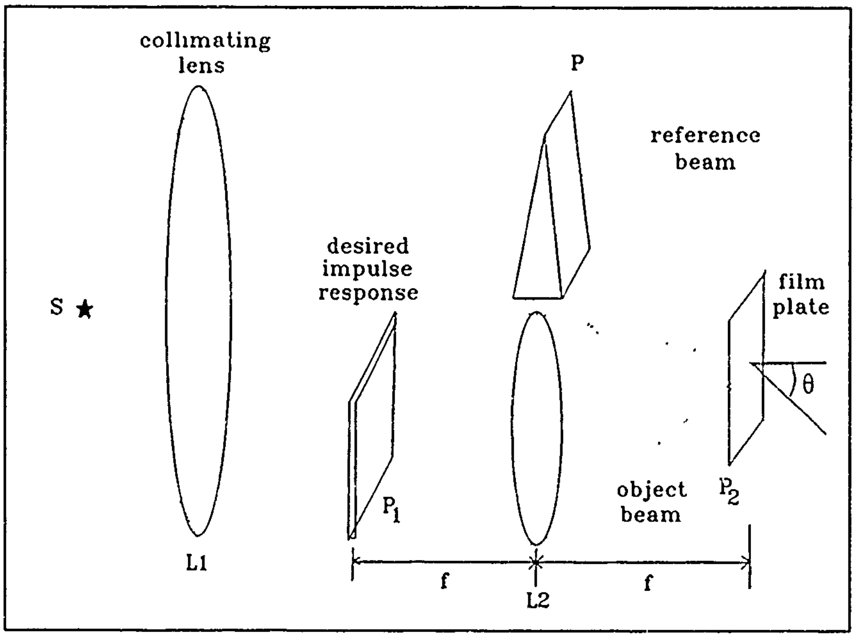


Figure 14. Schematic for recording a frequency-plane mask. (11:172)

this purpose. Prior to this important discovery, amplitude masks and phase masks were recorded separately. This limitation allowed reproduction of only very simple transfer functions. Vander Lugt's technique of interferometrically recorded frequency-plane masks overcame this limitation. (31)

The optical setup for recording a Vander Lugt filter is shown in Figure 14. Briefly, a collimated light source is used to generate an interference pattern on the film plate at P_2. The interference pattern is caused by the reference beam (collimated light deflected at angle θ by prism P) mixing with the object beam (Fourier transform of object with desired impulse response at P_1). The film plate will record the amplitude and phase of the desired transfer function as an interference pattern. Goodman provides a complete explanation and derivation of Vander Lugt filtering (11:171).

Generally, developing Vander Lugt filters requires holographic techniques which utilize chemical processing of high resolution film. In this thesis thermoplastic holograms were created electrically.

3.2.2 Thermoplastic Holograms A Newport HC-300 Holographic recording device was used to store the interference pattern of the Vander Lugt filter. The thermoplastic recording medium used by the HC-300 is developed electrically in about one minute. Thermoplastic recording plates are guaranteed to produce high quality holograms for 300 exposure/erasure cycles or one year. It is possible to obtain as many as 1000 exposure/erasure cycles for up to 3 years; however, diffraction efficiency will be reduced.

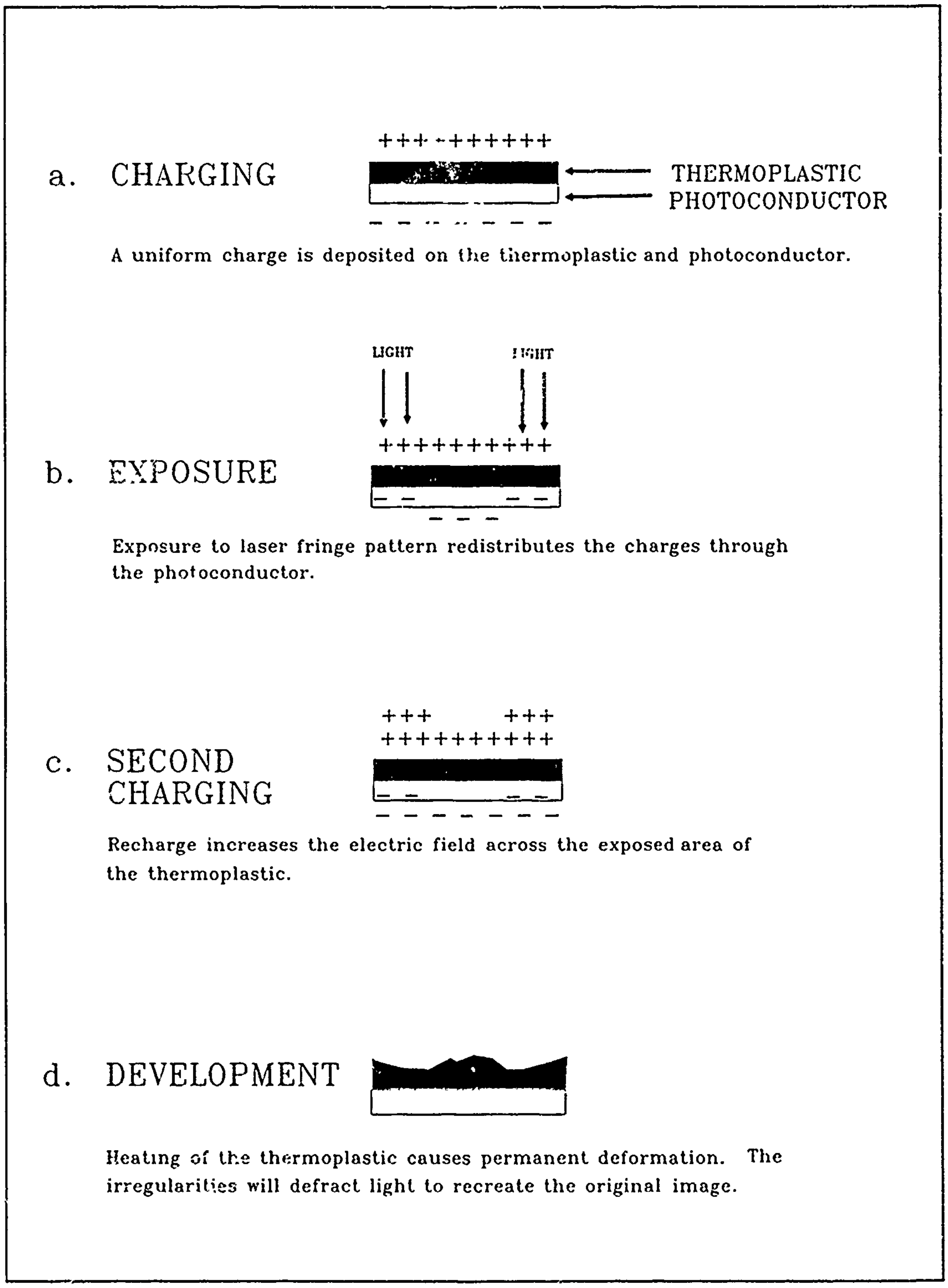
The thermoplastic recording plate is a transparent four layer structure: quartz substrate, transparent heating elements, photoconductor and thermoplastic material. A typical hologram is created in a four-step process shown in Figure 15 (for simplicity, only 2 layers of the thermoplastic medium are shown).

First, a uniform charge is deposited on the plate using a moveable coronotron cylinder located in the camera. The coronotron scans up and down, one cycle, across the thermoplastic plate. Once the plate is charged, it must be kept in complete darkness due to its sensitivity to light.

Next, the plate is exposed to both the object and reference beams. The exposure time is calculated by the HC-500 Holographic System Controller. Exposure time will depend mainly on the intensity of the two light beams on the thermoplastic plate. This exposure of the plate will redistribute the charge according to the interference fringe pattern developed by the object and reference beams.

The third step is a second charging of the thermoplastic plate by the movable coronotron. This charging will increase the charge density on the exposed areas of the plate.

Finally, the thermoplastic is developed by heating the plate. A current is passed through the transparent electrodes, delivering 20 joules in 20 msec into the thermoplastic. Areas of the plate which received a high charge density tend to thin out while other areas bulge.



A uniform charge is deposited on the thermoplastic and photoconductor.

Exposure to laser fringe pattern redistributes the charges through the photoconductor.

Recharge increases the electric field across the exposed area of the thermoplastic.

Heating of the thermoplastic causes permanent deformation. The irregularities will defract light to recreate the original image.

Figure 15. Four-step process for recording a thermoplastic hologram. (17:2)

The resulting surface-relief phase hologram can achieve diffraction efficiencies nominally of 10%. High diffraction efficiency is possible because the surface-relief hologram does not absorb light like a conventional amplitude hologram.

For this research, the holographic setup is shown in Figure 16. The HC-310 Holographic Camera held the thermoplastic plate during the recording process. Filtered nitrogen was used as a cooling gas for the HC-310. An HC-320 Holographic Camera Controller and a HC-500 Holographic System Controller operated together to control the object and reference beam shutters, exposure time, beam intensity ratio, and camera. Prior to developing a hologram, the beam intensity ratio was adjusted using a ratio detector mounted in the HC-310 camera. A beam ratio of 10 to 1, reference beam to object beam, was achieved by using neutral density filters. Prior to making a hologram, the ratio detector must be removed from the camera and replaced with a thermoplastic plate.

Once the holographic system was setup and adjusted correctly, holograms were created in one minute by pressing the SGL (Single exposure) button on the HC- 500. Pressing this button initiates an erase process, as well as the four-step hologram recording process discussed earlier. Detailed system information is available in the Newport Operator's Manuals (17, 18).

3.2.3 Optical setups A coherent processing system for optical correlations can utilize the Vander Lugt filter discussed in section 3.2.1. The basic system is shown in Figure 17. A point source S is collimated by lens L_1 to strike an object at P_i. At P_f the Fourier transform of the object at P_i is optically multiplied by the frequency plane mask generated earlier. Lens L_3 performs another Fourier transform rather than an inverse Fourier transform; therefore, the output at P_c is flipped.

Plane P_c of Figure 17 will produce three separate light fields (Figure 18). First, the field centered at the origin of P_c is of little interest because mathematically it is the addition of two terms which cannot be separated. Next, the convolution of the object at P_i with the impulse response of the Vander Lugt filter is located at (0, -f sinθ). Finally, the

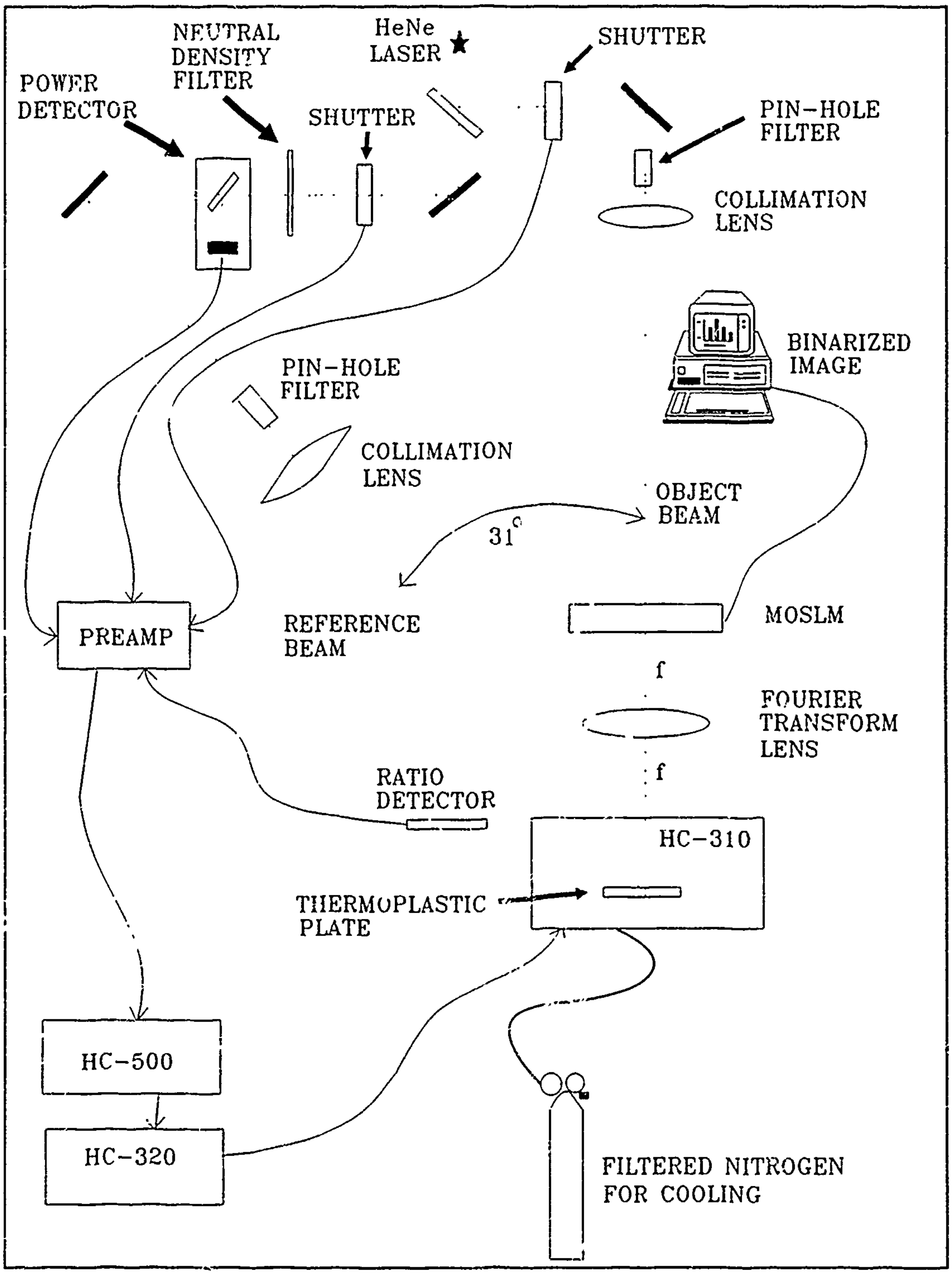


Figure 16. Holographic setup used to record a frequency-plane mask.

correlation of the object at P_t with the impulse response of the Vander Lugt filter is located at $(0, f \sin\theta)$. It is the last field we are concerned with for this research. A complete derivation for processing Vander Lugt optical correlations is provided by Goodman (11:174).

Two basic optical setups were constructed for this thesis due to the different methods of wavelet implementation. First, wavelet dilation is controlled using a single aperture to reduce the size of the collimated object beam. As mentioned in section 3.1.3, the other method for controlling wavelet dilation is by spatial filtering of the MOSLM display.

3.2.3.1 Wavelet Implementation Using a Single Aperture For the first setup, the actual optical components of the correlation system are shown in the schematic of Figure 19. The laser source was a 60mW HeNe (632.8nm) mounted beneath a Newport optical table. A beam splitter directed the laser light into object and reference beam paths. The path lengths were equal to within .5cm, well within the coherence length of the laser. An angle of 31 degrees was used between the object and reference beams in order to obtain a spatial frequency of 800 lines/mm at the holographic camera. Wavefront reconstruction is greatest for a spatial frequency of 800 lines/mm because this is where diffraction efficiency is highest (17).

Following a lengthy alignment process for the object and reference beams, correlations are performed automatically. The Vander Lugt filter is first created using an input scene displayed on the MOSLM. Binarization methods for displaying the input scene on the MOSLM are explained in Appendix B. Optical correlations are performed by programming a 128×128 wavelet on the MOSLM and blocking the reference beam using a remotely switched shutter. Wavelet dilation is controlled using an aperture immediately in front of the MOSLM. For smaller wavelets, only the central portion of the MOSLM would be illuminated by the collimated object beam through the aperture. The Fourier transform of the wavelet is achieved by lens L_3 and is aligned with the frequency-plane mask held by the holographic camera. The correlation results are viewed with a CCD camera located in-line with the reference beam path. The CCD camera sends the images to a TARGA

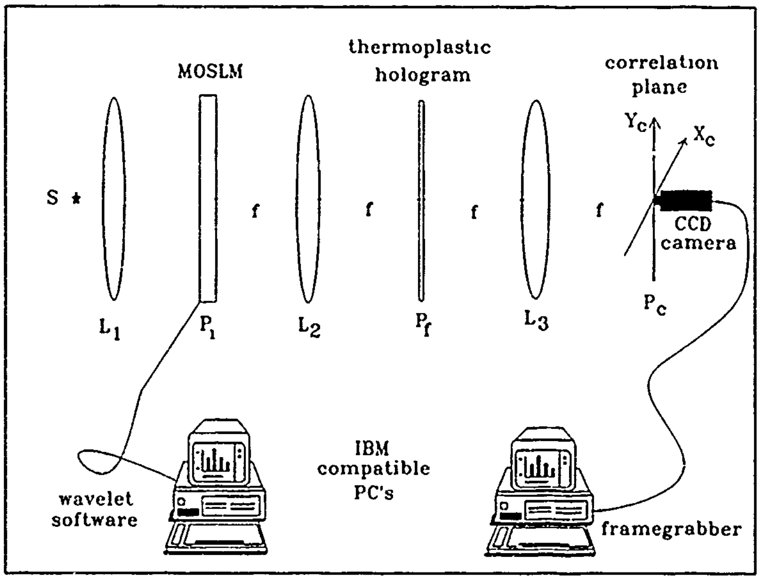


Figure 17. Vander Lugt correlation setup.

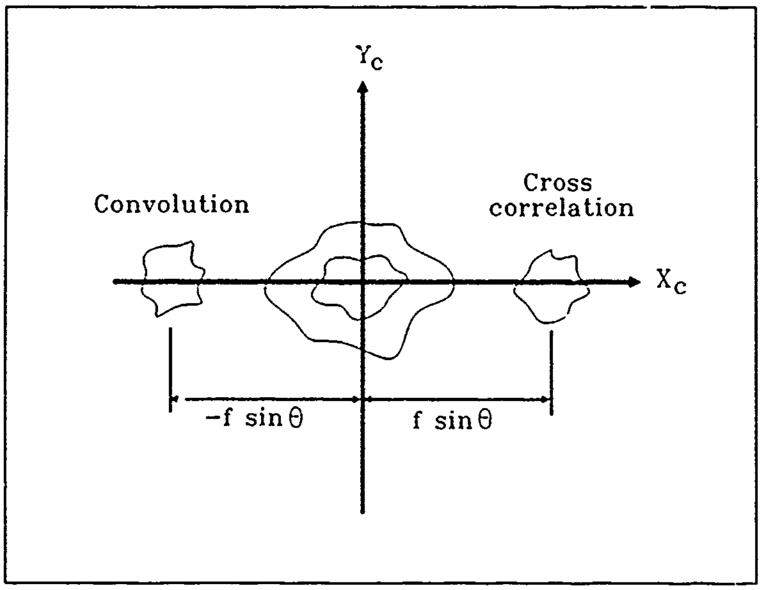


Figure 18. Output from correlation setup in Figure 17.

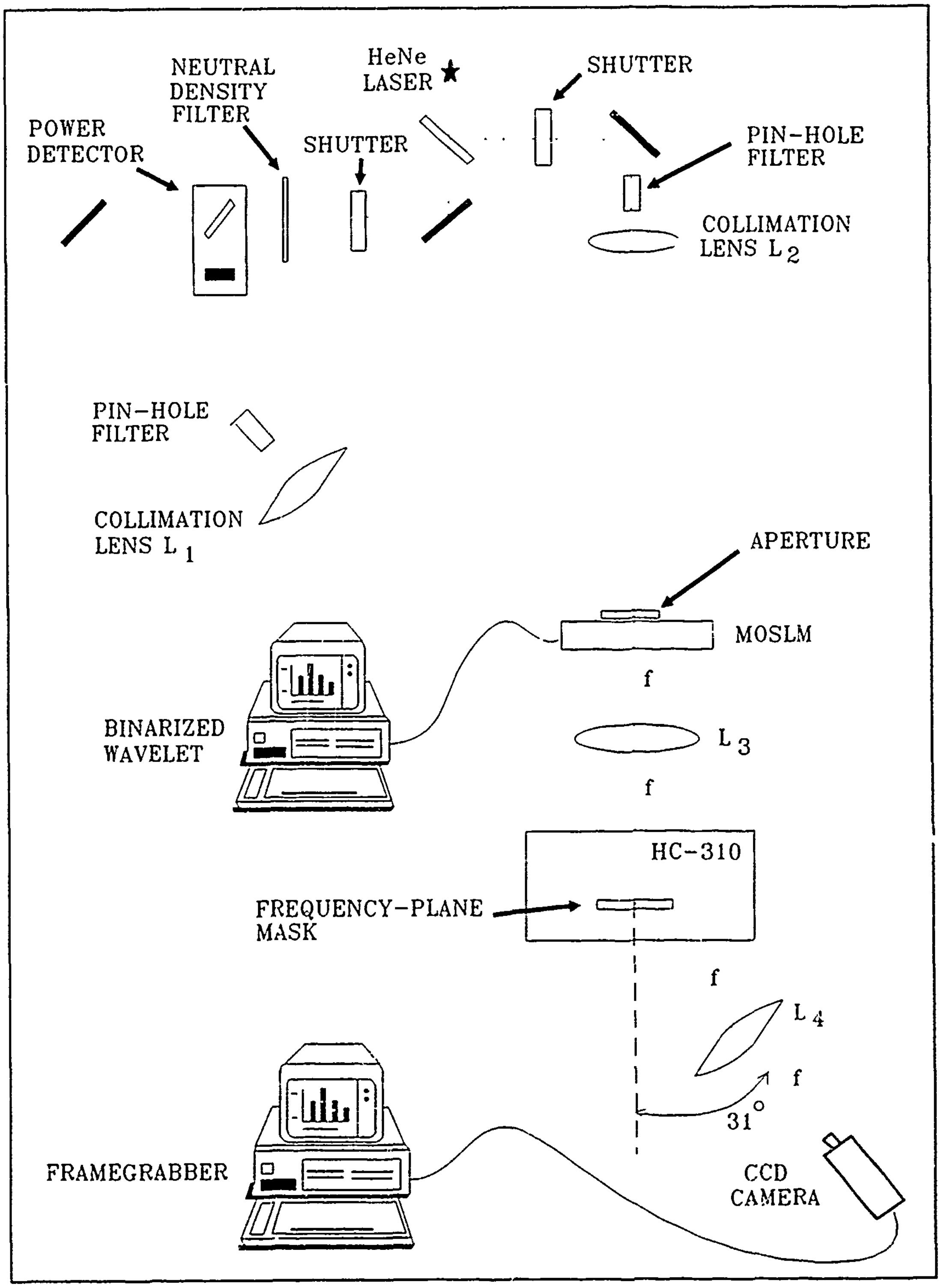


Figure 19. Correlation setup with single aperture wavelet implementation.

framegrabber for viewing on a television set and saving on a floppy disk. Framegrabber operation is reviewed in Appendix A.

3.2.3.2 Wavelet Implementation Using Spatial Filtering The second setup is identical to the first except for two additional lenses (L_3 ans L_4) and a spatial filter to control wavelet dilation (Figure 20). For this setup, lens L_3 Fourier transforms the wavelet displayed on the MOSLM. The Fourier transform is spatially filtered using a low pass (pinhole) filter as described in section 3.1.3. A filtered image of the Fourier transformed wavelet is then imaged onto the holographic camera by lenses L_4 and L_5. Instead of using an aperture for smaller wavelets, as in the first setup, the worm-like neutral state is filtered.

Correlations are viewed in real-time as various wavelets are programmed onto the MOSLM. The holographic camera performs as holder for the Vander Lugt filter, eliminating the need to align the filter with the MOSLM. If correlation with a different frequency-plane mask is desired, a new thermoplastic hologram must be made. Optical joint transform correlation techniques can eliminate the need for holography (7).

3.3 Summary

Segmentation of an image is performed by the following method: 1) binarize and display an input image on a MOSLM device; 2) record a frequency plane mask of this binarized image using thermal holography; 3) implement a Harr wavelet onto the MOSLM; 4) optically correlate the input image with the wavelet using a Vander Lugt filter as a coherent processing system. All the relevant technologies are discussed in this chapter for performing segmentation by this method. Correlation results are discussed in the next chapter.

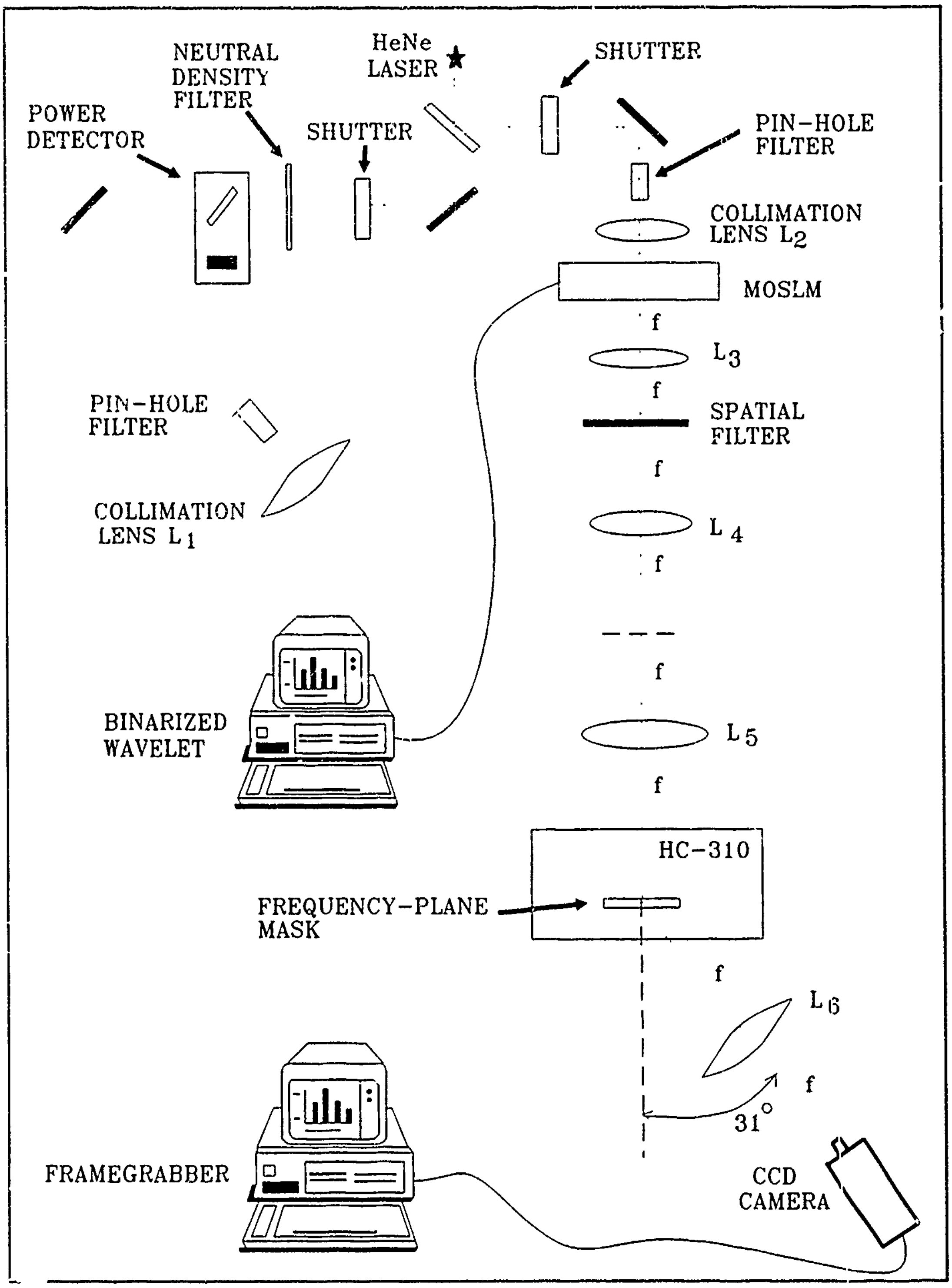


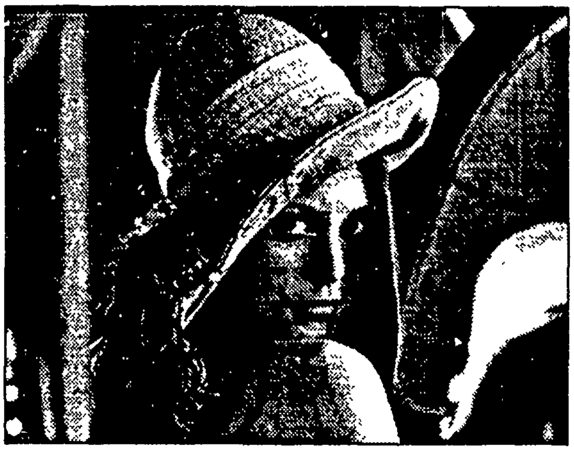
Figure 20. Correlation setup with spatial filtering wavelet implementation.

IV. Results and Discussions

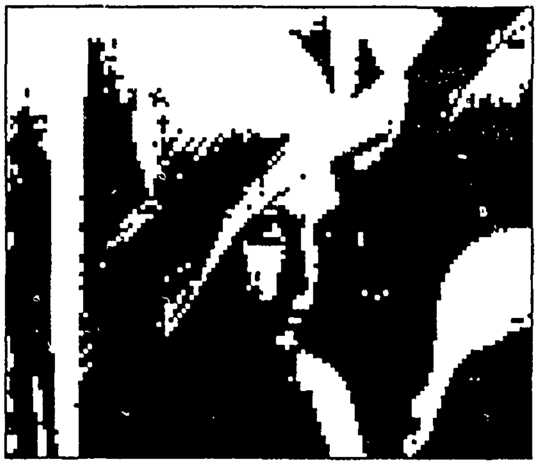
This chapter, divided in six parts, presents and discusses results based on the methodology described in chapter III. The reference image used for image segmentation is discussed in the first section. Next, correlation tests are presented for both experimental setups shown in chapter III. The third section covers correlation results using a signal aperture wavelet implementation. Included in this section are discussions on aperture selection as well as correlation results for all three binarization methods. The fourth section covers correlation using a spatial filtering wavelet implementation is covered. This section will also discuss correlation results from each of the three reference image binarization techniques. Lastly, a short summary will conclude the chapter.

4.1 Reference Image

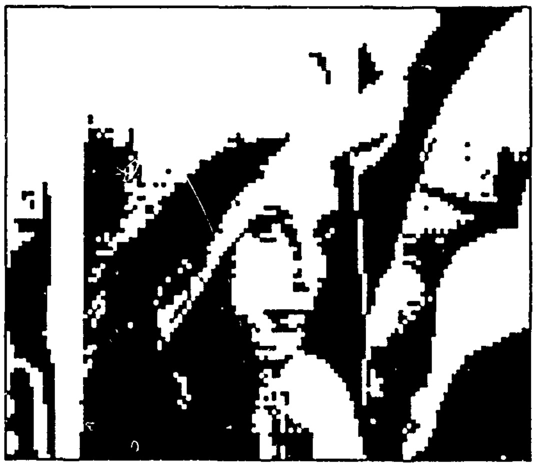
The reference image, Lenna (Figure 21a), is used throughout this research. This reference image was scanned (300 dpi) from the original photograph. A 400 × 512 TARGA file was generated using a CCD camera and a framegrabber to capture the reference image. The three binarization methods used to implement Lenna on the MOSLM were compressed to 100 × 128 from the 400 × 512 TARGA file (Figures 21b thru 21d). The first method binarizes the reference image based on an average pixel value of the entire image. The second method binarizes the reference image based on a localized 4 × 4 pixel average of the entire image followed by average pixel value over the remaining compressed image. The third method binarizes the reference image based on a localized 3 × 3 pixel average of the entire image "ANDed" with the average pixel value of the entire image. A complete explanation of the three binarization methods, including Turbo C source code, is provided in Appendix B.



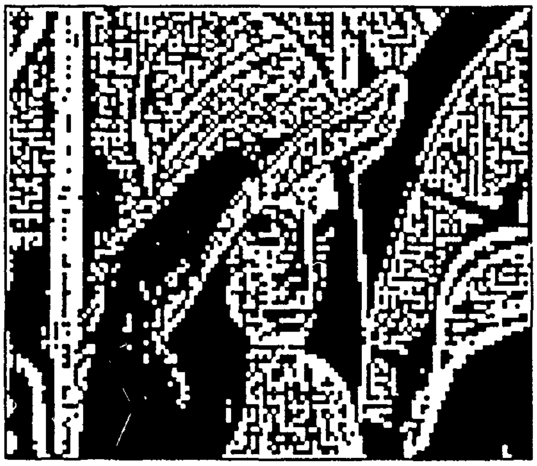
a.



b.



c.



d.

Figure 21. Reference Image and Binarization Techniques. a. Scanned reference image (300 dpi). b. Binarized reference image based on an average pixel value over entire image (Method 1). c. Binarized reference image based on localized 4 × 4 pixel average followed by average pixel value over remaining image (Method 2). d. Binarized reference image based on localized 3 × 3 pixel average "ANDed" with average pixel value over entire image (Method 3).

4.2 Correlation Test

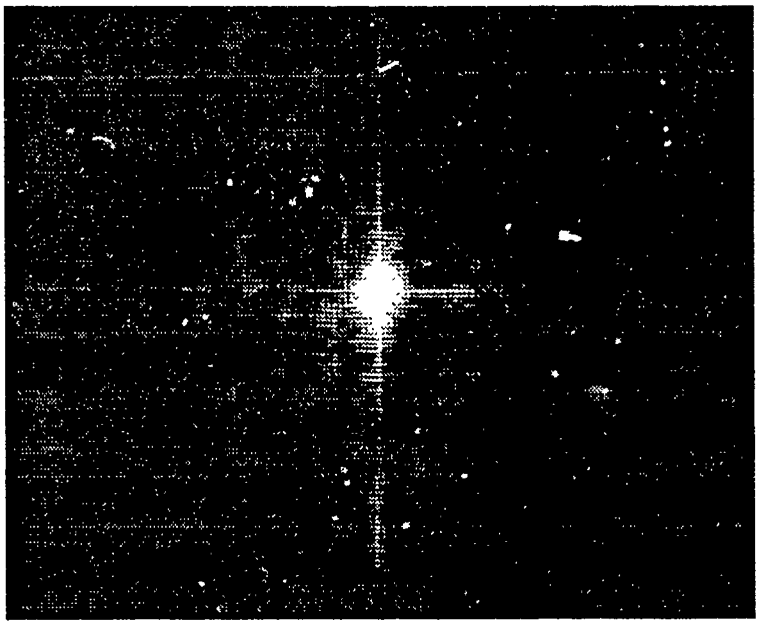
Testing of both optical wavelet implementation setups discussed in chapter III was accomplished in order to verify correlation results. A frequency-plane mask was generated of a binarized image of Lenna programmed onto the MOSLM. The displayed image of Lenna on the MOSLM was then correlated with the frequency-plane mask of Lenna generated earlier. Figure 22a and Figure 22b show results of correlating "Lenna with Lenna" using the setups of Figure 19 and Figure 20 respectively. Both experimental setups yield good correlation results.

4.3 Correlation Using Single Aperture Wavelet Implementation

4.3.1 Aperture Selection Square and circular apertures were tested with and without an optical wavelet on the MOSLM. The idea was to determine the best aperture for optical wavelet correlation results. Figure 23 shows the results of correlation with different apertures. The binarized image displayed in Figure 21b was used for all correlation results appearing in Figure 23.

First, a .4mm square aperture with no wavelet on the MOSLM is correlated with a hologram (Figure 23a). The results is an averaging of the binarized reference image of Lenna. The Fourier transform of the wavelet using a square aperture is periodic in nature and varies as a two-dimensional sinc function (10). Off-axis frequency components of the 2-D sinc function fall off rapidly and do not multiply well with the off-axis frequency components recorded on the hologram. On the other hand, on-axis frequency components of the 2-D sinc function multiply very well in the frequency-plane with the recorded hologram. The result is a multiplication of only low frequency components off-axis, while high and low components are multiplied on-axis. Details of the binarized image are blurred and no particular edge enhancement is achieved.

Next, a .4mm square aperture is used with a vertical wavelet implemented on the MOSLM (Figure 23b). No significant difference is achieved with the wavelet because the effect of the square aperture overwhelms the differentiation effect of the wavelet. Using a


a.

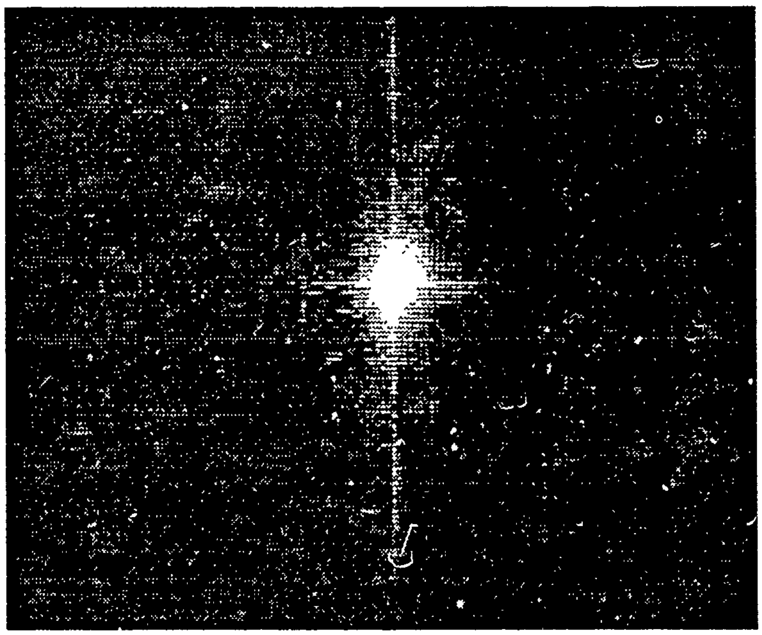

b.

Figure 22. Correlations of "Lenna with Lenna." a. Using setup of Figure 19. b. Using setup of Figure 20.

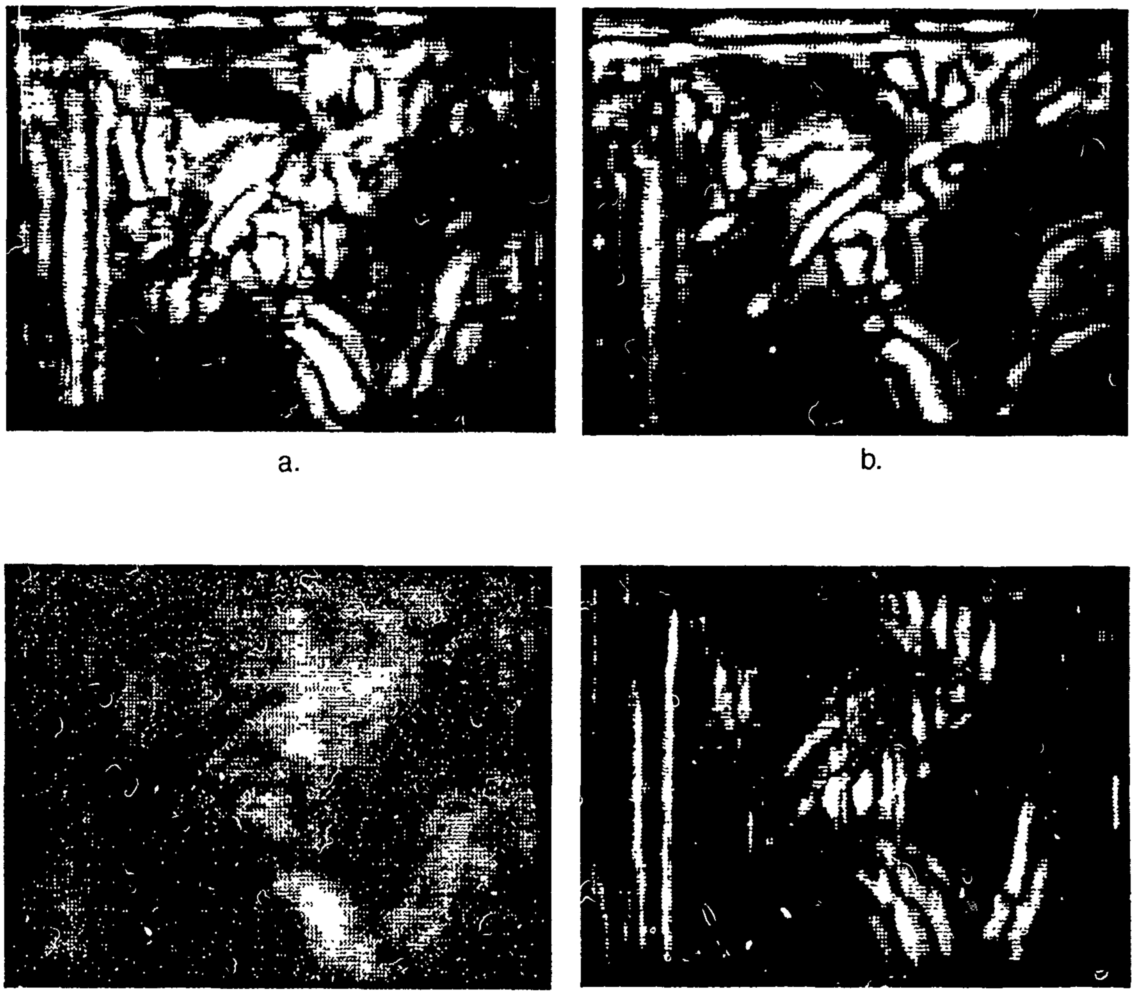
square aperture the optical wavelet contribution to the reconstructed image is very difficult to ascertain.

Experimenting with a .5mm circular aperture was first conducted with no wavelet on the MOSLM (Figure 23c). Similar to Figure 23a, Figure 23c shows blurring of the binarized reference image. However, the bright details of the image are washed out. The Fourier transform of the wavelet using a circular aperture is periodic in nature and varies as a two-dimensional sombrero function (10). Off-axis as well as on-axis frequency components multiply equally well with the frequency-mask of the binarized reference image. Nevertheless, since the sombrero function falls off faster than the sinc function, only low frequency Fourier components will be multiplied using the circular aperture. Put another way, the Airy disk, resulting from the circular aperture's Fourier transform, will illuminate only the center low frequency spectra recorded on the hologram. The result is that high frequency details of the picture are lost.

Lastly, a .5mm circular aperture was tested with an optical wavelet programmed on the MOSLM (Figure 23d). Here we can see the best effect the wavelet has on the reconstructed image. The circular aperture provides a dilation for the wavelet while the wavelet itself produces a differentiation effect. The vertical wavelet differentiates between areas of high contrast in the binarized reference image. The .5mm circular aperture was used for all remaining results in this section.

4.3.2 Binarization Method 1 Correlation results with the single aperture wavelet implementation technique varied greatly, depending on the binarization method used. Binarization method 1 produced very good segmentation of the reference image (Figure 24). A vertical wavelet correlated with the thermal hologram was used to segment vertical edges from the image (Figure 24a); similarly, a horizontal wavelet was used to segment horizonal edges from the image (Figure 24b).

Comparing Figure 24a with Figure 21b, the vertical details of the image are clearly distinguished by the wavelet. Any vertical change in contrast caused the wavelet to act as


a.

b.

c.

d.

Figure 23. Aperture selection results for maximum wavelet effect; hologram is using binarization method 1. a. Correlation with a .4mm square aperture and no wavelet. b. Correlation with a .4mm square aperture over a vertical wavelet. c. Correlation with a .5mm diameter circular aperture and no wavelet. d. Correlation with a .5mm diameter circular aperture over a vertical wavelet (best wavelet correlation result).

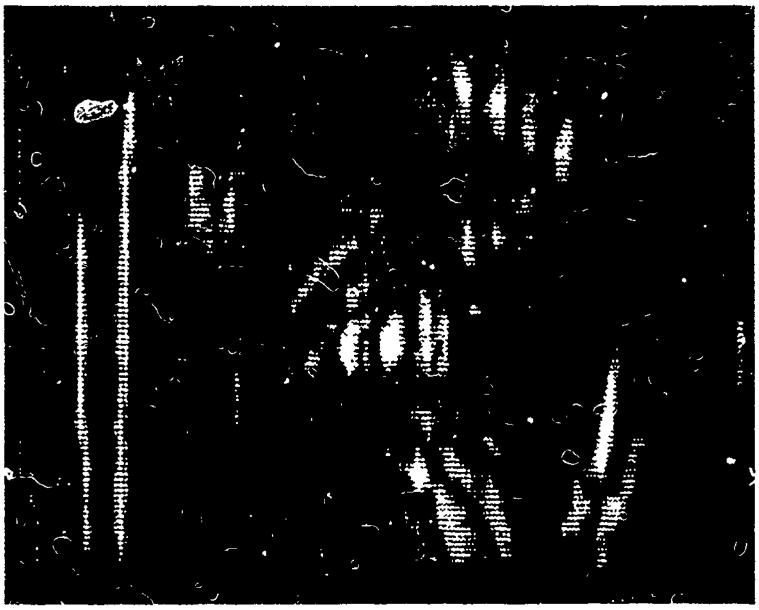
a differentiator; consequently, the light intensity increases at that point in the correlation. Likewise, comparing Figure 24b with Figure 21b, the horizonal edges from the image stand out. Notice the change in contrast at the sides of the image also produce high correlation with the optical wavelet.

4.3.3 Binarization Method 2 Changing image binarization techniques requires making a new thermal hologram. Moreover, no two holograms are ever the same in quality or diffraction efficiency. The correlation results for binarization method 2 produced the best segmentation results; although correlation contrast was not as good as that discussed in section 4.3.2. This reduced correlation contrast appears to be due to hologram quality rather than binarization methods. Although numerous holograms for binarization method 2 were produced, none of the holograms appeared to diffract as much light as the hologram made with binarization method 1.

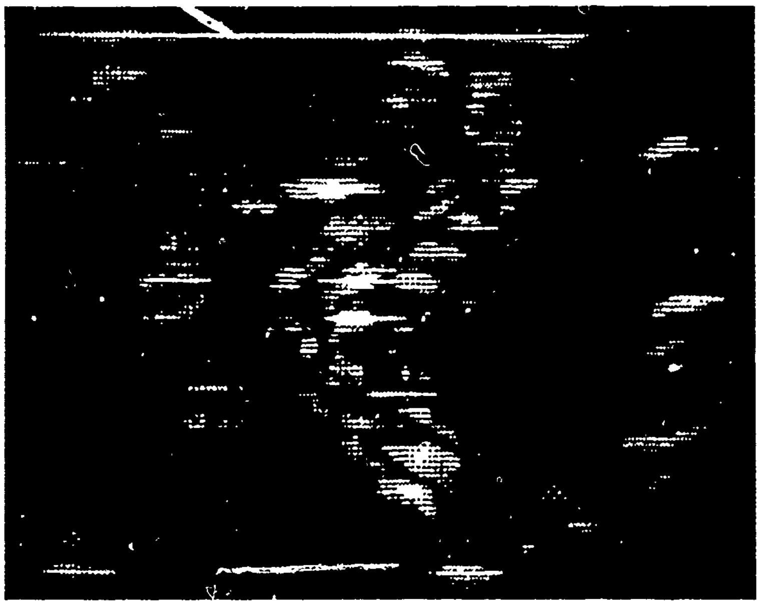
Comparing Figure 25a with Figure 21c, the vertical details of the image are differentiated quite well. However, the poor contrast of this correlation result tends to detract from the segmentation capability. Additional vertical edges are segmented with binarization method 2 which were not detected using binarization method 1. Therefore, binarization method 2 produced the best image segmentation using the single aperture wavelet implementation.

Similarly, comparing Figure 25b with Figure 21c, the horizontal details of the reference image are highlighted using a horizontal wavelet. Again, some horizontal details are revealed in this segmented version which were not seen in Figure 24b.

4.3.4 Binarization Method 3 Image binarization based on a localized 3×3 pixel average "ANDed" with the average pixel value over the entire scene has been very effective when used in a joint transform correlation scheme (7). This binarization method, shown earlier in Figure 21d displays much of the image texture lost in the other two binarization methods. However, comparing Figure 26a with Figure 21d, it is clear that the wavelet does not segment the horizontal details as well with binarization method 3. This lack of

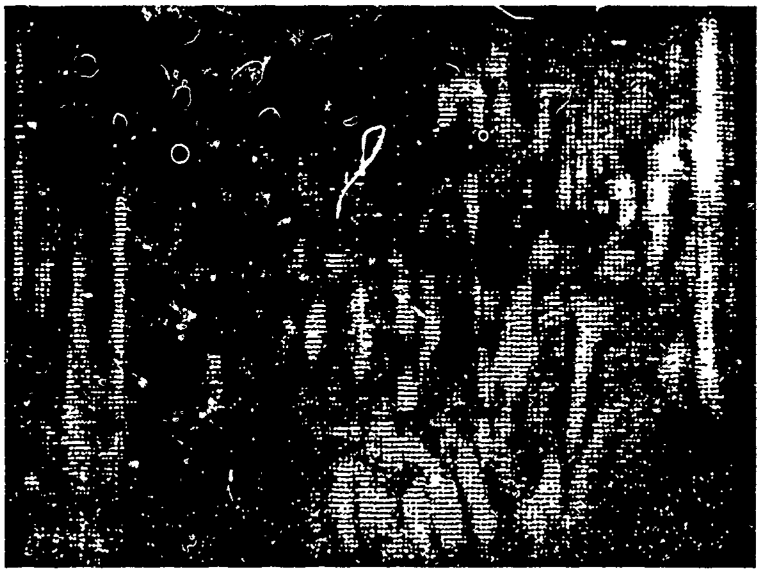


a.

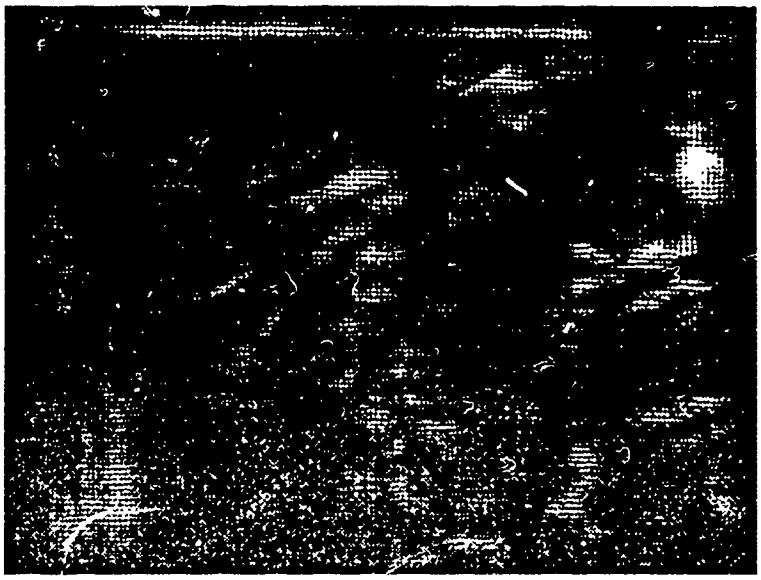


b.

Figure 24. Correlation using single aperture wavelet implementation and a binarized reference image based on an average pixel value over entire image (binarization method 1). a. Vertical wavelet results. b. Horizontal wavelet results.



a.



b.

Figure 25. Correlation using single aperture wavelet implementation and a binarized reference image based on localized 4 × 4 pixel average followed by average pixel value over remaining image (binarization method 2). a. Vertical wavelet results. b. Horizontal wavelet results.

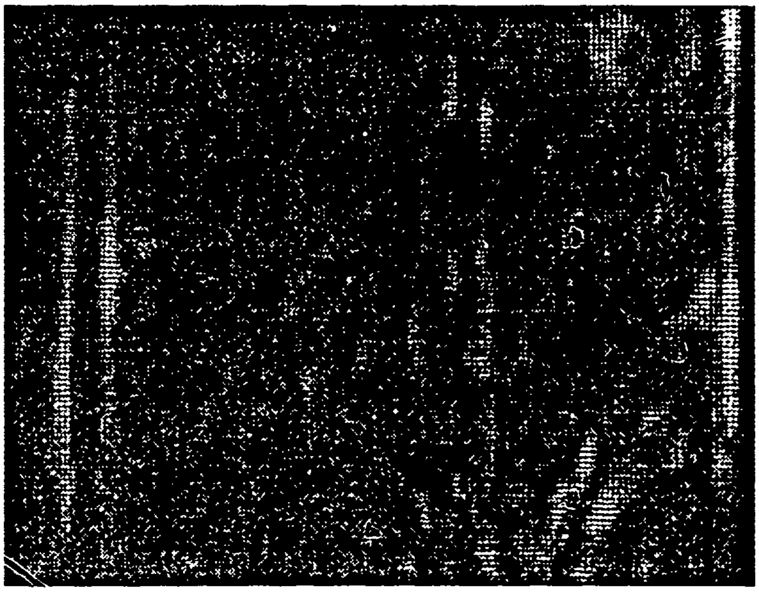
segmentation is primarily caused by not using a small enough wavelet during correlation. A smaller wavelet could not be implemented in this research because the 60mW HeNe laser used was not powerful enough When using a single aperture wavelet implementation, most of the laser light is blocked by the aperture. A small wavelet dilation requires a high number of photons to project through the small aperture onto the hologram. Because hologram diffraction efficiency was typically 10% in this research, much of the laser power was not directly used. With a more powerful laser, binarization method 3 has the potential to perform the best segmentation.

Comparing Figure 26b with Figure 21d, very few horizontal edges were segmented from the image. The wavelet (.5mm dilation) was virtually lost among all the changes in contrast within this binarized image; consequently, the wavelet could not differentiate between such small sized contrast changes.

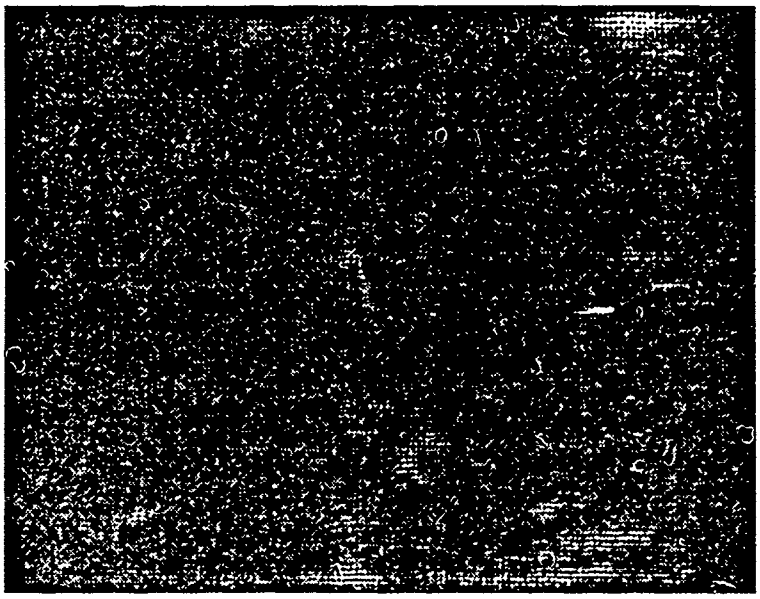
4.4 Correlation Using Spatial Filtering Wavelet Implementation

The spatial filtering of a wavelet, explained earlier in section 3.1.3, did not produce an aperture dark enough to block all light. Control of wavelet dilation, although performed easily by electronically clocking different wavelet sizes into the MOSLM, was compromised by the light passed by the MOSLM zero state. The Fourier transform of the MOSLM programmed to display a wavelet showed a definite annular pattern. Through spatial filtering of this annular ring created by the zero "worm" state, the image of the wavelet should have a dark aperture. Indeed, the aperture was apparent (Figure 12), but not opaque enough.

4.4.1 Binarization Method 1 Comparing Figure 27 with Figure 21b, the 128×128 and 64×64 wavelets did not highlight any edges. The smaller wavelets were capable of contrast differentiation. The 16×16 and 8×8 pixel wavelets, Figure 27d and Figure 27e respectively, produced the best segmentation for binarization method 1.



a.



b.

Figure 26. Correlation using single aperture wavelet implementation and a binarized reference image based on localized 3 × 3 pixel average "ANDed" with average pixel value over entire image (binarization method 3). a. Vertical wavelet results. b. Horizontal wavelet results.

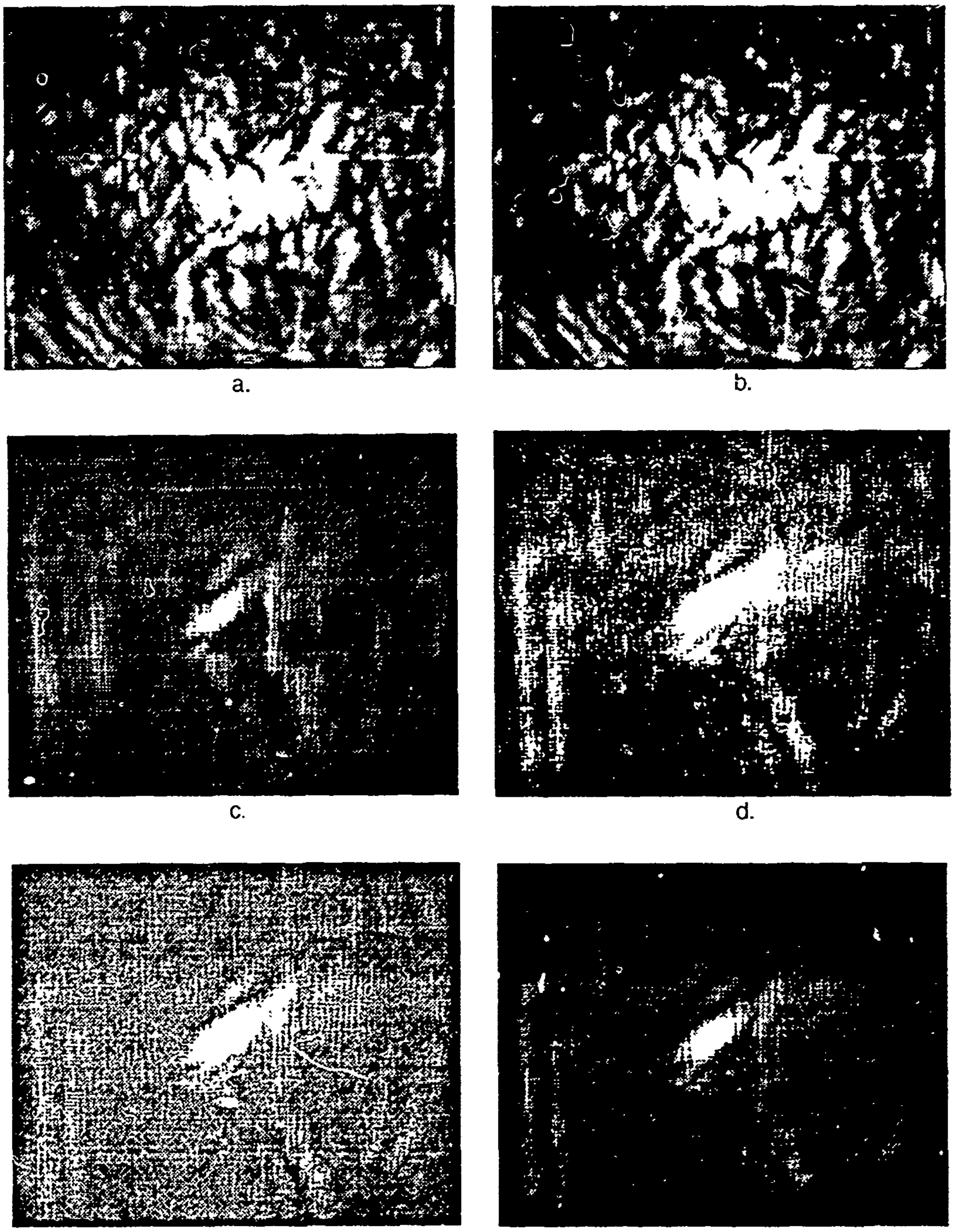


Figure 27. Correlation using spatial filtering wavelet implementation and a binarized reference image based on an average pixel value over entire image (binarization method 1). Correlation results using the following wavelets: a. 128 × 128; b. 64 × 64; c. 32 × 32; d. 16 × 16; e. 8 × 8; f. 4 × 4.

4.4.2 Binarization Method 2 Comparing Figure 28 with Figure 21c, there is no great resemblance. In fact, the only way to adequately describe the correlation results in Figure 28 is as unidentified flattened fauna. The wavelets should have performed as well as the correlation results of section 4.4.1. Perhaps the difference in hologram quality was partially to blame; however, numerous holograms of Figure 21c were produced with the same correlation results as shown in Figure 28.

4.4.3 Binarization Method 3 The results displayed in Figure 29 compared with the binarized image in Figure 21d show little similarity. While Figure 29c-29e do appear to segment the vertical object to the left of Lenna, there is no other visible segmentation. Again the poor results are primarily due to the light which passes through the zero state of the MOSLM. The wavelet dilation is not complete because this zero state cannot be totally filtered.

4.5 Digital Correlation Results

Digital wavelet analysis using the same reference image was performed by Laing (14) and Smiley (26). Although the same reference image was used, resolution of the digital results is 512 × 512 pixels; conversely, optical results were only 128 × 128 pixels. Figure 30 shows digital correlation results for Harr wavelet dilations of 16 × 16, 8 × 8, and 4 × 4 pixels. Following histogram manipulation, setting only pixels with low and high projection value to 1 and all other pixels to zero, the binarized versions of Figures 30a 30c,and 30e are displayed in Figures 30b, 30d, and 30f respectively. Optical wavelet correlation results compare favorably with their equivalent digital results. Vertical and horizontal edges in Figure 30b can be found in Figures 24a and 24b respectively.

4.6 Summary

The reference image used throughout this thesis is discussed as well as correlation tests for the two experimental setups. Three different image binarization methods are used with two different optical wavelet implementations to produce image segmentation.

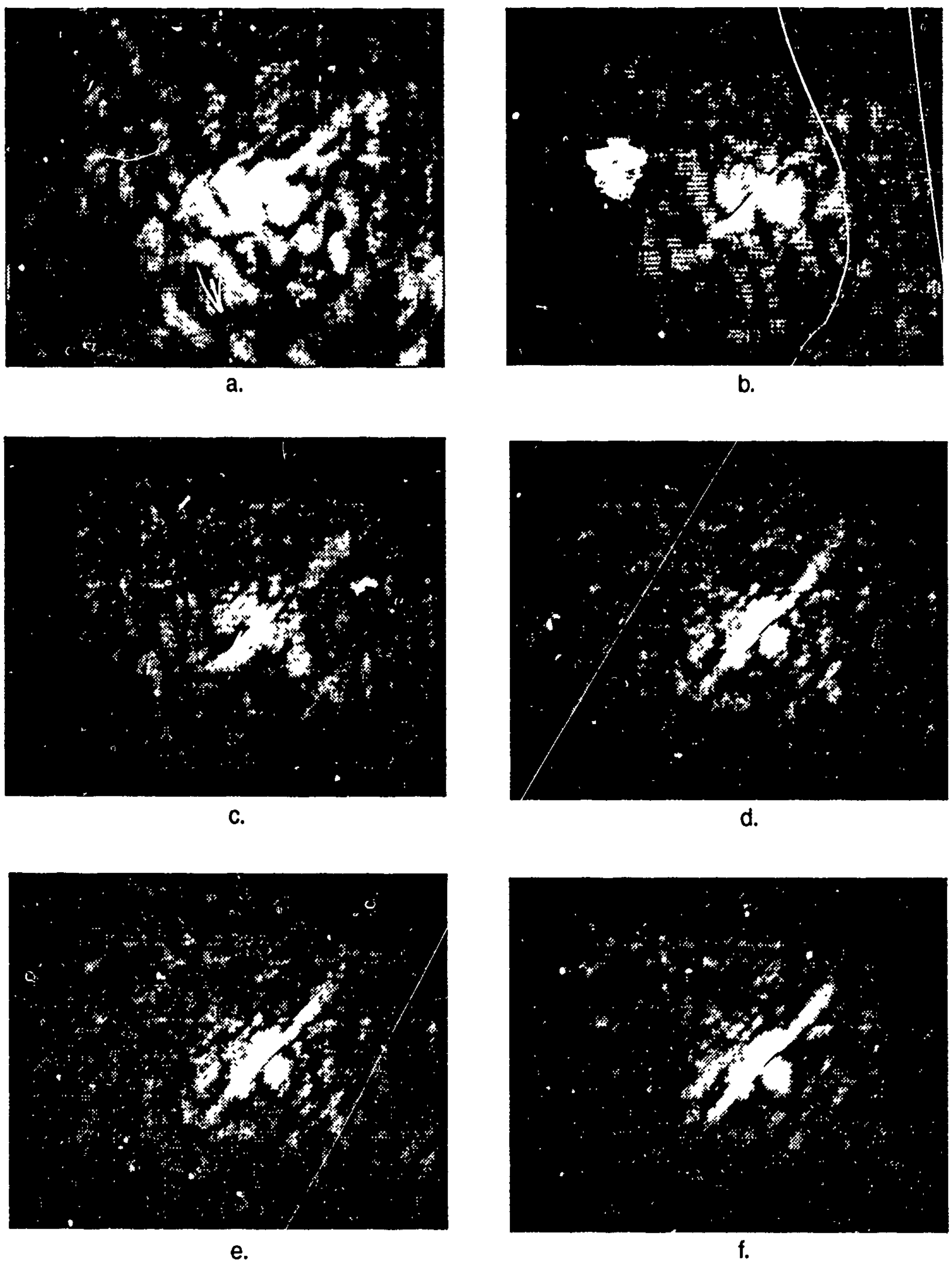


Figure 28. Correlation using spatial filtering wavelet implementation and a binarized reference image based on localized 4 x 4 pixel average followed by average pixel value over remaining image (binarization method 2). Correlation results using the following wavelets: a. 128 x 128; b. 64 x 64; c. 32 x 32; d. 16 x 16; e. 8 x 8; f. 4 x 4.

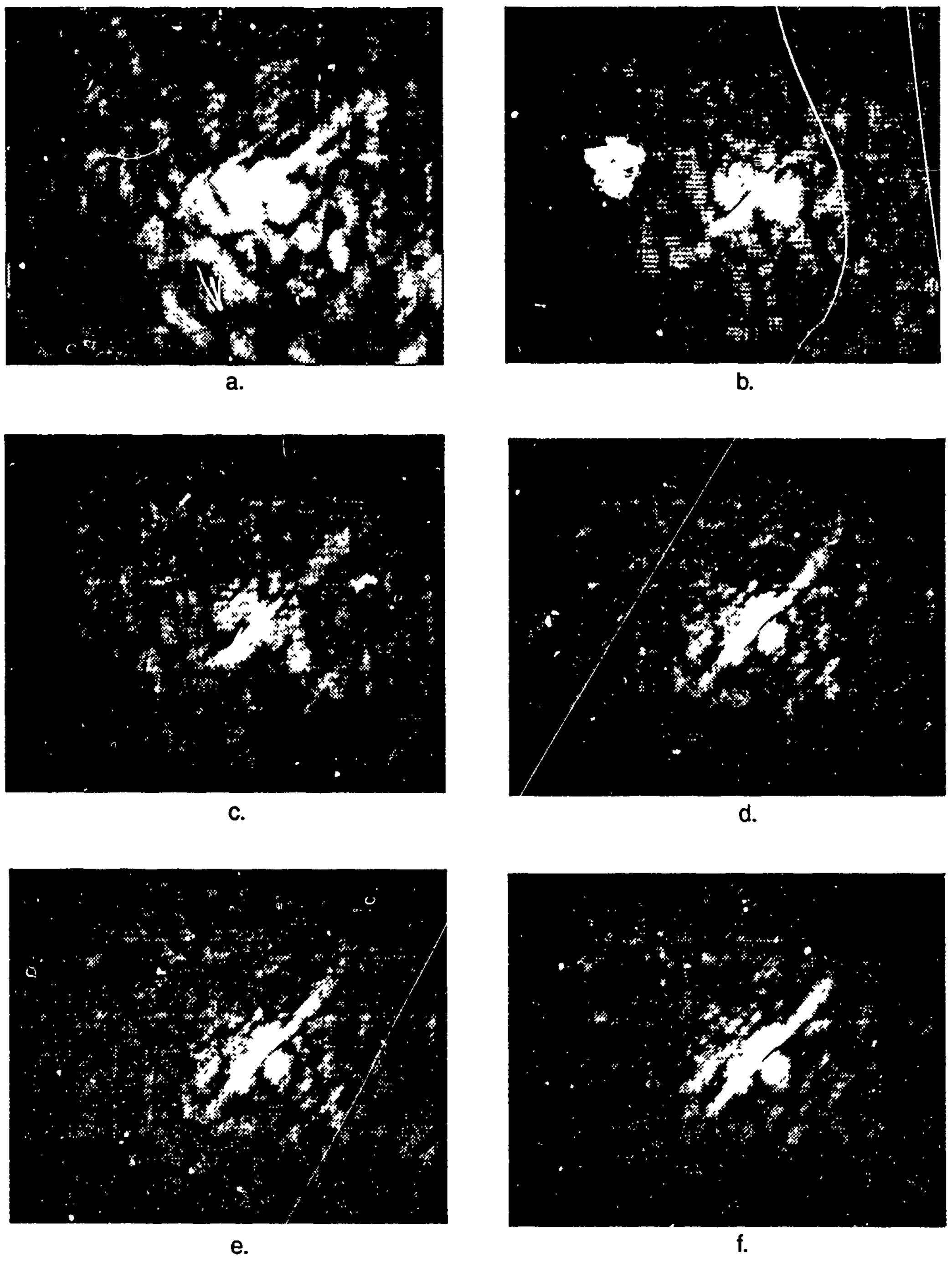


Figure 28. Correlation using spatial filtering wavelet implementation and a binarized reference image based on localized 4 × 4 pixel average followed by average pixel value over remaining image (binarization method 2). Correlation results using the following wavelets: a. 128 × 128; b. 64 × 64; c. 32 × 32; d. 16 × 16; e. 8 × 8; f. 4 × 4.

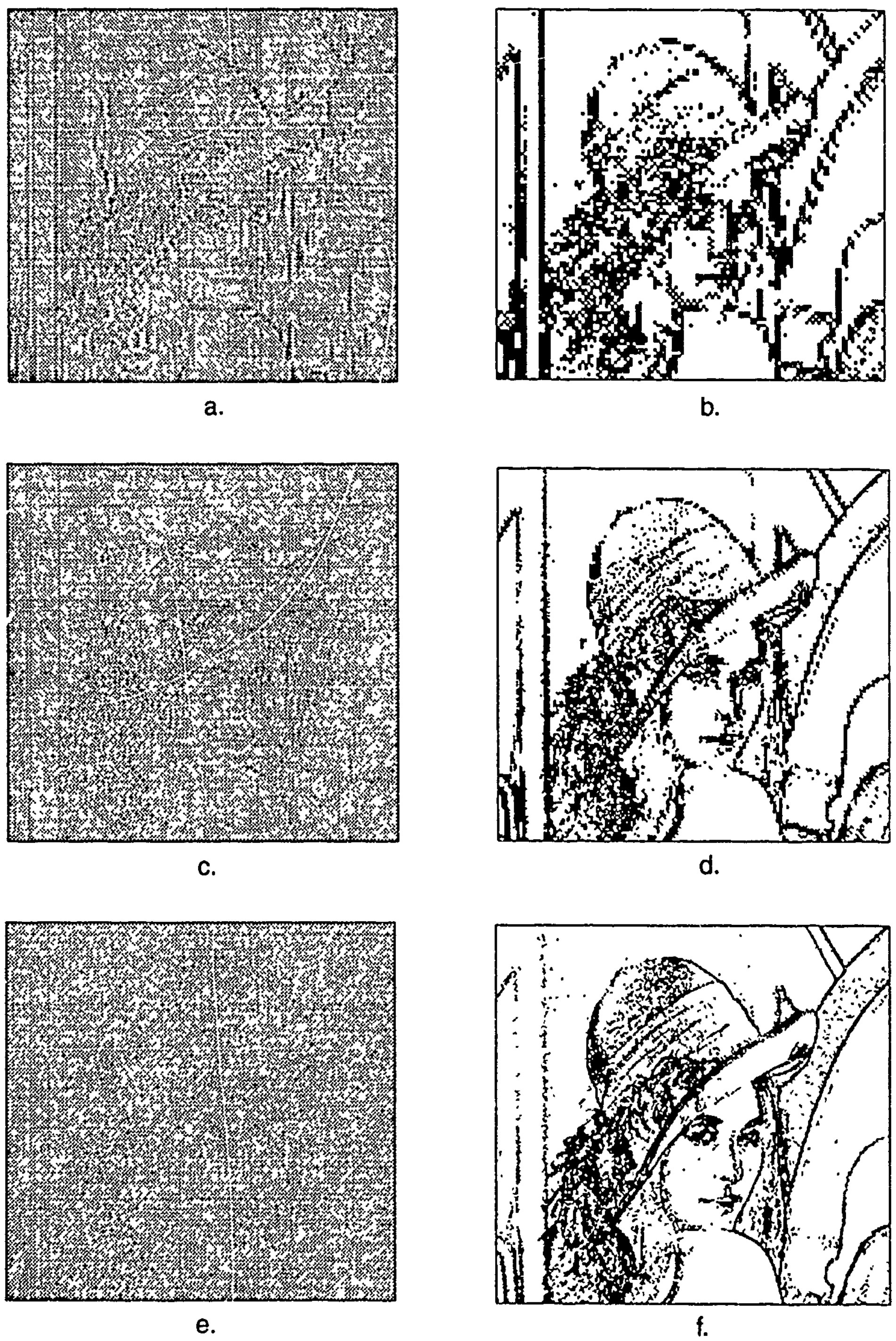


Figure 30. Digital Correlation results using several wavelet dilations. a. 16 × 16 wavelet correlation; b. 16 × 16 wavelet correlation followed by binarization and image inversion; c. 8 × 8 wavelet correlation; d. 8 × 8 wavelet correlation followed by binarization and image inversion; e. 4 × 4 wavelet correlation; d. 4 × 4 wavelet correlation followed by binarization and image inversion.

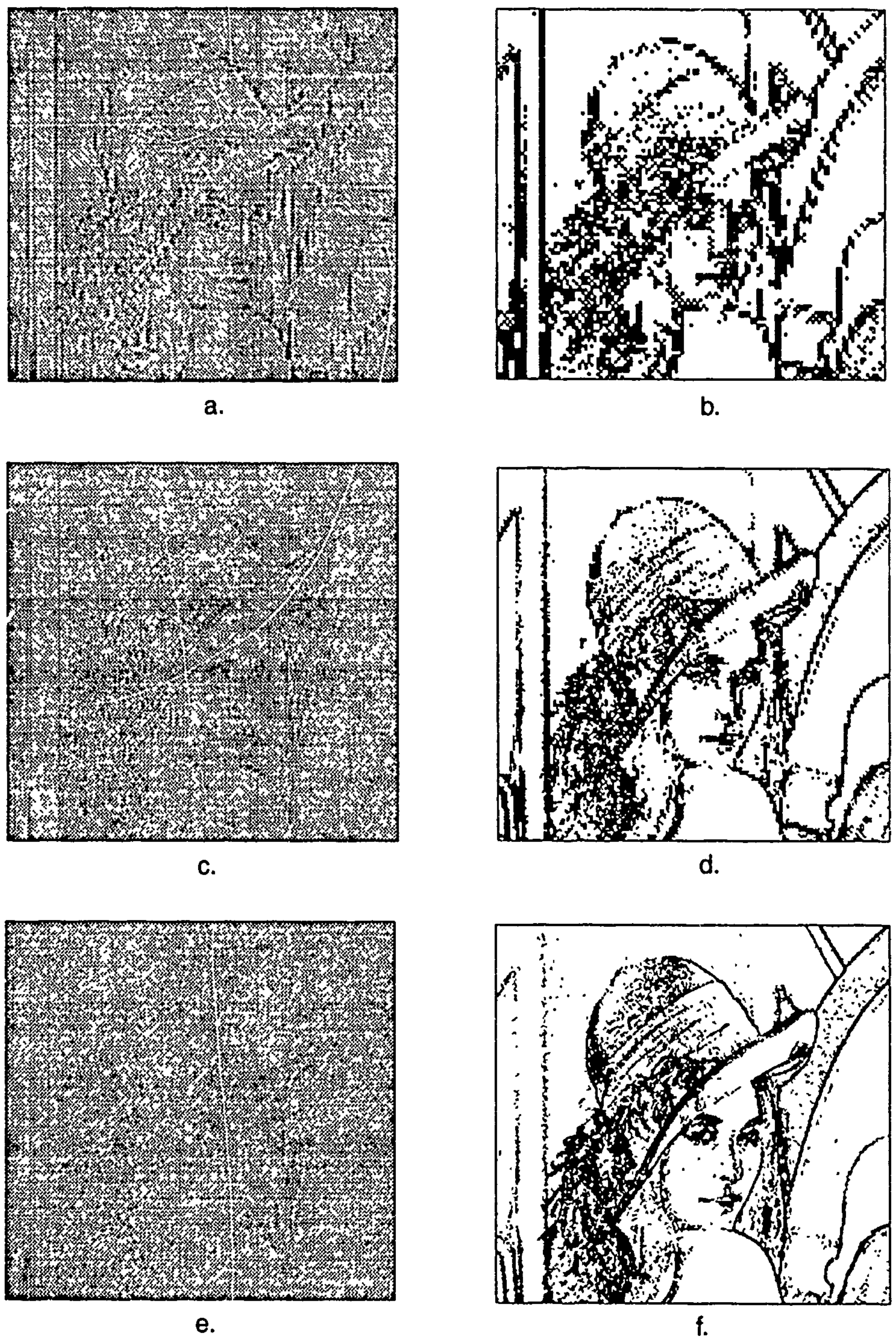


Figure 30. Digital Correlation results using several wavelet dilations. a. 16 × 16 wavelet correlation; b. 16 × 16 wavelet correlation followed by binarization and image inversion; c. 8 × 8 wavelet correlation; d. 8 × 8 wavelet correlation followed by binarization and image inversion; e. 4 × 4 wavelet correlation; d. 4 × 4 wavelet correlation followed by binarization and image inversion.

V. Conclusions and Recomendations

5.1 Summary

This research introduces an optical wavelet method of segmenting potential targets from an image. There are two main goals in this research effort. First, implementation of an optical Harr wavelet using a 128 × 128 pixel magneto-optic spatial light modulator (MOSLM). Second, actual segmentation of objects from an image through Vander Lugt correlation of a binarized image with a binarized wavelet.

Two methods of controlling wavelet dilation were explored. The first method used a single aperture placed in front of the MOSLM to control the amount of collimated light impinging on the displayed wavelet. This single aperture method produced good segmentation results because the zero-state of the Harr wavelet was completely opaque. The second method of controlling wavelet dilation used a spatial filtering setup to block high-order diffraction of the zero state on the MOSLM. Although using spatial filtering is a clever idea, it did not produce an aperture opaque enough to control dilation for the wavelet. Consequently, light was passed through the zero state precluding formation of a true optical wavelet and confusing the correlation results.

Three techniques of in.age binarization were implemented for correlation with the wavelet dilation methods. The first binarization technique found the average threshold of the entire image, then binarized the image using this threshold (method 1). The second binarization technique found local 4 × 4 pixel thresholds, binarized the image according to these local thresholds, and then found the average threshold to binarize the remaining image (method 2). The third binarization technique found local 3 × 3 pixel thresholds, binarized the image according to these local thresholds, and then "ANDed" this locally binarized image with a binarized image based on the average threshold of the entire scene (method 3).

By combining the two optical wavelet dilation methods with the three image binarization techniques, six possible experimental results were obtained.

5.2 Conclusions

Implementation of an optical Harr wavelet was achieved using the MOSLM. The magneto-optic rotation of linearly polarized light, clockwise or counterclockwise, provided the +1 and -1 wavelet states. In addition, control of wavelet dilation using a single circular aperture outperformed spatial filtering of the zero MOSLM state.

Using a .5mm diameter circular aperture with binarization method 1, good segmentation was achieved. Both vertical and horizontal wavelets were used separately in order to segment vertical and horizontal image details respectively. A high diffraction efficiency hologram provided the best correlation contrast for this segmentation scheme.

Correlation using a .5mm diameter circular aperture with binarization method 2 performed the best segmentation. Again, vertical and horizontal wavelets provided differentiation of image details. Segmented details were seen in these correlation results which were not seen earlier using binarization method 1. However, a hologram with a lower diffraction efficiency produced lower correlation contrast.

Using the same .5mm diameter circular aperture with binarization method 3 produced poor results. This binarization method, developed earlier for improved joint transform correlation results (7), did not correlate well with the optical wavelet. A small enough wavelet could not be implemented to adequately differentiate between the small sized areas of contrast developed by the binarization. In order to implement a smaller wavelet, a high power (3-5 watt) laser will probably be needed. Because the best binarized details of the reference image were obtained using binarization method 3, a high power laser may enable this scheme to yield the best overall segmentation results.

Controlling the wavelet dilation with spatial filtering resulted in disappointing results. Because the zero state on the MOSLM produces high-order diffraction, spatial filtering of this zero state should produce an opaque aperture for the wavelet. In fact, the diffraction produced by the zero-state through a 15cm lens was unable to be completely filtered. Thus, the correlation results using all three binarization methods were very poor.

Experimenting with a spatial filter to control wavelet dilation and binarization method 1, very little segmentation could be observed. No segmentation was observed with the other two binarization methods.

5.3 Recommendations

Vander Lugt correlation is based on creating a frequency- plane mask using holography. The higher the diffraction efficiency of the hologram, the better the correlation in the output-plane. Since thermal holograms already have a 10% diffraction efficiency, compared with 1-3% for conventional holograms, improving the frequency-plane mask would be difficult. Conversely, improving the signal-to-noise ratio of the diffracted light through the hologram can be achieved by using a higher power laser. By replacing the 60mW HeNe laser used in this research with a 3-5 watt Argon laser, necessary photons for good correlation would be available. A high power laser would then allow smaller wavelets to be correlated using the single aperture method of controlling wavelet dilation. Smaller optical wavelets will yield better image segmentation.

Spatial filtering of the MOSLM zero state to control wavelet dilation may still be an alternative. Using a long focal length lens to Fourier transform the MOSLM wavelet would spread the annular ring produced by the zero state. Spatial filtering of this frequency spectra would be easier if it were larger. However, an experimental setup using a long focal length lens would have to be quite creative to fit on an optical bench. Two-lens magnification of the frequency-plane is also an option; on the other hand, alignment of five lenses in series becomes increasingly difficult.

Appendix A. *Framegrabber Operation*

AFIT has several AT&T Truevision Advanced Raster Graphics Adapter (TARGA) 8 framegrabbers capable of capturing images in real-time. The problem is that most of these boards in AFIT's possession do not work properly. When a functional TARGA 8 framegrabber is set up correctly, it supports image resolution of up to 512 by 482 pixels with 256 levels of gray.

The information presented here is intended to help first- time TARGA users set up the basic system. AT&T TARGA documentation and software providing detailed information is available (1, 3, 2, 28, 30, 29).

First, it is necessary to set the TARGA input impedance dip switch located between the VIDEO IN and VIDEO OUT ports (1:2-5). Failure to properly set this switch can result in an unsyncronized monitor display. Next, the hardware configuration is set using ten dip switches located above the PC expansion connector (1:B-1 thru B-4). The TARGA 8 User's Guide notes "switches 1 and 0 must always be set to 1 and switch 9 must be 0". This statement is not correct; in fact, the position of these switches depends on the speed of the computer. For computers operating faster than 6MHz, switches 1 and 0 should be set to 0 and switch 9 should be set to 1.

After installing the configured TARGA board, a camera or VCR can be connected to the VIDEO IN port. The output video signal complies with the National Television Systems Committee (NTSC) standard; consequently, any monochrome composite monitor can be connected to the VIDEO OUT port. Digital RGB computer monitors cannot display the NTSC signal.

Following all hardware connections, initialization software will correctly configure the TARGA framegrabber. The initialization program TRUE_INIT, located on the TARGA 8 Demonstration Diskette, allows selection of several display options. When finished, TRUE_INIT will add command lines to the AUTOEXEC.BAT file. As a result, the TARGA board will be correctly configured every time the computer is turned on.

The framegrabber system is now ready for operation by means of the user friendly program TRUEART.EXE located on the Demonstration diskette. This program allows display of the signal on the VIDEO IN port or a digitized image previously stored on the TARGA framegrabber. A complete description of this software and it's capabilities is available in the User's Guide (1:3-2 thru 3-9).

Finally, digitized images saved in the TARGA format can be converted to encapsulated postscript (EPS) format via software. The program TRUEPS.EXE, located on the Trutilities disk, will perform the necessary conversion. Once a digitized image has been converted to an EPS file, the file is imported to any computer system capable of printing EPS files.

Appendix B. *Binarization Source Code*

The following program is very useful for testing MOSLM binarization methods on an optical bench. A short explanation of the program's features is provided in the introduction of the source code itself. TARGA file formatting is available for use with the MOSLM, another size TARGA file, or a reduced scene binarization. Also, an option for scene average threshold multiplication is allowed.

```
/*******************************************************************
  SLMTGA.C

              FILE FORMATING PROGRAM FOR TARGA FILES
                          by Ken Zabel
                         August 19, 1991

  This is a modification of the code written by John Cline to convert TARGA
  files to a format useable by the Magneto-Optical SLM device.  This code
  uses a 204k TARGA (400x512 picture), converting it to either a 16k TARGA
  (100x128 picture) or a 2k SLM (used with a Magneto-Optic device) file
  format.  The conversion is done using one of three methods.  One of the
  methods used, SCENEAVG, finds the average threshold of the entire scene
  and then binarizes based on that threshold value.  Another method used is
  Clines method (CLINETGA) which binarizes based on an AND condition
  between the local average threshold and the entire scene average
  threshold.  And the last method used is what I called the double average
  method (DBLAVTGA). This method reduces the 400x512 picture to 100x128 by
  finding the average of a 4x4 piece and then binarizing the reduced
  image based on the scene average threshold.
 *******************************************************************/

#include "stdio.h"
#include "string.h"
#include "math.h"
#define CLEAR "\x1B[2J"            /*CLEAR SCREEN*/
#define default1_length 12       /*LENGTH OF DEFAULT1 STRING*/
char inname[80], outname[80];      /*FILE NAMES FOR INPUT AND OUTPUT*/

main()
{
 int i;
 char menu1;

 printf(CLEAR);
 printf("This program will read a 204k TARGA file and\n");
 printf("transform it into a 16k TARGA file or a file to\n");
 printf("be used with the MOSLM \n\n");
 printf("Enter 1 for a file to be used with the MOSLM\n");
 printf("Enter 2 for another size targa file\n");
 printf("Enter 3 for a reduced scene binarization\n");
 printf("(original method)\n");
 menu1 = getche();
 printf("\n");

 switch(menu1)
 {
  case '1':
  printf(CLEAR);
  SLMFILE();
  break;
  case '2':
  printf(CLEAR);
  TGAFILE();
  break;
  case '3':
  printf(CLEAR);
  SCENE();
  break;
  default:
  printf("%c is not a valid choice\n",menu1);
  break;
 } /* end switch */
} /* end main */

SLMFILE()
{
 char menu2;

 printf("\n\nThere are two methods Cline and double average\n");
 printf("\nEnter 1 to do a Cline method");
 printf("\nEnter 2 to do a Double Average method\n");

 menu2 = getche();
 printf("\n");

 switch(menu2)
 {
  case '1':
   printf(CLEAR);
   CLINESLM();
   break;
  case '2':
   printf(CLEAR);
   DBLAVSLM();
   break;
  default:
   printf("%c is an invalid response\n",menu2);
   break;
 } /* end switch */
} /* end SLMFILE */


/*************************CLINESLM()******************************/
```

```
CLINESLM()
{
 int i,j,k,index,r,c,l,m;
 char buff[2048], slm[16][128];
 unsigned char bfin[512][4], bfout[128][128], bfout1[128][128];
 unsigned holder;
 float thrshld=0,multiplier,threshold;
 FILE *file1, *file2;

 printf("Subroutine CLINESLM:  Now opening files\n\n");

 INFILE(inname);     /*CALLING ROUTINES TO GET INPUT  FILENAME*/
 OUTFILE(outname);                    /*OUTPUT*/

 file1=fopen(inname,"rb");        /*OPEN FILES*/
 file2=fopen(outname,"wb"),
 printf("\n");

 printf("The Cline method uses an AND condition between the \n");
 printf("scene average threshold and the localized (3X3)\n");
 printf("threshold to binarize the file.  It also allows for a \n");
 printf("multiplier of the scene average threshold.\n");
 printf("\nEnter the value of the multiplier ");
 printf("(mean=1,1.5 x  mean=1.5):");

 scanf("%f",&multiplier);
 printf("\n\n");

 fseek(file1,18,0);  /*POSITION POINTER IN ORDER TO SKIP HEADER*/

 printf("Reading data\n");
 for(i=14;i<114;i++)          /*READ DATA INTO A BUFFER, 2048 BYTES*/
 {              /*AT A TIME AND DO THIS 100 TIMES   */
  fread(buff,1,2048,file1);
  index = 0;

  for(r=0;r<4;r++)         /*FROM THE 2048 BYTES WE CAN GET */
  {              /*FOUR ROWS AND 512 COLUMNS INTO */
   for(c=0;c<512;c++)      /*THE ARRAY BFIN          */
   {
  bfin[c][r]=buff[index++];
   }
  }

  for(r=0;r<4;r+=4)      /*SUMMING THE ELEMENTS IN A 4X4 BLOCK*/
  {           /*FOR THE ENTIRE 2048 BYTE IN BFIN  */
  for(c=0;c<512;c+=4)
  {
   holder=(bfin[c][r]  +bfin[c+1][r]  +bfin[c+2][r]  +bfin[c+3][r]
     +bfin[c][r+1]+bfin[c+1][r+1]+bfin[c+2][r+1]+bfin[c+3][r+1]
     +bfin[c][r+2]+bfin[c+1][r+2]+bfin[c+2][r+2]+bfin[c+3][r+2]
     +bfin[c][r+3]+bfin[c+1][r+3]+bfin[c+2][r+3]+bfin[c+3][r+2]+4);

   bfout[c/4][i] = holder/16; /*FINDS THE AVERAGE OF THE 4X4 BLOCK*/
  }            /*PLACE AVERAGE VALUE IN ARRAY BFOUT*/
 }
}

for(r=0;r<14;r++)
{
 for(c=0;c<128;c++)
 {
  bfout1[c][r]=0;  /*SET FIRST 14 ROWS IN BFOUT1 TO ZERO*/
 }
}

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  thrshld+=bfout[c][r]; /*SUM ALL THE VALUES OF THE REDUCED PICTURE*/
 }
}

thrshld=multiplier*thrshld/12800;    /*FIND SCENE AVERAGED THRESHOLD*/
printf("Average threshold=%.2f\n",thrshld);

/*BINARIZE BASED ON AN AND CONDITION*/

printf("Binarize based on local and scene averaged threshold\n");

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  threshold=0;
  if(bfout[c][r]≤thrshld)      /*BINARIZE BASED ON SCENE AVERAGE*/
  bfout1[c][r]=1;
   else
  {
   for(l=r-1;l<r+2;l++)
   {
    for(m=c-1;m<c+2;m++)
```

```
      {
       threshold+=bfout[m][l]; /*SUM VALUES IN A 3X3 BLOCK*/
      }
     }
   if(bfout[c][r]≤threshold/9)  /*BINARIZE BASED ON THE 3X3 AVERAGE*/
     bfout1[c][r]=1;
    else
     bfout1[c][r]=0;
    }
   }
 }

 for(r=114;r<128;r++)
 {
  for(c=0;c<128;c++)
  {
   bfout1[c][r]=0;      /*SET LAST 14 ROWS OF BFOUT1 TO ZERO*/
  }
 }

 /*CONVERT 8 PIXELS INTO AN 8 BIT CHAR FOR ADDRESSING THE SLM*/

 printf("Creating SLM format \n");
 for(c=0;c<128;c++)
 {
  for(r=0;r<16;r++)
  {
   slm[15-r][c]=bfout1[c][8*r];
   for(j=1;j<8;j++)
   {
  slm[15-r][c]=2*slm[15-r][c]+bfout1[c][8*r+j];
   }
  }
 }

 printf("Data being written to file"):
 fwrite(slm,2048,1,file2);    /*WRITE TO FILE THE SLM ARRAY*/
 fclose(file1);       /*CLOSE FILES*/
 fclose(file2);

} /* end CLINESLM() */


/******************************DBLAVSLM()*************************/

DBLAVSLM()
{
```

```
int i,j,k,index,r,c;
char buff[2048], slm[16][128];
unsigned char bfin[512][4], bfout[128][128];
unsigned holder;
float thrshld=0,multiplier=.9;
FILE *file1, *file2;

printf("Subroutine DBLAVSLM:  Now opening files\n\n");

INFILE(inname);       /*CALLING ROUTINE TO GET INPUT  FILENAME*/
OUTFILE(outname);                          /*OUTPUT*/

file1=fopen(inname,"rb");  /*OPEN FILES*/
file2=fopen(outname,"wb");
printf("\n");

fseek(file1,18,0);  /*POSITION POINTER IN ORDER TO SKIP HEADER*/

printf("Reading Data\n");

for(i=14;i<114;i++)          /*READ DATA INTO A BUFFER, 2048 BYTES*/
{              /*AT A TIME AND DO THIS 100 TIMES    */
 fread(buff,1,2048,file1);
 index = 0;

 for(r=0;r<4;r++)        /*FROM THE 2048 BYTES WE CAN GET*/
 {              /*FOUR ROWS AND 512 COLUMNS INTO*/
  for(c=0;c<512;c++)   /*THE ARRAY BFIN        */
  {
 bfin[c][r]=buff[index++];
  }
 }

 for(r=0;r<4;r+=4)        /*SUMMING THE ELEMENTS IN A 4X4 BLOCK*/
 {              /*FOR THE ENTIRE 2048 BYTE IN BFIN   */
  for(c=0;c<512;c+=4)
  {
   holder=(bfin[c][r]  +bfin[c+1][r]  +bfin[c+2][r]  +bfin[c+3][r]
     +bfin[c][r+1]+bfin[c+1][r+1]+bfin[c+2][r+1]+bfin[c+3][r+1]
     +bfin[c][r+2]+bfin[c+1][r+2]+bfin[c+2][r+2]+bfin[c+3][r+2]
     +bfin[c][r+3]+bfin[c+1][r+3]+bfin[c+2][r+3]+bfin[c+3][r+2]+4);

   bfout[c/4][i] = holder/16;  /*FINDS THE AVERAGE OF THE 4X4 BLOCK*/
  }              /*PLACE AVERAGE VALUE IN ARRAY BFOUT*/
 }
}
```

```
for(r=0;r<14;r++)
{
 for(c=0;c<128;c++)
 {
  bfout[c][r]=0;      /*SET FIRST 14 ROWS IN BFOUT1 TO ZERO*/
 }
}

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  thrshld+=bfout[c][r];  /*SUM ALL THE VALUES OF THE REDUCED PICTURE*/
 }
}

thrshld=multiplier*thrshld/12800;   /*FIND SCENE AVERAGED THRESHOLD*/
printf("Average threshold=%.2f\n",thrshld);

printf("Binarize based on scene average");

for(r=14;r<114;r++)      /*BINARIZE ON SCENE AVERAGE*/
{
 for(c=0;c<128;c++)
 {
  if(bfout[c][r]<thrshld)
 bfout[c][r]=0:
   else
 bfout[c][r]=1;
 }
}

for(r=114;r<128;r++)
{
 for(c=0;c<128;c++)
 {
  bfout[c][r]=0;   /*SET LAST 14 ROWS EQUAL TO ZERO*/
 }
}

/*CONVERT 8 PIXELS INTO AN 8 BIT CHAR FOR ADDRESSING THE SLM*/

printf("\nCreating SLM format \n");
for(c=0;c<128;c++)
{
 for(r=0;r<16;r++)
 {
   slm[15-r][c]=bfout[c][8*r];
   for(j=1;j<8;j++)
   {
  slm[15-r][c]=2*slm[15-r][c]+bfout[c][8*r+j];
   }
  }
 }

 printf("Data being written to file\n");

 fwrite(slm,2048,1,file2);   /*WRITE TO FILE THE SLM ARRAY*/
 fclose(file1);        /*CLOSE FILES*/
 fclose(file2);

} /* end DBLAVSLM() */


/****************************TGAFILE()*******************/

TGAFILE()
{
 char menu3;

 printf("\n\n");
 printf("There are two methods we have used, Cline and \n");
 printf("Double average.  Cline's method involves ANDING the\n");
 printf("scene threshold with the local 3X3 threshold.  The\n");
 printf("double average method reduces the original 204k targa");
 printf("file (400X512) to a 16k targa file (100X128)\n");
 printf("by taking a 4X4 block and averaging it to one pixel,\n");
 printf("it then does a scene average on the 100X128 scene and ");
 printf("uses this threshold\n");
 printf("to binarize the scene\n\n");

 printf("Enter 1 to do a Cline's method\n");
 printf("Enter 2 to do a Double average method\n");
 menu3 = getche();
 printf("\n");

  switch(menu3)
  {
   case '1':
   printf(CLEAR);
   CLINETGA();
   break;
   case '2':
   printf(CLEAR);
   DBLAVTGA();
   break;
   default:
   printf("%c is an invalid response\n",menu3);
  } /* end switch */
} /* end TGAFILE */


/******************************DBLAVTGA()************************/

DBLAVTGA()
{
 int i,j,k,index,r,c;
 char buff[2048];
 unsigned char bfin[512][4], bfout[128][128];
 unsigned holder;
 float thrshld=0,multiplier;
 static char tgbu[] = {'\x00','\x00','\x03','\x00','\x00','\x00',
      '\x00','\x00','\x00','\x00','\x00','\x00',
      '\x80','\x00','\x80','\x00','\x08','\x00'};
 FILE *file1, *fiie2, *file3;

 printf("Subroutine DBLAVTGA:  Now opening files\n\n");

 INFILE(inname);       /*CALLING ROUTINE TO GET INPUT  FILENAME*/
 OUTFILE(outname);                /*OUTPUT*/

 file1=fopen(inname,"rb");  /*OPEN FILES*/
 file2=fopen(outname,"wb");
 printf("\n");

 fwrite(tgbu,18,1,file2);  /*WRITE HEADER TO THE NEW TARGA FILE*/

 printf("This is for a tga file using double average\n");
 printf("Enter a multiplier (mean = 1): \n");
 scanf("%f",&multiplier);
 printf("\n");

 fseek(file1,18,0);  /*POSITION POINTER IN ORDER TO SKIP HEADER*/

 printf("Reading Data\n");
 for(i=14;i<114;i++)       /*READ DATA INTO A BUFFER, 2048 BYTES*/
 {              /*AT A TIME AND DO THIS 100 TIMES    */
  fread(buff,1,2048,file1);
  index = 0;

  for(r=0;r<4;r++)       /*FROM THE 2048 BYTES WE CAN GET*/
```

```
{           /*FOUR ROWS AND 512 COLUMNS INTO*/
 for(c=0;c<512;c++)  /*THE ARRAY BFIN      */
 {
bfin[c][r]=buff[index++];
 }
}

for(r=0;r<4;r+=4)       /*SUMMING THE ELEMENTS IN A 4X4 BLOCK*/
{           /*FOR THE ENTIRE 2048 BYTE IN BFIN  */
 for(c=0;c<512;c+=4)
 {
  holder=(bfin[c][r]  +bfin[c+1][r]  +bfin[c+2][r]  +bfin[c+3][r]
    +bfin[c][r+1]+bfin[c+1][r+1]+bfin[c+2][r+1]+bfin[c+3][r+1]
    +bfin[c][r+2]+bfin[c+1][r+2]+bfin[c+2][r+2]+bfin[c+3][r+2]
    +bfin[c][r+3]+bfin[c+1][r+3]+bfin[c+2][r+3]+bfin[c+3][r+2]+4);

  bfout[c/4][i] = holder/16;  /*FINDS THE AVERAGE OF THE 4X4 BLOCK*/
 }                      /*PLACE AVERAGE VALUE IN ARRAY BFOUT*/
}
}

for(r=0;r<14;r++)
{
 for(c=0;c<128;c++)
 {
  bfout[c][r]=0x00;       /*SET FIRST 14 ROWS IN BFOUT1 TO ZERO*/
 }
}

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  thrshld+=bfout[c][r];  /*SUM ALL THE VALUES OF THE REDUCED PICTURE*/
 }
}

thrshld=multiplier*thrshld/12800;  /*FIND SCENE AVERAGED THRESHOLD*/
printf("Average threshold=%.2f\n",thrshld);

printf("Binarize based on scene average\n");

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  if(bfout[c][r]<thrshld)  /*BINDARIZE BASED ON SCENE AVERAGE*/
 bfout[c][r]=0x00;
   else
 bfout[c][r]=0xFF;
 }
}

for(r=114;r<128;r++)
{
 for(c=0;c<128;c++)
 {
  bfout[c][r]=0x00;      /*SET LAST 14 ROWS OF BFOUT TO ZERO*/
 }
}

printf("Data being written to a file\n");

fwrite(bfout,16384,1,file2);  /*WRITE TO FILE THE ARRAY BFOUT*/
fclose(file1);      /*CLOSE FILES*/
fclose(file2);

} /* end DBLAVTGA() */


/***************************CLINETGA()**************************/

CLINETGA()
{
 int i,j,k,index,r,c,l,m;
 char buff[2048];
 unsigned char bfin[512][4], bfout[128][128], bfout1[128][128];
 unsigned holder;
 float thrshld=0,multiplier,threshold;
 static char tgbu[] = {'\x00','\x00','\x03','\x00','\x00','\x00',
      '\x00','\x00','\x00','\x00','\x00','\x00',
      '\x80','\x00','\x80','\x00','\x08','\x00'};

 FILE *file1, *file2, *file3;

 printf("Subroutine CLINETGA: Now opening files\n\n");

 INFILE(inname);     /*CALLING ROUTINE TO GET INPUT  FILENAME*/
 OUTFILE(outname);                    /*OUTPUT*/

 file1=fopen(inname,"rb");  /*OPEN FILES*/
 file2=fopen(outname,"wb");
 printf("\n");
```

```
fwrite(tgbu,18,1,file2);  /*WRITE HEADER TO THE NEW TARGA FILE*/

printf("The Cline method uses an AND condition between\n");
printf("the scene average threshold and the localized (3X3)\n");
printf("threshold to binarize the file.  It also allows for\n");
printf("a multiplier of the scene average threshold.\n");
printf("\nEnter the value of the multiplier ");
printf("(mean=1, 1.5 times mean=1.5):");
scanf("%f",&multiplier);

fseek(file1,18,0);  /*POSITION POINTER IN ORDER TO SKIP HEADER*/

printf("Reading Data\n");

for(i=14;i<114;i++)      /*READ DATA INTO A BUFFER, 2048 BYTES*/
{            /*AT A TIME AND DO THIS 100 TIMES  */
 fread(buff.1.2048.file1);
 index = 0;

 for(r=0;r<4;r++)        /*FROM THE 2048 BYTES WE CAN GET*/
 {            /*FOUR ROWS AND 512 COLUMNS INTO*/
  for(c=0;c<512;c++)      /*THE ARRAY BFIN          */
  {
 bfin[c][r]=buff[index++];
  }
 }

 for(r=0;r<4;r+=4)      /*SUMMING THE ELEMENTS IN A 4X4 BLOCK*/
 {            /*FOR THE ENTIRE 2048 BYTE IN BFIN  */
  for(c=0;c<512;c+=4)
  {
  holder=(bfin[c][r] +bfin[c+1][r] +bfin[c+2][r] +bfin[c+3][r]
    +bfin[c][r+1]+bfin[c+1][r+1]+bfin[c+2][r+1]+bfin[c+3][r+1]
    +bfin[c][r+2]+bfin[c+1][r+2]+bfin[c+2][r+2]+bfin[c+3][r+2]
    +bfin[c][r+3]+bfin[c+1][r+3]+bfin[c+2][r+3]+bfin[c+3][r+2]+4);

  bfout[c/4][i] = holder/16;  /*FINDS THE AVERAGE OF THE 4X4 BLOCK*/
  }            /*PLACE AVERAGE VALUE IN ARRAY BFOUT*/
 }
}

for(r=0;r<14;r++)
{
 for(c=0;c<128;c++)
 {
  bfout1[c][r]=0x00;  /*SET FIRST 14 ROWS IN BFOUT1 TO ZERO*/
 }
```

```
}

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  thrshld+=bfout[c][r];    /*SUM THE VALUES OF THE REDUCED PICTURE*/
 }
}

thrshld=multiplier*thrshld/12800;  /*FIND SCENE AVERAGED THRESHOLD*/
printf("Average threshold=%.2f\n",thrshld);

/*BINARIZE BASED ON AN AND CONDITION*/

printf("Binarize based on scene and local average threshold\n");

for(r=14;r<114;r++)
{
 for(c=0;c<128;c++)
 {
  threshold=0;
  if(bfout[c][r]≤thrshld)       /*BINARIZE BASED ON SCEN AVERAGE*/
  bfout1[c][r]=0x00;
   else
  {
   for(l=r-1;l<r+2;l++)
   {
    for(m=c-1;m<c+2;m++)
    {
     threshold+=bfout[m][l];   /*SUM VALUES IN A 3X3 BLOCK*/
    }
   }
  if(bfout[c][r]≤threshold/9)   /*BINARIZE BASED ON THE 3X3 AVERAGE*/
    bfout1[c][r]=0x00;
  else
    bfout1[c][r]=0xFF;
  }
 }
}

for(r=114;r<128;r++)
{
 for(c=0;c<128;c++)
 {
  bfout1[c][r]=0x00;   /*SET LAST 14 ROWS OF BFOUT1 TO ZERO*/
 }
```

```
  }

 printf("Data being written to file\n");

 fwrite(bfout1,16384,1,file2);  /*WRITE TO FILE THE ARRAY BFOUT1*/
 fclose(file1);        /*CLOSE FILES*/
 fclose(file2);

} /* end CLINETGA */


/**********************************SCENE()****************/

SCENE()
{
  char menu4;

  printf("In this method we simply binarize based on the\n");
  printf("reduced scene averaged threshold.\n");
  printf("\nEnter 1 to process an SLM file");
  printf("\nEnter 2 to process a TGA file");
  menu4 = getche();
  printf("\n");

  switch(menu4)
  {
   case '1':
     printf(CLEAR);
     SCENESLM();
     break;
   case '2':
     printf(CLEAR);
     SCENETGA();
     break;
   default:
     printf("%c is an invalid response\n",menu4);
     break;
  } /* end switch */
} /* end SCENE */


/**********************************SCENESLM()************/

SCENESLM()
{
  unsigned temp[128][128];
  int i, j, x, y, col, row, index;
```

```
char slm[16][128], targabuffer[2048];
float thrshld=0;
FILE *fr1, *fr2;

printf("Subroutine SCENESLM:  Now opening files\n\n");

INFILE(inname);          /*CALLING ROUTINE TO GET INPUT  FILENAME*/
OUTFILE(outname);                         /*OUTPUT*/

fr1=fopen(inname,"rb");  /*OPEN FILES*/
fr2=fopen(outname,"wb");
printf("\n");

printf("reading in data \n");

fseek(fr1,18,0);  /*POSITION POINTER IN ORDER TO SKIP HEADER*/

printf("Initializing the array TEMP\n");

for(row=0;row<128;row++)
{
 for(col=0;col<128;col++)
 {
  temp[col][row]=0;      /*INITIALIZE THE ARRAY TEMP*/
 }
}

printf("Reading Data and summing elements\n");

for(row=14;row<114;row++)          /*RETRIEVE DATA SEQUENTIALLY INTO*/
{                         /*100x128 ARRAY            */
  fread(targabuffer,1,2048,fr1);  /*READ FOUR ROWS OF CCD DATA*/
  index=0;
  for(j=0;j<4;j++)        /*SUM 4 CCD ROWS*/
  {
 for(col=0;col<128;col++)
 {
   for(i=0;i<4;i++)       /*SUM 4 CCD COLUMNS*/
   {
     temp[col][row]=targabuffer[index++]+temp[col][row];
   }
 }
  }
  for(col=0;col<128;col++)
  {
 thrshld+=temp[col][row];     /*SUM ALL VALUES OF TEMP ARRAY*/
  }
```

```
}

/*CALCULATE MEAN OF PIXEL VALUES TO USE AS A THRESHOLD*/

printf("calculating threshold \n");
thrshld = thrshld/12800;
printf("Average threshold = %.2f\n",thrshld);

/*BINARIZE PIXELS BASED ON THRESHOLD*/

printf("Binarize based on scene average threshold\n");

for(row=0;row<14;row++)
{
  for(col=0;col<128;col++)
  {
 temp[col][row]=0;      /*SET FIRST 14 ROWS EQUAL TO ZERO*/
  }
}

for(row=14;row<114;row++)
{
  for(col=0;col<128;col++)
  {
 if(temp[col][row]<thrshld)    /*BINARIZE BASED ON SCENE*/
   temp[col][row]=0;      /*AVERAGED THRESHOLD    */
  else
   temp[col][row]=1;
  }
}

for(row=114;row<128;row++)
{
  for(col=0;col<128;col++)
  {
 temp[col][row]=0;      /*SET LAST 14 ROWS TO ZERO*/
  }
}

/*CONVERT 8 PIXELS INTO AN 8 BIT CHAR FOR ADDRESSING THE SLM*/

printf("Creating SLM format \n");
for(x=0;x<128;x++)
{
  for(y=0;y<16;y++)
  {
 slm[15-y][x]=temp[x][8*y];
 for(j=1;j<8;j++)
 {
  slm[15-y][x]=2*slm[15-y][x]+temp[x][8*y+j];
 }
  }
 }

 printf("Data being written to file\n");

 fwrite(slm,sizeof(slm),1,fr2);   /* WRITE TO FILE THE ARRAY SLM*/
 fclose(fr1);             /*CLOSE FILE*/
 fclose(fr2);

} /* end SCENESLM */


/**********************************SCENETGA()*********************/

SCENETGA()
{
 int i, j, x, y, r, c, col, row, index;
 unsigned temp[128][128];
 char outb[128][128], targabuffer[2048];
 float thrshld=0;
 static char tgbu[] = {'\x00','\x00','\x03','\x00','\x00','\x00',
       '\x00','\x00','\x00','\x00','\x00','\x00',
       '\x80','\x00','\x80','\x00','\x08','\x00'};
 FILE *fr1, *fr2;

 printf("Subroutine SCENETGA:  Now opening files\n\n");

 INFILE(inname);       /*CALLING ROUTINE TO GET INPUT  FILENAME*/
 OUTFILE(outname);                     /*OUTPUT*/

 fr1=fopen(inname,"rb");  /*OPEN FILES*/
 fr2=fopen(outname,"wb");
 printf("\n");

 fwrite(tgbu,18,1,fr2);    /* WRITE HEADER TO THE NEW TARGA FILE*/

 fseek(fr1,18,0);  /*POSITION POINTER IN ORDER TO SKIP HEADER*/

 printf("Initializing array TEMP\n");

 for(r=0:r<128;r++)
 {
  for(c=0:c<128:c++)
  {
   temp[c][r]=0;   /*INITIALIZE THE TEMP ARRAY*/
  }
 }

printf("Reading Data and Summing elements\n");

/*RETRIEVE DATA SEQUENTIALLY INTO 100X128 ARRAY*/
for(row=14;row<114;row++)
{
  fread(targabuffer,1,2048,fr1);   /*READ FOUR ROWS OF CCD DATA*/
  index=0;
  for(j=0;j<4;j++)           /*SUM 4 CCD ROWS*/
  {
 for(col=0;col<128;col++)
 {
   for(i=0;i<4;i++)          /*SUM 4 CCD COLUMNS*/
   {
     temp[col][row]+=targabuffer[index++];
   }
 }
  }
  for(col=0;col<128;col++)
  {
 thrshld+=temp[col][row];   /*SUM ALL VALUES OF TEMP ARRAY*/
  }
}

/*CALCULATE MEAN OF PIXEL VALUES TO USE AS A THRESHOLD*/

printf("calculating threshold \n");
thrshld = thrshld/12800;
printf("Average threshold = %.2f\n",thrshld);

/*BINARIZE PIXELS BASED ON THRESHOLD*/

printf("Binarize based on the scene averaged threshold\n");

for(r=0;r<14;r++)
{
 for(c=0;c<128;c++)
 {
  outb[c][r]=0x00;          /*SET FIRST 14 ROWS EQUAL TO ZERO*/
 }
}

for(r=14;r<114;r++)
```

```
 {
  for(c=0;c<128;c++)
  {
   if(temp[c][r]<thrshld)       /*BINARIZE BASED ON THE SCENE*/
  outb[c][r]=0x00;       /*AVERAGED THRESHOLD        */
  else
  outb[c][r]=0xFF;
  }
 }

for(r=114;r<128;r++)
{
 for(c=0;c<128;c++)
 {
  outb[c][r]=0x00;             /*SET LAST 14 ROWS EQUAL TO ZERO*/
 }
}

 printf("Data being written to file\n");

 fwrite(outb,sizeof(outb),1,fr2);   /* WRITE TO FILE THE ARRAY OUTB*/
 fclose(fr1);              /*CLOSE FILES*/
 fclose(fr2);

} /* end SCENETGA */


/********************************INFILE()***************************/

INFILE(cptr)
char *cptr;
{
  static char default1[]={"j:lenna.tga"};  /*A DEFAULT INPUT FILENAME*/
  char x;
  int i=0,k,j;

  printf("Enter the entire filename and path for the \n ");
  printf("input file(<Return> is defaulted to J:\lenna.tga):\n");

  do        /*READS IN A FILENAME AND PATH UP TO 81 CHARACTERS*/
  {        /*OR UNTIL IT READS A RETURN CHARACTER*/
   i++;
   k=i-1;
   x=getchar();
   if(k<31)     /*THE POINTER CPTR INCREMENTS TO EACH ELEMENT OF ARRAY*/
  *(cptr+k)=x;
  } while(x ≠ '\n');   /*AFTER READING A RETURN THE LOOP ENDS*/
 *(cptr+k)='\0';  /*PLACES A NULL CHARACTER IN THE LAST POSITION */
      /*OF THE STRING POINTED TO BY CPTR*/

      /*IF THE FIRST CHARACTER POINTED TO IN THE STRING*/
 if(*cptr=='\0') /*IS THE NULL CHARACTER THEN USE THE DEFAULT FILENAME*/
  {
   for(j=0;j<default1_length;j++)
   *(cptr+j)=default1[j]; /*INCREMENT POINTER THROUGH EACH ELEMENT IN*/
  }           /*THE ARRAY AND EQUATE IT TO THE SAME ELEMENT*/
          /*IN THE DEFAULT1 ARRAY*/
} /* end INFILE() */


/********************************OUTFILE()************************/

OUTFILE(cptr)
char *cptr;
{
 char x;
 int i=0,k;

 printf("Enter the entire filename and path for the\n");
 printf(" output file:\n");

      /*READS IN A FILENAME AND PATH UP TO 81 CHARACTERS*/
 do        /*OR UNTIL IT READS A RETURN CHARACTER*/
 {
  i++;
  k=i-1;
  x=getchar();
  if(k<81)       /*THE POINTER CPTR INCREMENTS TO EACH ELEMENT OF ARRAY*/
 *(cptr+k)=x;
 } while(x ≠ '\n');  /*AFTER READING A RETURN THE LOOP ENDS*/

 *(cptr+k)='\0';  /*PLACES A NULL CHARACTER IN THE LAST POSITION */
      /*OF THE STRING POINTED TO BY CPTR*/
} /* end OUTFILE() */
æ
```

Appendix C. *Wavelet Source Code*

All wavelets displayed on the MOSLM were written in the Turbo C programming language. Source code is provided for wavelet sizes of 128 × 128, 64 × 64, 32 × 32, 16 × 16, 8 × 8, and 4 × 4 pixels. Through careful review of these programs any wavelet size can easily be programmed.

In all of the following programs, wavelet +1 and -1 states are written prior to programming the zero "worm" state. Additionally, notice there in no saturation pulse (hex B801) written after the zero state.

As mentioned in section 3.1.2, the optical wavelets are written to the MOSLM in 16 separate byte rows. These byte rows are programmed from top-left to bottom-right. Writing wavelet data as byte rows instead of byte columns minimized random bad pixels.

```
/***********************************************************************
 wavlt128.c

          AIR FORCE INSTITUTE OF TECHNOLOGY
       WAVELET PATTERN GENERATOR FOR THE SLM
                  by Capt Steve Pinski
                     July 8, 1991

 This program will display a vertical Harr wavelet pattern on the
 SLM. The 2-dimentional wavelet is 128 x 128 pixels .
***********************************************************************/

#include "stdio.h"
#include "string.h"
#include "time.h"
unsigned r, q;

main()
{
  char far *semetex;
   semetex = (char far *) 0xb0000000;        /*SLM is at b000 segment*/
   *(semetex+0x800)=1;                       /*erase SLM*/
   wait(1000);                               /*wait or lose pixels*/

  /*write rows for +1 state*/
   for(q=0x0; q<0x40; q++)
   {
     *(semetex+q+128*0)=0x3f;
    *(semetex+q+128*0)=0xcf;
    *(semetex+q+128*0)=0xf3;
    *(semetex+q+128*0)=0xfc;
    *(semetex+q+128*1)=0x3f;
    *(semetex+q+128*1)=0xcf;
    *(semetex+q+128*1)=0xf3;
    *(semetex+q+128*1)=0xfc;
    *(semetex+q+128*2)=0x3f;
    *(semetex+q+128*2)=0xcf;
    *(semetex+q+128*2)=0xf3;
    *(semetex+q+128*2)=0xfc;
    *(semetex+q+128*3)=0x3f;
    *(semetex+q+128*3)=0xcf;
    *(semetex+q+128*3)=0xf3;
    *(semetex+q+128*3)=0xfc;
    *(semetex+q+128*4)=0x3f;
    *(semetex+q+128*4)=0xcf;
    *(semetex+q+128*4)=0xf3;
    *(semetex+q+128*4)=0xfc;
```

```
        *(semetex+q+128*5)=0x3f;
        *(semetex+q+128*5)=0xcf;
        *(semetex+q+128*5)=0xf3;
        *(semetex+q+128*5)=0xfc;
        *(semetex+q+128*6)=0x3f;
        *(semetex+q+128*6)=0xcf;
        *(semetex+q+128*6)=0xf3;
        *(semetex+q+128*6)=0xfc;
        *(semetex+q+128*7)=0x3f;
        *(semetex+q+128*7)=0xcf;
        *(semetex+q+128*7)=0xf3;
        *(semetex+q+128*7)=0xfc;
        *(semetex+q+128*8)=0x3f;
        *(semetex+q+128*8)=0xcf;
        *(semetex+q+128*8)=0xf3;
        *(semetex+q+128*8)=0xfc;
        *(semetex+q+128*9)=0x3f;
        *(semetex+q+128*9)=0xcf;
        *(semetex+q+128*9)=0xf3;
        *(semetex+q+128*9)=0xfc;
        *(semetex+q+128*10)=0x3f;
        *(semetex+q+128*10)=0xcf;
        *(semetex+q+128*10)=0xf3;
        *(semetex+q+128*10)=0xfc;
        *(semetex+q+128*11)=0x3f;
        *(semetex+q+128*11)=0xcf;
        *(semetex+q+128*11)=0xf3;
        *(semetex+q+128*11)=0xfc;
        *(semetex+q+128*12)=0x3f;
        *(semetex+q+128*12)=0xcf;
        *(semetex+q+128*12)=0xf3;
        *(semetex+q+128*12)=0xfc;
        *(semetex+q+128*13)=0x3f;
        *(semetex+q+128*13)=0xcf;
        *(semetex+q+128*13)=0xf3;
        *(semetex+q+128*13)=0xfc;
        *(semetex+q+128*14)=0x3f;
        *(semetex+q+128*14)=0xcf;
        *(semetex+q+128*14)=0xf3;
        *(semetex+q+128*14)=0xfc;
        *(semetex+q+128*15)=0x3f;
        *(semetex+q+128*15)=0xcf;
        *(semetex+q+128*15)=0xf3;
        *(semetex+q+128*15)=0xfc;

}
*(semetex+0x801)=1;                /*write pulse for saturation*/
```

```
}

/** ** ** ** ** ** ** ** *** ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.                            */

wait(t)
{
   time_t start, finish;
   time(&start);
   time(&finish);
   while(+difftime(finish, start) < t/1000)
   {
      time(&finish);
   }
}
/*******************************************************************/
```

```
/*****************************************************************
wavlt64.c

        AIR FORCE INSTITUTE OF TECHNOLOGY
     WAVELET PATTERN GENERATOR FOR THE SLM
                by Capt Steve Pinski
                   July 16, 1991

This program will display a Harr wavelet pattern on the SLM.
The 2-dimentional wavelet is 64 x 64 pixels.
*****************************************************************/

#include "stdio.h"
#include "string.h"
#include "time.h"
unsigned r, q;

main()
{
  char far *semetex;
  semetex = (char far *) 0xb0000000;            /*SLM is at b000 segment*/
  *(semetex+0x800)=1;                           /*erase SLM*/
  wait(1000);                                   /*wait or lose pixels*/

  /*write rows of wavelet in +1 state*/
  for(q=0x20; q<0x40; q++)
  {
    *(semetex+q+128*4)=0x3f;
    *(semetex+q+128*4)=0xcf;
    *(semetex+q+128*4)=0xf3;
    *(semetex+q+128*4)=0xfc;
    *(semetex+q+128*5)=0x3f;
    *(semetex+q+128*5)=0xcf;
    *(semetex+q+128*5)=0xf3;
    *(semetex+q+128*5)=0xfc;
    *(semetex+q+128*6)=0x3f;
    *(semetex+q+128*6)=0xcf;
    *(semetex+q+128*6)=0xf3;
    *(semetex+q+128*6)=0xfc;
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;
    *(semetex+q+128*8)=0x3f;
    *(semetex+q+128*8)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
  *(semetex+q+128*9)=0x3f;
  *(semetex+q+128*9)=0xcf;
  *(semetex+q+128*9)=0xf3;
  *(semetex+q+128*9)=0xfc;
  *(semetex+q+128*10)=0x3f;
  *(semetex+q+128*10)=0xcf;
  *(semetex+q+128*10)=0xf3;
  *(semetex+q+128*10)=0xfc;
  *(semetex+q+128*11)=0x3f;
  *(semetex+q+128*11)=0xcf;
  *(semetex+q+128*11)=0xf3;
  *(semetex+q+128*11)=0xfc;
}
wait(3000);

 *(semetex+0x801)=1;            /*write pulse for contrast*/

wait(3000);                     /*wait or lose pixels*/

/*write rows above wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
  *(semetex+q+128*0)=0x3f;
  *(semetex+q+128*0)=0xcf;
  *(semetex+q+128*0)=0xf3;
  *(semetex+q+128*0)=0xfc;
  *(semetex+q+128*1)=0x3f;
  *(semetex+q+128*1)=0xcf;
  *(semetex+q+128*1)=0xf3;
  *(semetex+q+128*1)=0xfc;
  *(semetex+q+128*2)=0x3f;
  *(semetex+q+128*2)=0xcf;
  *(semetex+q+128*2)=0xf3;
  *(semetex+q+128*2)=0xfc;
  *(semetex+q+128*3)=0x3f;
  *(semetex+q+128*3)=0xcf;
  *(semetex+q+128*3)=0xf3;
  *(semetex+q+128*3)=0xfc;

 }
/*write rows below wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
  *(semetex+q+128*12)=0x3f;
  *(semetex+q+128*12)=0xcf;
  *(semetex+q+128*12)=0xf3;
  *(semetex+q+128*12)=0xfc;
  *(semetex+q+128*13)=0x3f;
  *(semetex+q+128*13)=0xcf;
  *(semetex+q+128*13)=0xf3;
  *(semetex+q+128*13)=0xfc;
  *(semetex+q+128*14)=0x3f;
  *(semetex+q+128*14)=0xcf;
  *(semetex+q+128*14)=0xf3;
  *(semetex+q+128*14)=0xfc;
  *(semetex+q+128*15)=0x3f;
  *(semetex+q+128*15)=0xcf;
  *(semetex+q+128*15)=0xf3;
  *(semetex+q+128*15)=0xfc;

 }
/*write rows left of wavelet in 0 state*/
 for(q=0x0; q<0x20; q++)
 {
  *(semetex+q+128*4)=0x3f;
  *(semetex+q+128*4)=0xcf;
  *(semetex+q+128*4)=0xf3;
  *(semetex+q+128*4)=0xfc;
  *(semetex+q+128*5)=0x3f;
  *(semetex+q+128*5)=0xcf;
  *(semetex+q+128*5)=0xf3;
  *(semetex+q+128*5)=0xfc;
  *(semetex+q+128*6)=0x3f;
  *(semetex+q+128*6)=0xcf;
  *(semetex+q+128*6)=0xf3;
  *(semetex+q+128*6)=0xfc;
  *(semetex+q+128*7)=0x3f;
  *(semetex+q+128*7)=0xcf;
  *(semetex+q+128*7)=0xf3;
  *(semetex+q+128*7)=0xfc;
  *(semetex+q+128*8)=0x3f;
  *(semetex+q+128*8)=0xcf;
  *(semetex+q+128*8)=0xf3;
  *(semetex+q+128*8)=0xfc;
  *(semetex+q+128*9)=0x3f;
  *(semetex+q+128*9)=0xcf;
  *(semetex+q+128*9)=0xf3;
  *(semetex+q+128*9)=0xfc;
  *(semetex+q+128*10)=0x3f;
  *(semetex+q+128*10)=0xcf;
  *(semetex+q+128*10)=0xf3;
  *(semetex+q+128*10)=0xfc;
  *(semetex+q+128*11)=0x3f;
  *(semetex+q+128*11)=0xcf;
```

```
    *(semetex+q+128*11)=0xf3;
    *(semetex+q+128*11)=0xfc;
   }
  /*write rows right of wavelet in 0 state*/
   for(q=0x60; q<0x80; q++)
   {
    *(semetex+q+128*4)=0x3f;
    *(semetex+q+128*4)=0xcf;
    *(semetex+q+128*4)=0xf3;
    *(semetex+q+128*4)=0xfc;
    *(semetex+q+128*5)=0x3f;
    *(semetex+q+128*5)=0xcf;
    *(semetex+q+128*5)=0xf3;
    *(semetex+q+128*5)=0xfc;
    *(semetex+q+128*6)=0x3f;
    *(semetex+q+128*6)=0xcf;
    *(semetex+q+128*6)=0xf3;
    *(semetex+q+128*6)=0xfc;
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;
    *(semetex+q+128*8)=0x3f;
    *(semetex+q+128*8)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
    *(semetex+q+128*9)=0x3f;
    *(semetex+q+128*9)=0xcf;
    *(semetex+q+128*9)=0xf3;
    *(semetex+q+128*9)=0xfc;
    *(semetex+q+128*10)=0x3f;
    *(semetex+q+128*10)=0xcf;
    *(semetex+q+128*10)=0xf3;
    *(semetex+q+128*10)=0xfc;
    *(semetex+q+128*11)=0x3f;
    *(semetex+q+128*11)=0xcf;
    *(semetex+q+128*11)=0xf3;
    *(semetex+q+128*11)=0xfc;
   }

}

/** ** ** ** ** ** ** ** * * * ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.

wait(t)
```

```
{
   time_t start, finish;
   time(&start);
   time(&finish);
   while(+difftime(finish, start) < t/1000)
   {
      time(&finish);
   }
}
/*************************************************************  *******/
```

```
 *(semetex+0x801)=1;                 /*write pulse for contrast*/

wait(1000);                          /*wait or lose pixels*/

/*write rows above wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
  *(semetex+q+128*0)=0x3f;
  *(semetex+q+128*0)=0xcf;
  *(semetex+q+128*0)=0xf3;
  *(semetex+q+128*0)=0xfc;
  *(semetex+q+128*1)=0x3f;
  *(semetex+q+128*1)=0xcf;
  *(semetex+q+128*1)=0xf3;
  *(semetex+q+128*1)=0xfc;
  *(semetex+q+128*2)=0x3f;
  *(semetex+q+128*2)=0xcf;
  *(semetex+q+128*2)=0xf3;
  *(semetex+q+128*2)=0xfc;
  *(semetex+q+128*3)=0x3f;
  *(semetex+q+128*3)=0xcf;
  *(semetex+q+128*3)=0xf3;
  *(semetex+q+128*3)=0xfc;
  *(semetex+q+128*4)=0x3f;
  *(semetex+q+128*4)=0xcf;
  *(semetex+q+128*4)=0xf3;
  *(semetex+q+128*4)=0xfc;
  *(semetex+q+128*5)=0x3f;
  *(semetex+q+128*5)=0xcf;
  *(semetex+q+128*5)=0xf3;
  *(semetex+q+128*5)=0xfc;
 }
/*write rows below wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
  *(semetex+q+128*10)=0x3f;
  *(semetex+q+128*10)=0xcf;
  *(semetex+q+128*10)=0xf3;
  *(semetex+q+128*10)=0xfc;
  *(semetex+q+128*11)=0x3f;
  *(semetex+q+128*11)=0xcf;
  *(semetex+q+128*11)=0xf3;
  *(semetex+q+128*11)=0xfc;
  *(semetex+q+128*12)=0x3f;
  *(semetex+q+128*12)=0xcf;
  *(semetex+q+128*12)=0xf3;
```

```
    *(semetex+q+128*12)=0xfc;
    *(semetex+q+128*13)=0x3f;
    *(semetex+q+128*13)=0xcf;
    *(semetex+q+128*13)=0xf3;
    *(semetex+q+128*13)=0xfc;
    *(semetex+q+128*14)=0x3f;
    *(semetex+q+128*14)=0xcf;
    *(semetex+q+128*14)=0xf3;
    *(semetex+q+128*14)=0xfc;
    *(semetex+q+128*15)=0x3f;
    *(semetex+q+128*15)=0xcf;
    *(semetex+q+128*15)=0xf3;
    *(semetex+q+128*15)=0xfc;

   }
  /*write rows left of wavelet in 0 state*/
   for(q=0x0; q<0x30; q++)
   {
    *(semetex+q+128*6)=0x3f;
    *(semetex+q+128*6)=0xcf;
    *(semetex+q+128*6)=0xf3;
    *(semetex+q+128*6)=0xfc;
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;
    *(semetex+q+128*8)=0x3f;
    *(semetex+q+128*8)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
    *(semetex+q+128*9)=0x3f;
    *(semetex+q+128*9)=0xcf;
    *(semetex+q+128*9)=0xf3;
    *(semetex+q+128*9)=0xfc;
   }
  /*write rows right of wavelet in 0 state*/
   for(q=0x50; q<0x80; q++)
   {
    *(semetex+q+128*6)=0x3f;
    *(semetex+q+128*6)=0xcf;
    *(semetex+q+128*6)=0xf3;
    *(semetex+q+128*6)=0xfc;
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;
    *(semetex+q+128*8)=0x3f;
    *(semetex+q+128*8)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
    *(semetex+q+128*9)=0x3f;
    *(semetex+q+128*9)=0xcf;
    *(semetex+q+128*9)=0xf3;
    *(semetex+q+128*9)=0xfc;
  }

}

/** ** ** ** ** ** ** ** *** ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.

wait(t)
{
  time_t start, finish;
  time(&start);
  time(&finish);
  while(+difftime(finish, start) < t/1000)
  {
    time(&finish);
  }
}
/*****************************************************************/
```

```
/*********************************************************************
 wavlt16.c

         AIR FORCE INSTITUTE OF TECHNOLOGY
       WAVELET PATTERN GENERATOR FOR THE SLM
                  by Capt Steve Pinski
                    July 24, 1991

 This program will display a Harr wavelet pattern on the SLM.
 The 2-dimentional wavelet is 16 x 16 pixels.
 *********************************************************************/

#include "stdio.h"
#include "string.h"
#include "time.h"
unsigned r, q;

main()
{
  char far *semetex;
   semetex = (char far *) 0xb0000000;          /*SLM is at b000 segment*/
   *(semetex+0x800)=1;                         /*erase SLM*/
   wait(1000);                                 /*wait or lose pixels*/

   /*write 16 rows for wavelet*/
   for(q=0x38; q<0x40; q++)
   {


    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;
    *(semetex+q+128*8)=0x3f;
    *(semetex+q+128*8)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;


   }
  wait(1000);

   *(semetex+0x801)=1;                   /*write pulse for contrast*/

  wait(1000);                    /*wait or lose pixels*/

  /*write rows above wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
    *(semetex+q+128*0)=0x3f;
  *(semetex+q+128*0)=0xcf;
  *(semetex+q+128*0)=0xf3;
  *(semetex+q+128*0)=0xfc;
  *(semetex+q+128*1)=0x3f;
  *(semetex+q+128*1)=0xcf;
  *(semetex+q+128*1)=0xf3;
  *(semetex+q+128*1)=0xfc;
  *(semetex+q+128*2)=0x3f;
  *(semetex+q+128*2)=0xcf;
  *(semetex+q+128*2)=0xf3;
  *(semetex+q+128*2)=0xfc;
  *(semetex+q+128*3)=0x3f;
  *(semetex+q+128*3)=0xcf;
  *(semetex+q+128*3)=0xf3;
  *(semetex+q+128*3)=0xfc;
  *(semetex+q+128*4)=0x3f;
  *(semetex+q+128*4)=0xcf;
  *(semetex+q+128*4)=0xf3;
  *(semetex+q+128*4)=0xfc;
  *(semetex+q+128*5)=0x3f;
  *(semetex+q+128*5)=0xcf;
  *(semetex+q+128*5)=0xf3;
  *(semetex+q+128*5)=0xfc;
  *(semetex+q+128*6)=0x3f;
  *(semetex+q+128*6)=0xcf;
  *(semetex+q+128*6)=0xf3;
  *(semetex+q+128*6)=0xfc;
 }
/*write rows below wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
  *(semetex+q+128*9)=0x3f;
  *(semetex+q+128*9)=0xcf;
  *(semetex+q+128*9)=0xf3;
  *(semetex+q+128*9)=0xfc;
  *(semetex+q+128*10)=0x3f;
  *(semetex+q+128*10)=0xcf;
  *(semetex+q+128*10)=0xf3;
  *(semetex+q+128*10)=0xfc;
  *(semetex+q+128*11)=0x3f;
  *(semetex+q+128*11)=0xcf;
  *(semetex+q+128*11)=0xf3;
  *(semetex+q+128*11)=0xfc;
  *(semetex+q+128*12)=0x3f;
```

```
     *(semetex+q+128*12)=0xcf;
     *(semetex+q+128*12)=0xf3;
     *(semetex+q+128*12)=0xfc;
     *(semetex+q+128*13)=0x3f;
     *(semetex+q+128*13)=0xcf;
     *(semetex+q+128*13)=0xf3;
     *(semetex+q+128*13)=0xfc;
     *(semetex+q+128*14)=0x3f;
     *(semetex+q+128*14)=0xcf;
     *(semetex+q+128*14)=0xf3;
     *(semetex+q+128*14)=0xfc;
     *(semetex+q+128*15)=0x3f;
     *(semetex+q+128*15)=0xcf;
     *(semetex+q+128*15)=0xf3;
     *(semetex+q+128*15)=0xfc;

   }
  /*write rows left of wavelet in 0 state*/
   for(q=0x0; q<0x38; q++)
   {
     *(semetex+q+128*7)=0x3f;
     *(semetex+q+128*7)=0xcf;
     *(semetex+q+128*7)=0xf3;
     *(semetex+q+128*7)=0xfc;
     *(semetex+q+128*8)=0x3f;
     *(semetex+q+128*8)=0xcf;
     *(semetex+q+128*8)=0xf3;
     *(semetex+q+128*8)=0xfc;
   }
  /*write rows right of wavelet in 0 state*/
   for(q=0x48; q<0x80; q++)
   {
     *(semetex+q+128*7)=0x3f;
     *(semetex+q+128*7)=0xcf;
     *(semetex+q+128*7)=0xf3;
     *(semetex+q+128*7)=0xfc;
     *(semetex+q+128*8)=0x3f;
     *(semetex+q+128*8)=0xcf;
     *(semetex+q+128*8)=0xf3;
     *(semetex+q+128*8)=0xfc;
   }

}

/** ** ** ** ** ** ** ** *** ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.
```

```
wait(t)
{
   time_t start, finish;
   time(&start);
   time(&finish);
   while(+difftime(finish, start) < t/1000)
   {
      time(&finish);
   }
}
/*******************************************************************/
```

```
/*****************************************************************************
 wavlt8.c

          AIR FORCE INSTITUTE OF TECHNOLOGY
       WAVELET PATTERN GENERATOR FOR THE SLM
                  by Capt Steve Pinski
                     July 24, 1991

 This program will display a Harr wavelet pattern on the SLM.
 The 2-dimentional wavelet is 8 x 8 pixels.
 *****************************************************************************/

#include "stdio.h"
#include "string.h"
#include "time.h"
unsigned r, q;

main()
{
  char far *semetex;
   semetex = (char far *) 0xb0000000;        /*SLM is at b000 segment*/
   *(semetex+0x800)=1;                       /*erase SLM*/
   wait(1000);                               /*wait or lose pixels*/

   /*write 8 rows for wavelet*/
   for(q=0x3c; q<0x40; q++)
   {
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
   }
  wait(1000);

   *(semetex+0x801)=1;              /*write pulse for contrast*/

  wait(1000);                  /*wait or lose pixels*/

  /*write rows above wavelet in 0 state*/
   for(q=0x0; q<0x80; q++)
   {
    *(semetex+q+128*0)=0x3f;
    *(semetex+q+128*0)=0xcf;
    *(semetex+q+128*0)=0xf3;
    *(semetex+q+128*0)=0xfc;
    *(semetex+q+128*1)=0x3f;
    *(semetex+q+128*1)=0xcf;
  *(semetex+q+128*1)=0xf3;
  *(semetex+q+128*1)=0xfc;
  *(semetex+q+128*2)=0x3f;
  *(semetex+q+128*2)=0xcf;
  *(semetex+q+128*2)=0xf3;
  *(semetex+q+128*2)=0xfc;
  *(semetex+q+128*3)=0x3f;
  *(semetex+q+128*3)=0xcf;
  *(semetex+q+128*3)=0xf3;
  *(semetex+q+128*3)=0xfc;
  *(semetex+q+128*4)=0x3f;
  *(semetex+q+128*4)=0xcf;
  *(semetex+q+128*4)=0xf3;
  *(semetex+q+128*4)=0xfc;
  *(semetex+q+128*5)=0x3f;
  *(semetex+q+128*5)=0xcf;
  *(semetex+q+128*5)=0xf3;
  *(semetex+q+128*5)=0xfc;
  *(semetex+q+128*6)=0x3f;
  *(semetex+q+128*6)=0xcf;
  *(semetex+q+128*6)=0xf3;
  *(semetex+q+128*6)=0xfc;
  *(semetex+q+128*7)=0xf3;
  *(semetex+q+128*7)=0xfc;
 }
/*write rows below wavelet in 0 state*/
 for(q=0x0; q<0x80; q++)
 {
  *(semetex+q+128*8)=0x3f;
  *(semetex+q+128*8)=0xcf;
  *(semetex+q+128*9)=0x3f;
  *(semetex+q+128*9)=0xcf;
  *(semetex+q+128*9)=0xf3;
  *(semetex+q+128*9)=0xfc;
  *(semetex+q+128*10)=0x3f;
  *(semetex+q+128*10)=0xcf;
  *(semetex+q+128*10)=0xf3;
  *(semetex+q+128*10)=0xfc;
  *(semetex+q+128*11)=0x3f;
  *(semetex+q+128*11)=0xcf;
  *(semetex+q+128*11)=0xf3;
  *(semetex+q+128*11)=0xfc;
  *(semetex+q+128*12)=0x3f;
  *(semetex+q+128*12)=0xcf;
  *(semetex+q+128*12)=0xf3;
  *(semetex+q+128*12)=0xfc;
  *(semetex+q+128*13)=0x3f;
```

```
    *(semetex+q+128*13)=0xcf;
    *(semetex+q+128*13)=0xf3;
    *(semetex+q+128*13)=0xfc;
    *(semetex+q+128*14)=0x3f;
    *(semetex+q+128*14)=0xcf;
    *(semetex+q+128*14)=0xf3;
    *(semetex+q+128*14)=0xfc;
    *(semetex+q+128*15)=0x3f;
    *(semetex+q+128*15)=0xcf;
    *(semetex+q+128*15)=0xf3;
    *(semetex+q+128*15)=0xfc;

  }
 /*write rows left of wavelet in 0 state*/
  for(q=0x0; q<0x3c; q++)
  {
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
  }
 /*write rows right of wavelet in 0 state*/
  for(q=0x44; q<0x80; q++)
  {
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0xfc;
  }

}

/** ** ** ** ** ** ** ** *** ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.

wait(t)
{
   time_t start, finish;
   time(&start);
   time(&finish);
   while(+difftime(finish, start) < t/1000)
   {
      time(&finish);
   }
}
/*****************************************************************************/
```

```
/*******************************************************************
 wavlt4.c

            AIR FORCE INSTITUTE OF TECHNOLOGY
          WAVELET PATTERN GENERATOR FOR THE SLM
                   by Capt Steve Pinski
                     July 26, 1991

 This program will display a Harr wavelet pattern on the SLM.
 The 2-dimentional wavelet is 4 x 4 pixels.
 *******************************************************************/

#include "stdio.h"
#include "string.h"
#include "time.h"
unsigned r, q;

main()
{
  char far *semetex;
   semetex = (char far *) 0xb0000000;          /*SLM is at b000 segment*/
   *(semetex+0x800)=1;                         /*erase SLM*/
   wait(1000);                                 /*wait or lose pixels*/

   /*write 4 rows for wavelet*/
   for(q=0x3e; q<0x40; q++)
   {


     *(semetex+q+128*7)=0x3f;
     *(semetex+q+128*8)=0xfc;


   }
  wait(1000);

   *(semetex+0x801)=1;                 /*write pulse for contrast*/

  wait(1000);                          /*wait or lose pixels*/

  /*write rows above wavelet in 0 state*/
   for(q=0x0; q<0x80; q++)
   {
      *(semetex+q+128*0)=0x3f;
     *(semetex+q+128*0)=0xcf;
     *(semetex+q+128*0)=0xf3;
     *(semetex+q+128*0)=0xfc;
```

```
    *(semetex+q+128*1)=0x3f;
    *(semetex+q+128*1)=0xcf;
    *(semetex+q+128*1)=0xf3;
    *(semetex+q+128*1)=0xfc;
    *(semetex+q+128*2)=0x3f;
    *(semetex+q+128*2)=0xcf;
    *(semetex+q+128*2)=0xf3;
    *(semetex+q+128*2)=0xfc;
    *(semetex+q+128*3)=0x3f;
    *(semetex+q+128*3)=0xcf;
    *(semetex+q+128*3)=0xf3;
    *(semetex+q+128*3)=0xfc;
    *(semetex+q+128*4)=0x3f;
    *(semetex+q+128*4)=0xcf;
    *(semetex+q+128*4)=0xf3;
    *(semetex+q+128*4)=0xfc;
    *(semetex+q+128*5)=0x3f;
    *(semetex+q+128*5)=0xcf;
    *(semetex+q+128*5)=0xf3;
    *(semetex+q+128*5)=0xfc;
    *(semetex+q+128*6)=0x3f;
    *(semetex+q+128*6)=0xcf;
    *(semetex+q+128*6)=0xf3;
    *(semetex+q+128*6)=0xfc;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;
    *(semetex+q+128*7)=0xcf;
  }
/*write rows below wavelet in 0 state*/
  for(q=0x0; q<0x80; q++)
  {
    *(semetex+q+128*8)=0xf3;
    *(semetex+q+128*8)=0x3f;
    *(semetex+q+128*8)=0xcf;
    *(semetex+q+128*9)=0x3f;
    *(semetex+q+128*9)=0xcf;
    *(semetex+q+128*9)=0xf3;
    *(semetex+q+128*9)=0xfc;
    *(semetex+q+128*10)=0x3f;
    *(semetex+q+128*10)=0xcf;
    *(semetex+q+128*10)=0xf3,
    *(semetex+q+128*10)=0xfc;
    *(semetex+q+128*11)=0x3f;
    *(semetex+q+128*11)=0xcf;
    *(semetex+q+128*11)=0xf3;
    *(semetex+q+128*11)=0xfc;
    *(semetex+q+128*12)=0x3f;
    *(semetex+q+128*12)=0xcf;
    *(semetex+q+128*12)=0xf3;
    *(semetex+q+128*12)=0xfc;
    *(semetex+q+128*13)=0x3f;
    *(semetex+q+128*13)=0xcf;
    *(semetex+q+128*13)=0xf3;
    *(semetex+q+128*13)=0xfc;
    *(semetex+q+128*14)=0x3f;
    *(semetex+q+128*14)=0xcf;
    *(semetex+q+128*14)=0xf3;
    *(semetex+q+128*14)=0xfc;
    *(semetex+q+128*15)=0x3f;
    *(semetex+q+128*15)=0xcf;
    *(semetex+q+128*15)=0xf3;
    *(semetex+q+128*15)=0xfc;

   }
  /*write rows left of wavelet in 0 state*/
   for(q=0x0; q<0x3e; q++)
   {
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*8)=0xfc;
   }
  /*write rows right of wavelet in 0 state*/
   for(q=0x42; q<0x80; q++)
   {
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*8)=0xfc;
   }

}

/** ** ** ** ** ** ** ** *** ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.

wait(t)
{
   time_t start, finish;
   time(&start);
   time(&finish);
   while(+difftime(finish, start) < t/1000)
   {
      time(&finish);
   }
}
/****************************************************************************/
```

```
/**********************************************************************
 128horiz.c

          AIR FORCE INSTITUTE OF TECHNOLOGY
        WAVELET PATTERN GENERATOR FOR THE SLM
                by Capt Steve Pinski
                  August 8, 1991

 This program will display a Harr wavelet pattern on the SLM.  The
 2-dimentional wavelet is 128 x 128 pixels (rotated horizontally).
**********************************************************************/

#include "stdio.h"
#include "string.h"
#include "time.h"
unsigned r, q;

main()
{
  char far *semetex;
   semetex = (char far *) 0xb0000000;          /*SLM is at b000 segment*/
   *(semetex+0x800)=1;                         /*erase SLM*/
   wait(1000);                                 /*wait or lose pixels*/

  /*write rows for +1 state*/
   for(q=0x0; q<0x80; q++)
   {
    *(semetex+q+128*0)=0x3f;
    *(semetex+q+128*0)=0xcf;
    *(semetex+q+128*0)=0xf3;
    *(semetex+q+128*0)=0xfc;
    *(semetex+q+128*1)=0x3f;
    *(semetex+q+128*1)=0xcf;
    *(semetex+q+128*1)=0xf3;
    *(semetex+q+128*1)=0xfc;
    *(semetex+q+128*2)=0x3f;
    *(semetex+q+128*2)=0xcf;
    *(semetex+q+128*2)=0xf3;
    *(semetex+q+128*2)=0xfc;
    *(semetex+q+128*3)=0x3f;
    *(semetex+q+128*3)=0xcf;
    *(semetex+q+128*3)=0xf3;
    *(semetex+q+128*3)=0xfc;
    *(semetex+q+128*4)=0x3f;
    *(semetex+q+128*4)=0xcf;
    *(semetex+q+128*4)=0xf3;
    *(semetex+q+128*4)=0xfc;
    *(semetex+q+128*5)=0x3f;
    *(semetex+q+128*5)=0xcf;
    *(semetex+q+128*5)=0xf3;
    *(semetex+q+128*5)=0xfc;
    *(semetex+q+128*6)=0x3f;
    *(semetex+q+128*6)=0xcf;
    *(semetex+q+128*6)=0xf3;
    *(semetex+q+128*6)=0xfc;
    *(semetex+q+128*7)=0x3f;
    *(semetex+q+128*7)=0xcf;
    *(semetex+q+128*7)=0xf3;
    *(semetex+q+128*7)=0xfc;

   }
     *(semetex+0x801)=1;              /*write pulse for saturation*/

}

/** ** ** ** ** ** ** ** * * * ** ** ** ** ** ** ** **
 TURBO-C has a time delay function "delay()" that could be used
 instead of this wait() subroutine.

wait(t)
{
   time_t start, finish;
   time(&start);
   time(&finish);
   while(+difftime(finish, start) < t/1000)
   {
     time(&finish);
   }
}
/*************************************************************************/
```

Bibliography

1. AT&T Electronic Photography and Imaging Center, Indianapolis IN. *AT&T Truevision Advanced Raster Graphics Adapter TARGA 8 Users's Guide* (Release 1.1 Edition), October 1985.

2. AT&T Electronic Photography and Imaging Center, Indianapolis IN. *Truevision TARGA Demonstration Disk*, 1986.

3. AT&T Electronic Photography and Imaging Center, Indianapolis IN. *Truevision TARGA Software Tools Notebook* (Release 4.0 Edition), August 1988.

4. Ayer, Captain Kevin W. *Gabor Transforms for Forward Looking InfraRed Image Segmentation.* MS thesis, AFIT/GE/ENG/89D-1, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1989 (AD-A215046).

5. Bhanu, Bir and Richard D. Holben. "Model-Based Segmentation of FLIR Images," *IEEE Transactions on Aerospace and Electronic Systems*, *26*(1):2–11 (January 1990).

6. Childress, Timothy G. and J. Thomas Walrond. *Position, Scale and Rotation Invariant Pattern Recognition for Target Extraction and Identification.* MS thesis, AFIT/GE/ENG/88D-4, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1988.

7. Cline, Captain John D. *Hybrid Optical/Digital Architecture for Distortion Invariant Pattern Recognition.* MS thesis, AFIT/GEO/ENG/89D-02, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1989 (AD-867524).

8. Cline, Captain John D. and others. "New Binarization Techniques for Joint Transform Correlation," *Optical Engineering*, *29*:1088–1092 (September 1990).

9. Corporation, Semetex. *SIGHT-MOD by Semetex.* Technical Report. Torrance CA.

10. Gaskill, J. D. *Linear Systems, Fourier Transforms, and Optics.* New York: John Wiley & Sons, 1978.

11. Goodman, J. W. *Introduction to Fourier Optics.* New York: McGraw-Hill Book Company, 1968.

12. Hecht, Eugene. *Optics.* Reading, Mass.: Addison-Wesley Publishing Company, Inc., 1988.

13. Kast, Brian A. and others. "Implementation of Ternary Phase Amplitude Filters using a Magneto-optic Spatial Light Modulator," *Applied Optics*, *28*(6):1044–1046 (March 1989).

14. Laing, Captain John S. *Analysis of Visual Illusions Using Multiresolution Wavelet Decomposition Based Models*. MS thesis, AFIT/GE/ENG/91D, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1991 (123).

15. Mallat, Stephane G. "Multifrequency Channel Decomposition of Images and Wavelet Models," *IEEE Transactions on Acoustics, Speech, and Signal Processing, 37*(12):2091–2110 (December 1989).

16. Mallat, Stephane G. "A Theory for Multifrequency Signal Decomposition," *IEEE Transactions on Pattern Analysis and Machine Intelligence, 11*:674–693 (July 1989).

17. Newport Corporation, Fountain Valley CA. *Operator's Manual HC-300 Holographic Recording Device*, undated.

18. Newport Corporation, Fountain Valley CA. *Operator's Manual HC-500 Holographic System Controller*, undated.

19. Psaltis, Demetri and others. "Optical Image Correlation with a Binary Spatial Light Modulator," *Optical Engineering, 23*(6):698–704 (November 1984).

20. Qui, B. and M. G. Hartley. "Real-World Image Segmentation Using Edge Detection and Subtraction," *Electronic Letters, 26*(6):353–355 (January 1990).

21. Rogers, Maj Steven K. "Lecture Notes taken in EENG 620, Pattern Recognition I." School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, January 1991.

22. Roggemann, Michael C. *Multiple Sensor Fusion for Detecting Targets in FLIR and Range Images*. PhD dissertation, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 1989 (AD-A207577).

23. Ross, William E. and others. "Two-dimentional Magneto-optic Spatial Light Modulator for Signal Processing," *SPIE Real Time Signal Processing V, 341*:191–198 (1982).

24. Ruck, Dennis W., et al. "Multisensor Target Detection and Classification." In *Proceedings of the SPIE Conference on Infrared Sensors and Sensor Fusion, Volume 931*, pages 14–21, Bellingham, WA: SPIE Press, 1988.

25. Semetex Corporation, Torrance CA. *SIGHT-MOD GGG Development System Operations Manual*, 13 December 1989.

26. Smiley, Captain Steven E. *Image Segmentation Using Affine Wavelets*. MS thesis, AFIT/GE/ENG/91D, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1991.

27. Tong, Carl, et al. "Multisensor Data Fusion of Laser and Forward Looking Infrared (FLIR) for Target Segmentation and Enhancement." In *Proceedings of the SPIE Conference on Infrared Sensors and Sensor Fusion, Volume 782*, pages 10–19, Bellingham, WA: SPIE Press, 1987.

28. Truevision Inc., Indianapolis IN. *TARGA Software Tools - Library Disk* (Version 4.0 Edition), 19 May 1988.

29. Truevision Inc., Indianapolis IN. *TARGA Software Tools - Utilities Disk* (Version 4.0 Edition), 19 May 1988.

30. Truevision Inc., Indianapolis IN. *Truevision Trutilities Disk* (Version 1.1 Edition), 22 May 1989.

31. Vander Lugt, A. B. "Signal Detection by Complex Spatial Filtering," *IEEE Transactions on Information Theory*, *IT-10*(2):139–145 (April 1964).

32. Veronin, Captain Christopher P. *Optical Image Segmentation Using Wavelet Filtering Techniques*. MS thesis, AFIT/GEO/ENG/90D-09, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1990.

33. Veronin, Captain Christopher P. and others. "An Optical Image Segmentor Using Neural Based Wavelet Techniques," *Optical Engineering (accepted for publication)* (February 1992).

34. Waas, Jaye, "Telephone Interview," 18 July 1991. Semetex Corporation, Torrance CA.