2

NPS EC-91-003

NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A232 553

DTIC
S ELECTE D
MAR 07 1991
B

AUTOMATIC PARTICLE SIZING FROM ROCKET MOTOR HOLOGRAMS

John P. Powers

December 1990

Approved for public release; distribution unlimited.

Prepared for:
Air Force Astronautics Laboratory
Edwards AFB, CA 93523-5000

91 3 04 003

NAVAL POSTGRADUATE SCHOOL
Monterey, CA

Rear Admiral R.W. West, Jr.
Superintendent

Dr. Harrison Shull
Provost

Reproduction of all or part of this report is authorized.

This report was prepared as part of a research project entitled "Particulate behavior in exhaust nozzles and plumes of solid propellant rocket motors," sponsored by the Air Force Astronautics Laboratory, Edwards AFB, California.

This report was prepared by:

[illegible]

John P. Powers
Professor
Department of Electrical and
Computer Engineering

Reviewed by:

Michael A. Morgan

Michael A. Morgan
Chairman, Department of Electrical
and Computer Engineering

Released by:

[illegible]

Paul J. Marto
Dean of Research

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE	Form Approved OMB No 0704-0188
1a. REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NPS EC-91-003	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School; 6b OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Air Force Astronautics Laboratory
6c. ADDRESS (City, State, and ZIP Code) Monterey CA 93943	7b ADDRESS (City, State, and ZIP Code) AL/LSNE, Dr. Michael Holmes Edwards AFB, CA 93523-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION Air Force Astronautics Lab.; 8b. OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MIPR #: F04611-90-X-0057
8c. ADDRESS (City, State, and ZIP Code) AL/LSNE Edwards AFB CA 93523-5000	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO; PROJECT NO; TASK NO; WORK UNIT ACCESSION NO

11. TITLE (Include Security Classification)
AUTOMATIC PARTICLE SIZING FROM ROCKET MOTOR HOLOGRAMS (U)

12. PERSONAL AUTHOR(S)
John P. Powers

13a. TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Technical	FROM Jan 86 TO Dec 90	1990, Dec 1	

16 SUPPLEMENTARY NOTATION

17 COSATI CODES		
FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Particle sizing, holograms, rocket motor, image processing, image digitization

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Microscopic particles within an optical hologram reconstruction have been successfully measured using an image digitizer and a PC/AT computer. The hologram was recorded during a test burn of some solid rocket fueland captured a 2"x2" volume of burning particles as they lift from the fuel surface during combustion. The computer processes the digitized images using feature identification algorithms and sizing in the feature's horizontal dimension, its vertical dimension and area. The operation of the algorithms have been validated against calibration objects. Statistical tests show that about 1,300 particles from several image frames are required to obtain a representative size distribution. Overlying speckle degrades the resolution of the image and can be reduced by a variety of techniques. The performance of these speckle-reduction techniques has been measured and compared in the areas of speckle reduction, loss of resolution, and processing time. Program sizes and processing times have been compared for both FORTRAN and C-language versions of the processing program.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL Professor John Powers	22b TELEPHONE (Include Area Code) (408) 646-2679/2081; 22c OFFICE SYMBOL

DD Form 1473, JUN 86 Previous editions are obsolete SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603

UNCLASSIFIED

AUTOMATIC PARTICLE SIZING FROM ROCKET MOTOR HOLOGRAMS

J.P. Powers
Department of Electrical
and Computer Engineering
Naval Postgraduate School
Monterey, California

3 December 1990

Abstract

Microscopic particles within an optical hologram reconstruction have been successfully measured using an image digitizer and a PC/AT computer. The hologram was recorded during a test burn of some solid-rocket fuel and captures a 2"x2"x2" volume of burning particles as they lift from the fuel surface during combustion. The computer processes the digitized image using feature identification algorithms and sizing in the feature's horizontal dimension, its vertical dimension and its area. The operation of the algorithms has been validated against calibration objects. Statistical tests show that about 1,300 particles from several image frames are required to obtain a representative size distribution. Overlying speckle degrades the resolution of the image and can be reduced by a variety of techniques. The performance of these speckle-reduction techniques has been measured and compared in the areas of speckle reduction, loss of resolution, and processing time. Program size and processing time have been compared for both FORTRAN and C-language versions of the processing program.

DTIC QUALITY INSPECTED 3

Accession For
NTIS GRA&I ☑
DTIC TAB ☐
Unannounced ☐
Justification

By
Distribution/
Availability Codes

Dist	Avail and/or Special
A-1	

Contents

List of Figures

List of Tables

Chapter 1

OVERVIEW

1.1 Introduction

Investigations have been made to study the effects of addition of aluminum and other metallic particles on the magnitude of the performance losses in propellant motors. Performance is sensitive to the particle size distribution throughout the rocket motor and nozzle. Since no size distribution data exists in these regions of the motor for evaluation of existing analytical models, direct observation has been undertaken to provide the needed information.

Holographic techniques have been employed to capture the dynamics of the combustion chambers of small rocket motors while firing. These techniques have been refined and upgraded. Concurrently, improvements in the processing of the holograms to extract the particle size distributions are also necessary. Once the hologram has been successfully recorded, it is desirable to have a computer process the image to measure the particle size and to produce a statistical description of the particle size distribution. It is the purpose of this report to summarize the previous work in the development of these computer-processing techniques.

The steps required to produce the statistical distribution follow (more complete descriptions of each step are included later in this report):

1. image acquisition form the hologram reconstruction;
2. image digitization and storage on the computer system;
3. speckle reduction filtering to separate the particle image information from the overlaying speckle;

4. application of an image threshold to separate the particle features from the background features;

5. feature identification to find connected feature pixels and to "recognize" the connected pixels as a single object;

6. feature sizing to measure the number of features, the area, x–chord width, and y–chord width; and

7. histogram production using the size data of the prior step.

In this report the sizing problem is treated first and the speckle reduction techniques are discussed in a separate chapter.

1.2 Hologram reconstruction

As mentioned in the introduction, holographic techniques are used to capture the image of a rocket combustion chamber during firing. The pulsed ruby laser utilized in the recording process uses a glass diffuser in its illumination path. This diffuser is necessary to cut down the presence of Schlieren interference fringes produced by the thermal and density gradients surrounding the burning particles in the rocket motor [1].

The reconstru_tion process is shown in Fig 1.1. The hologram is reconstructed using a krypton laser and viewed through the microscope using 2x, 4x or 10x magnification. The diffuse light from the recording interferes with the reference wave of the krypton reconstruction laser. This random interference causes speckle to be introduced into the image. By attaching a 0.5 lux low–light–level camera and a video cassette recorder to the microscope, the speckle–corrupted image is recorded and preserved for later use. More detailed descriptions of the reconstruction process are in Refs. [1] and [2].

1.3 Image digitization and processing

Once on VCR tape, the desired image is played back and digitized by the PC/Vision image processing board controlled by either ImageAction or ITEX/PC software. The digitized image then may be filtered immediately or stored on floppy disk for later use.

An IBM PC/AT is the heart of the entire process of image manipulation, from digitizing an image to speckle reduction to the counting of the features. Installed on the IBM PC/AT are a PC/Vision frame–grabber board,

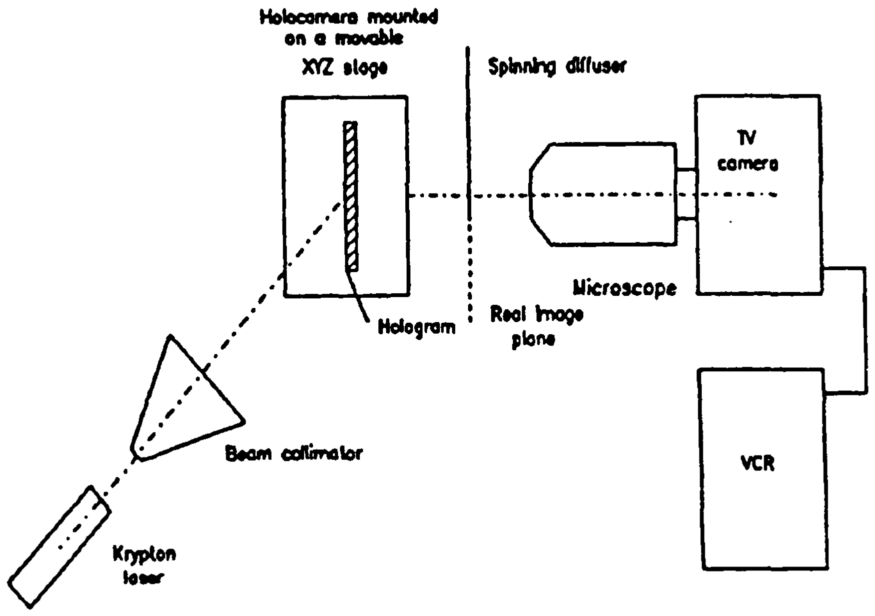


Figure 1.1: Hologram reconstruction setup.

ImageAction and ITEX/PC software (all by Imaging Technology, Inc.), the computer monitor, a video monitor and a video cassette recorder (VCR).

A video image from the VCR can be "grabbed" by the frame–grabber under control of the special software. Each of the picture elements (pixels) in the 512x480 array comprising a video image is assigned an integer gray level from 0 to 255 by the frame–grabber. Level 0 on the gray scale is blackest–black. Level 255 is whitest–white, while values in between are various shades of gray. Each pixel is uniquely addressable and its gray level alterable, thereby allowing the capacity to achieve digital filtering.

The ImageAction software is a set of menu–driven routines for use with a mouse. A totally–closed system, it can perform image graphics, image analysis, image processing and filtering. Particularly useful are the "grab" routine described in the preceding paragraph, and the "histogram" routine which outputs the image statistics (standard deviation, mean, variance) along with the image's histogram. More details are described in Reference [3].

The ITEX/PC software performs most of the same functions as the ImageAction software [3]. An important advantage is that the ITEX/PC Pascal subroutines can be called from main programs written in Microsoft versions of Pascal, C or FORTRAN. This allows the user a great deal of flexibility in programming.

The ITEX/PC package consists of subroutines that may be called from a main program to perform a specific function or to return a certain value. For example, if the subroutine

```
CALL THRESH (lowcut,highcut)
```

is used to threshold an image, all the pixel values between the integers "lowcut" and "highcut" will be displayed as a new value of 255 (white). All other pixels will be displayed as 0 (black). If, as in our work, "highcut" is set to 255 and "lowcut" varied by the user during run–time, a binary image results which presents black particles on a white background.

Sometimes an ITEX/PC routine is treated as a function. An example is

```
L = RPIXEL (x,y).
```

Here, the gray level of the pixel located at (x, y) is read and the value is assigned to the integer L. The value of L may then be varied by one of the filtering algorithms and the new value written back into the pixel at (x, y) using

CALL WPIXEL (x,y,L).

Some specifics of ITEX/PC bear mentioning. A standard television screen is made up of a 512x512 array of pixels. Of these, only 512x480 are actually visible. The lowest 32 rows are hidden out of view and may contain other information.

On the monochrome PC/Vision frame–grabber board, four look–up tables (LUTs) may be utilized. A LUT serves to transform pixel values before they are displayed. The standard LUT is linear, meaning that the output values equal the input values. Certain actions, such as thresholding, change the LUT values. The actual values of the pixels in the frame–grabber board memory remain unchanged by the threshold, but are simply altered by the LUT in such a way that a binary image appears on the screen. If it is desired to make the threshold permanent and actually change the frame–grabber memory to their new values, the MAPLUT subroutine must be called. If not, a call to INITIA to initialize the LUT's returns the screen to its "before–threshold" likeness.

The PC/Vision board was successfully installed in a PC/AT computer purchased to act as host. The ImageAction software proved to be extremely simple but powerful in manipulating the images. Late in the FY86, the ITEX/PC subroutine package was purchased and installed. It was successfully used in the study of speckle reduction techniques reported below.

In FY89 the image processing system was upgraded to a PC/Vision*plus* image processing systems, including ImageAction*plus* and ITEX/PC*plus* software. The board had more memory (to store two images), more LUTs, and a reorganized memory scheme to allow the board to be compatible with the software and boards offered by other vendors. While the subroutine calls in the ITEX/PC*plus* software had a different syntax, the functions performed were much the same as the prior versions of ITEX/PC.

1.4 Thresholding

A *histogram* of an image consists of a bar graph. A separate bar exists at each gray level, with the height of the bar proportional to the number of pixels at that gray level. (See the inset of Figure 1.2 for an example.) Thus, a histogram gives a visual representation of the general darkness or lightness of the image and how widely separated in gray level certain features may be. It can suggest where the best level to place a threshold might be.

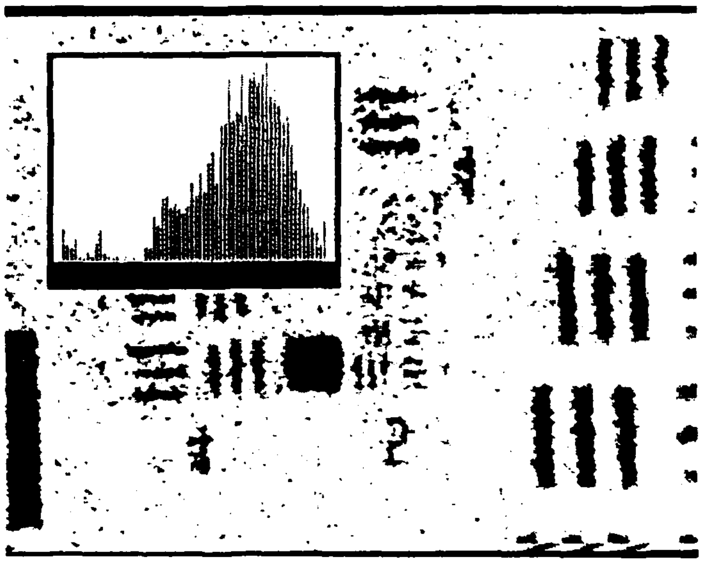


Figure 1.2: Histogram of representative image (inset).

Ideally, the feature data in an image would be very different from the background in gray level, as in Figure 1.3. A threshold could then be placed at a level midway between them with the result that the feature particles would be black and the background white. The programs described in this report could then be used to properly size and count the particles of interest.

In practice, Figure 1.4 is much more likely. The particles cover much of the range of gray levels, as does the background speckle. A threshold placed in an unfiltered image would cause portions of some particles to disappear while part of the speckle would appear as particles. (The overlap is also apparent in Figure 1.2.)

What is required is a method to process out the unwanted speckle noise in the background. With a clear separation between the particles of interest and the background, a threshold could then be applied with success. Figure 1.5 illustrates the result of a filtering operation to enhance the particle against the background. While complete separation of the particles from the background has not occurred, the hump at the center-left of the histogram indicates some improvement in the isolation of the particles from the background.

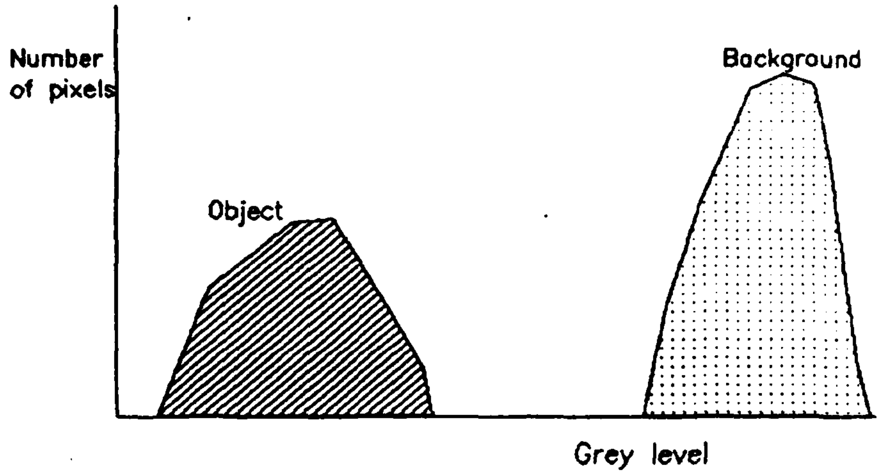


Figure 1.3: Ideal separation of dark object pixels from bright background pixels.

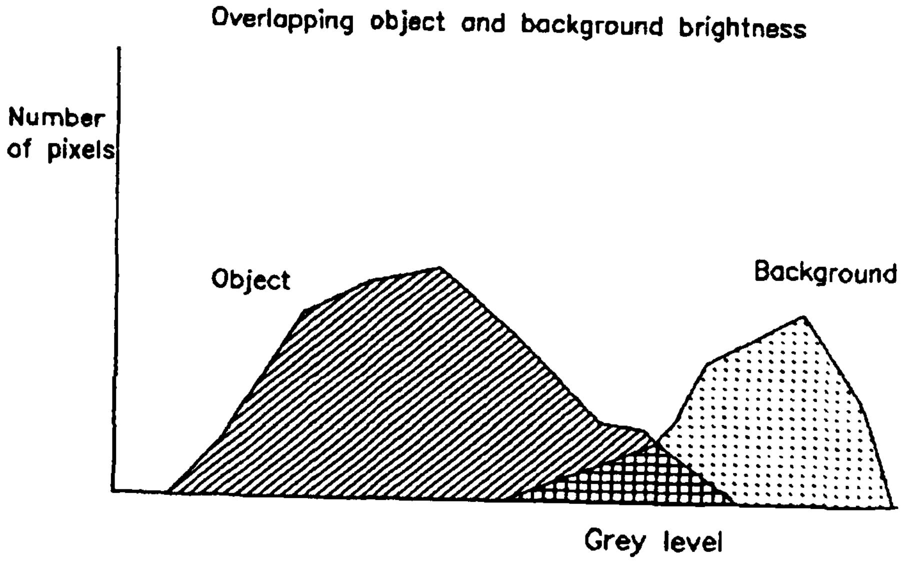


Figure 1.4: Histogram of representative image showing an overlap in the gray-scales of the object pixels and the background pixels.

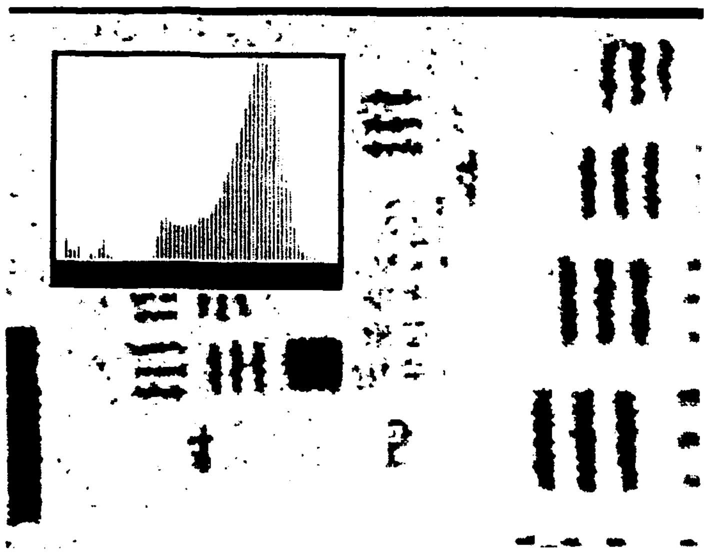


Figure 1.5: Histogram of image after filtering to enhance particles from background. Note hump in the center–left of histogram.

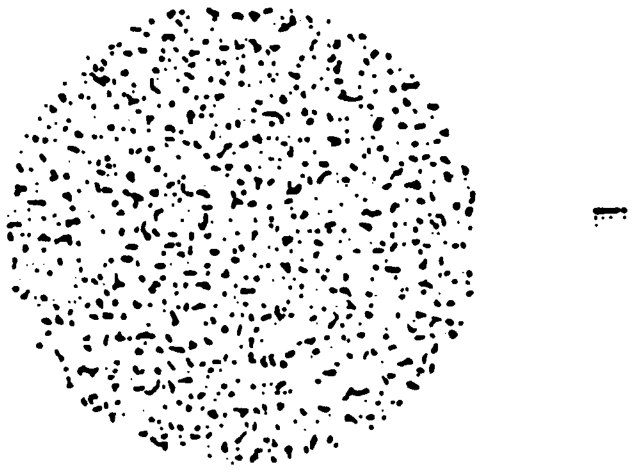


Figure 1.6: LEOS Inc. calibration reticle.

1.5 Test objects

Three test objects have been used in our investigations to provide a resolution calibration. These objects were imaged under white light as the highest contrast images with the most benign background. The objects wer϶ also recorded in holograms in the test setup to duplicate the test geometry.

The first object is a calibration reticle produced by LEOS, Inc., shown in Figure 1.6 The circular portion of the reticle consists of approximately 10,000 opaque circular features of twenty-three sizes ranging from five to ninety-three micrometers in diameter. These circles are photodeposited in an eight millimeter circular area. A rectangular array to the right of the circular test pattern consists the twenty-three standard sizes arranged in rows of five as seen in Figure 1.7.

The second object is the 1951 USAF Standard Resolution chart (see Figure 1.8). This object was used for resolution studies as it provides a more continuous measurement of resolution degradation than the LEOS calibration array.

The final object was the reconstruction from rocket motor holograms recorded during firing using propellant samples supplied by the Air Force.

Recently, we have begun a series of measurements [12] of scanning elec-

9	7	5		
27	24	21	17	12
43	40	37	34	31
61	56	52	50	47
93	87	81	74	67

Figure 1.7: Size of particles (in μm) in rectangular array portion of calibration reticle.

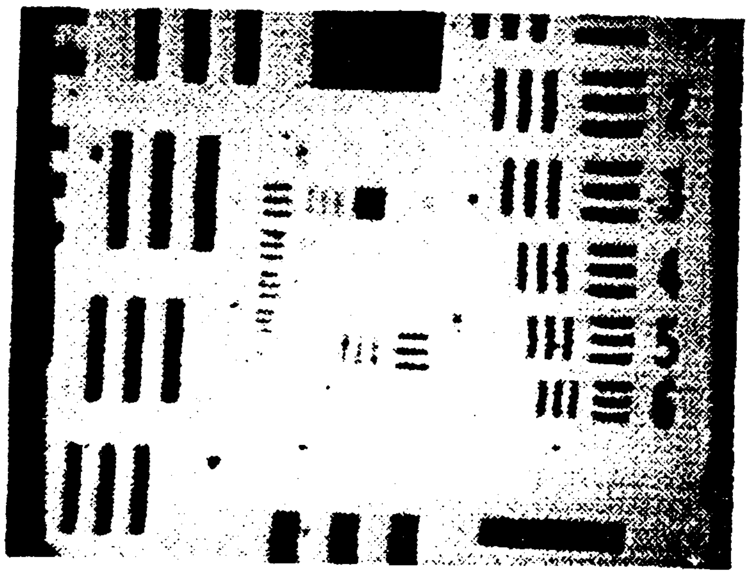


Figure 1.8: Image of the Air Force standard resolution chart.

tron microscope (SEM) images of exhaust particles caught on filter paper. The particle sizes range from 0.25 μm to 2.0 μm.

1.6 Thesis efforts

The work reported in this report has enjoyed the support of many thesis students working on degrees at NPS.

- CPT Larry Klooster [4], USA, began the investigation by installing and testing a Quantimet 720 image processing system.

- LT Phil Shook [5], USCG, followed Klooster and was able to improve on the operation of the Quantimet system and to obtain some preliminary results. The Quantimet instrument and its PDP-11 interface was difficult to operate and to keep running, however. The invention of image-processing boards for the PC/AT computer came at this time offering significant ease of operation without any lessening of image-processing power, so we decided to abandon the Quantimet effort and to concentrate on the PC system study.

- MAJ David Redman [6], Canadian Forces, installed and used the PC/Vision board, together with the ImageAction software, to digitize and process images. Redman developed the algorithm used for sizing the particles (based on Ref. [13]) and obtained preliminary measurement results for the calibration objects and holograms. Due to memory restrictions Redman was able to process only one-fourth of an image frame at a time. Later efforts removed that limitation. His work is discussed in more detail in the chapters on Measurement Results and on Code Optimization.

- CPT Tom Edwards [7], USMC, worked on the problem of reducing the speckle. He implemented three non-linear filters used in the synthetic aperture radar community and compared the performance of the speckle reduction filters against each other, using the "speckle index" concept to quantify the amount of the speckle. More information can be found in the chapter on Speckle Reduction.

- LT(jg) Sami Orguc [8], Turkish Navy, worked to optimize the FORTRAN code produced by Redman and Edwards. He was able to make use of calls to the video memory of the image processing board, rather than using arrays in the main computer memory, to process the full image frame rather the than the quarter-frame images that were previously processed. This work is discussed in more detail in the chapters on Measurement Results and on Code Optimization. He also investigated the use of convolution filters to reduce the speckle and measured data from experimentally-recorded holograms.

- MAJ Denis Carrier [9], Canadian Forces, continued the investigation into the speckle reduction filters. He measured the speckle reduction produced by the rotating mylar diffuser screen used in the reconstruction process and found it to be superior to the nonlinear filters. However, the nonlinear filters proved useful in reducing the already-reduced speckle that remained in the images produced with the rotating mylar screen. He also investigated using the computer to reduce the speckle by averaging several frames from the non-rotating screen. His work is discussed in more detail in the Speckle Reduction chapter. A second problem that Carrier solved was the question of adequate sample size to have confidence in the statistics. As one enlarges a data sample, at some point the statistics begin changing by only small amounts. Further increases in the sample size no longer produce a significant variation in the sample size. Carrier was able to quantify this effect and show that, for our holograms, a sample size of 1,200 particles was sufficient to produce a stable size distribution. A detailed discussion is found later in the section on sample size on page 21.

- LCDR Dana Kaeser [10], USN, modified the FORTRAN programs for use with an optimizing FORTRAN compiler (Microsoft FORTRAN, Ver. 4.0). He also ported the code for all operation into the C language for greater efficiency and operating speed. He introduced the concept of virtual arrays so that large data arrays could be processed without interference from the 640K DOS memory limit. (The fact that the ITEX/PC library had to be used with a Microsoft compiler limited operation of the image processing board to a DOS environment. Other packages are available that allow operation of the board under UNIX.) His work is discussed in the chapter on Code Optimization.

- LT Valerie Hockgraver [11], USN, optimized Kaeser's code for smaller code size and faster running times and integrated the modular programs into one master program that allows centralized use. More information is found in the chapter on Code Optimization.

- MAJ Yeoh–Lip Lee [12], Singapore Defense Forces, is currently investigating the use of the image processing programs to measure particle sizes in images produced by a scanning electron microscope (SEM). He is currently in the beginning stages of this work.

1.7 Organization of report

Following this introductory chapter, Chapter 2 describes the techniques used to identify and size the features in an image and presents some experimental results. It also presents results on the resolution of the images. Chapter 3 discusses techniques for reducing the overlying speckle in our images using various techniques. It quantitatively compares the techniques and presents the results in terms of the reduction in speckle, the loss in resolutio: and the processing time. Chapter 4 describes the various iterations of the code that developed for processing the hologram reconstructions. It traces the evolution of the programs from their original FORTRAN implementation to their present C–language versions. Chapter 5 presents some concluding comments and a summary.

Chapter 2

PARTICLE SIZING

2.1 Sizing process

The initial research effort by Redman [6] on the PC/AT image processing system focused on image capture, image digitization, and particle sizing. The ImageAction software was used under operator control to process the image to remove the speckle and background effects. FORTRAN programs were then used to process and size the images. Due to computation time and to computer memory size constraints, initial efforts worked with only one-quarter of the screen image (256x256 pixels). (Later, improved programming and subroutines removed those constraints.)

Redman processed the digitized images as follows:

- Image averaging to reduce speckle. Two techniques were explored; the first used a spinning mylar screen as the focus of the image. The motion of the mylar screen during the 1/30th of a second scan time causes the speckle to blur and to reduce the speckle contrast compared to the dark particles. Alternatively, digital averaging of images taken without the mylar disk being present was used to reduce the speckle contrast. It was found that after approximately fifteen images were added, no further reduction of speckle was observed. Additionally the speckle reduction was found to be no better than obtained with the rotating mylar disk. Since the rotating disk was experimentally easier to implement, it was chosen for use with all subsequent recorded images. (These results were later quantified by Carrier [9] and are discussed further in the following chapter on speckle reduction.)

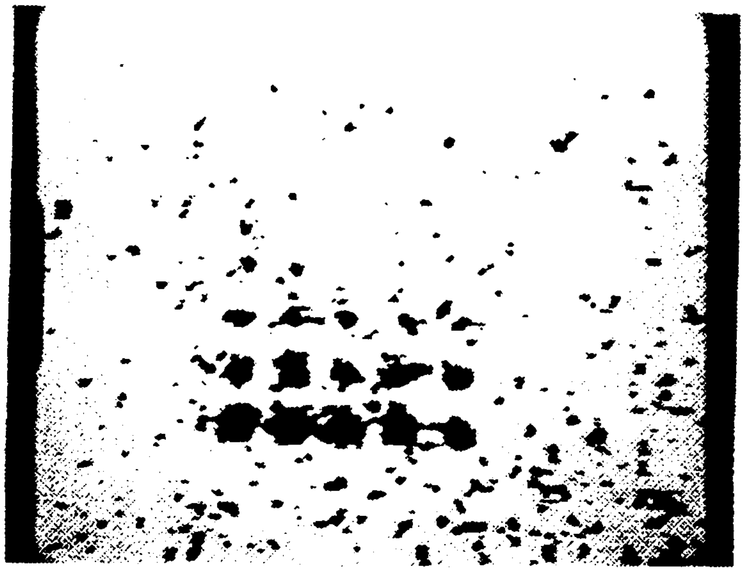


Figure 2.1: Image after threshold without any speckle reduction.

- Redman also further reduced the speckle with a processing algorithm that was empirically designed. Using built–in routines of the Image-Action software, a blur filter was applied, followed by a lowpass filter. The two–step process was then repeated. When correct filtering was done, the speckle was significantly reduced. It was noted however that resolution was also reduced and that close particles would tend to merge into one unresolved clump after processing. No attempt was made to quantize the loss of resolution at this point.

- The next processing step is the application of a threshold to remove all information from the background. Figures 2.1 and 2.2 shows a comparison illustrating the filtering effect. Figure 2.1 is a thresholded image after the image was averaged but not filtered before applying the threshold. Significant error due to the speckle effects is present. Figure 2.2 shows the same image but *with* the speckle reduction filter applied before applying the threshold. Far less noise is present. (The calibration array object also suffered some physical damage before the

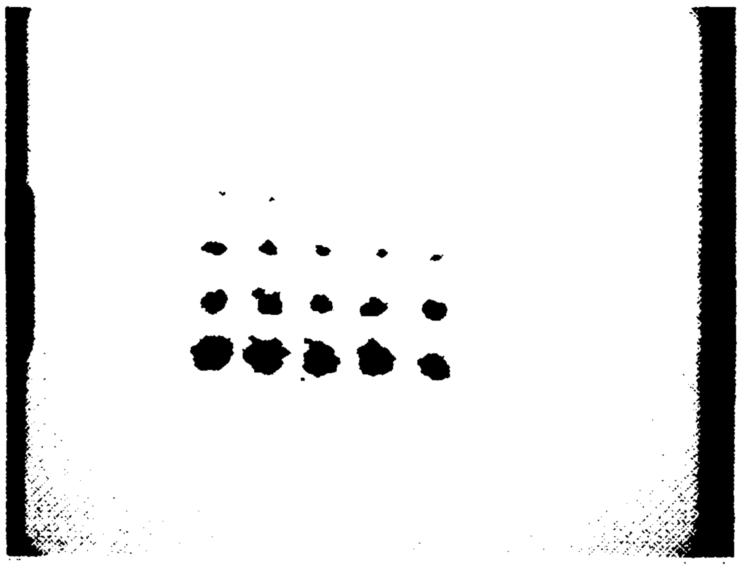


Figure 2.2: Image after threshold *with* speckle reduction.

image was made, causing the loss of the center circle in the top row of Figure 2.2 as well as some damage to some of the larger circles.)

- The thresholded image was then processed by the FORTRAN routine for object identification, counting and sizing. The object identification was done by scanning the data array for adjacent pixels in the horizontal and vertical directions only, with no check made on the diagonals in Redman's algorithm. (Orguc [8] added the diagonal checks in a later version of the algorithm.) Adjacent dark pixels are joined to form one object or "feature". The area and maximum chord widths are then computed for each object and written into a data table. (If desired, a roundness test can applied to the chord length measurements to eliminate nonspherical particles.) This data table was processed by a data processing program written by Redman to produce the final histogram data on particle size. (Later efforts used commercial statistical packages to handle the data analysis and histogram formation.)

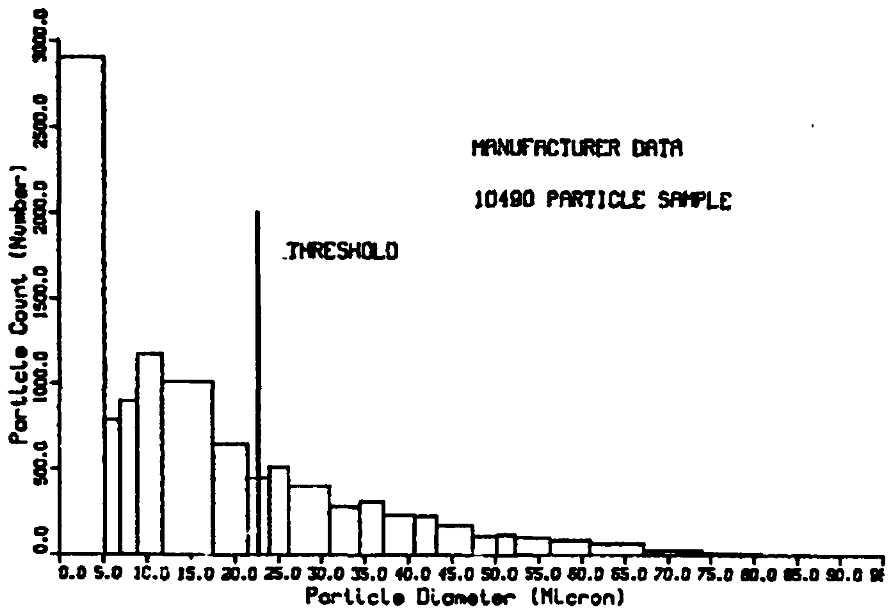


Figure 2.3: Predicted histogram of LEOS calibration object using manufacturer's size data.

2.2 Measurement results

Figure 2.3 [6] shows the histogram of particle size on the LEOS calibration array (Figure 1.6 on page 9) as provided by the manufacturer. Imaging tests showed that the system is unable to resolve particles smaller than the size marked as "threshold" on Figure 2.3. This resolution threshold is approximate due to the coarse quantization of the particle sizes in the sample.

Using the hologram reconstruction system, the smallest particle resolved was on the order of 20 μm in Redman's work. Adjusting the manufacturer's data for this resolution threshold produced the histogram of Figure 2.4.

The measured histogram from a hologram reconstruction of the calibration array with 228 particles in the field of view is shown in Figure 2.5. Again the lower size bin is not empty due to quantization effects and noise, but the general shape of the curve is in agreement with the size data supplied by the manufacturer.

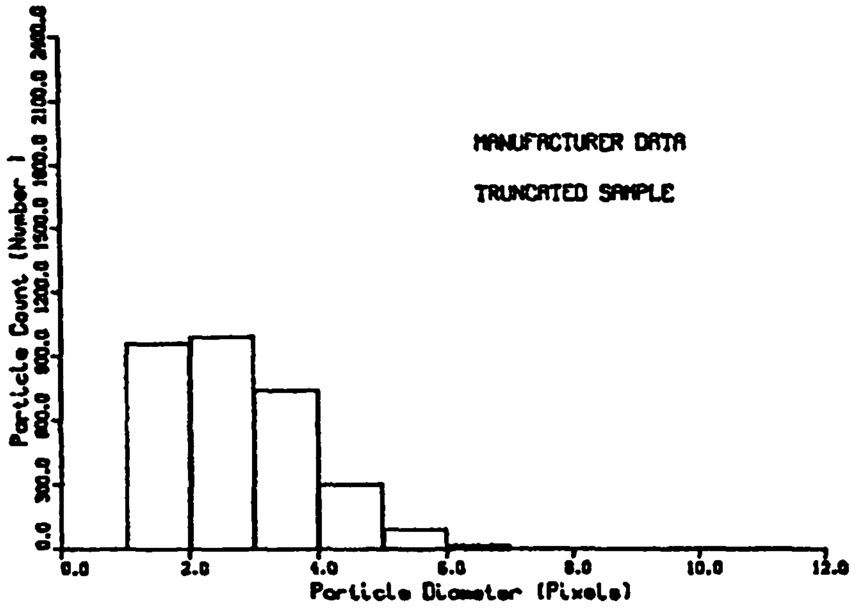


Figure 2.4: Adjusted histogram from manufacturer's data. Unresolved particles less than 20 μm in diameter have been removed from the data.

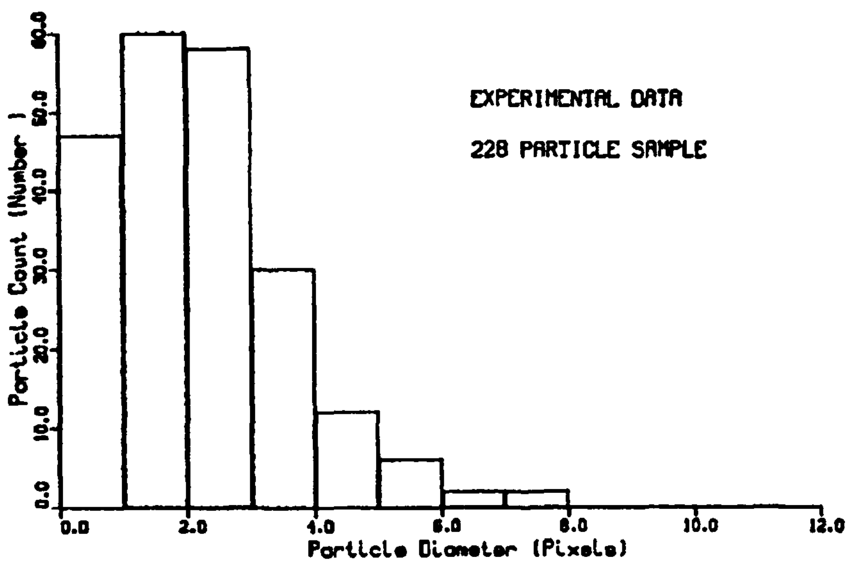


Figure 2.5: Histogram of measured size for 228 particles in the LEOS calibration object.

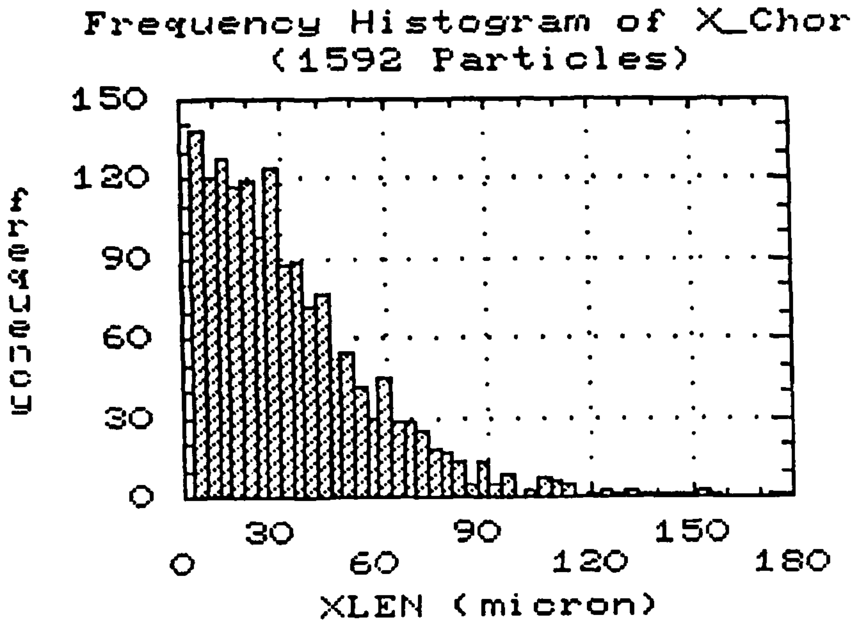


Figure 2.6: Histogram of particle sizes (in μm) for 1,592 particles contained in 16 fields of view.

The 20 μm minimum resolution was disappointing; we had expected to be able to resolve smaller particles based on a preliminary calculation [15]. From these preliminary resolution results, we decided to further explore the loss of resolution in the processing steps further by using the Air Force Resolution chart as an object, since it has more steps in its quantization of resolution. These steps provide a finer gradation in the ability to measure the resolution than the LEOS calibration array. Additionally, other techniques for the reduction of speckle were required for small objects since we suspected that the smaller objects might have been removed by Redman's blur filter steps. (The work on speckle reduction is reported in a separate chapter that follows.)

Later measurement results by Orguc [8] expanded the number of particles measured. Figure 2.6 shows a frequency histogram of particle size distribution as measured from an experimental hologram taken during a rocket firing. The horizontal axis is the x-width of the particle. The data represents 1,592 particles that were obtained by combining the data from 16 different regions in the hologram reconstruction. Other results with fewer particles are shown in the section on sample size that follows in this chapter.

2.3 Sample size

The field of view of the microscope system is determined by the magnification required to resolve the smallest particle. The number of particles within the field of view depends on the size of the field of view. For the small fields of view required for small particles, the number of views to completely cover a hologram can approach the millions. One question of interest, then, was how many particles were enough to provide a meaningful set of statistics. Figure 2.7 represents the data from 96 particles in one field of view; Figure 2.8 represents the data from 681 particles in 7 fields of view. These histograms are incomplete and show major changes in shape as more particles are added to the data sample. Figure 2.9 is the data from 984 particles in ten fields of view; it has the general shape of the distributions from larger numbers of particles for this hologram. For this hologram we experimentally concluded that approximately 1,000 particles need to be measured before the distribution stays about the same. Carrier [9] found that the proper statistical test to ensure an adequate sample size to have confidence in the probability distribution is the Kolmogorov–Smirnov two–sample test [16]. This test determines if two data samples are from the same distribution. (The Kolmogorov–Smirnov one–sample test decides how closely a single sample of data fits a hypothesized distribution.) We are given two samples; sample A has been made with n values and sample B has been made with m values (where $n > m$). We want to test the hypothesis: sample A has the same probability distribution (or sample–size frequency distribution) as sample B (or, conversely, we want to test whether the data samples are from different distributions). It is assumed that the samples are independent of each other (i.e., they are taken from independent fields of view with no overlap). The test requires that two cumulative distributions, $S_1(x)$ and $S_2(x)$, be made up from the samples (see Figure 2.10). The Kolmogorov statistic T_1 is the *largest* vertical distance between the distributions, as also shown in the figure. Then

$$T_1 = \text{MAXOF}\{\,|S_1(x) - S_2(x)|\,\}\,, \qquad (2.1)$$

wheie the MAXOF$\{\cdot\}$ operator takes the maximum value of its argument.

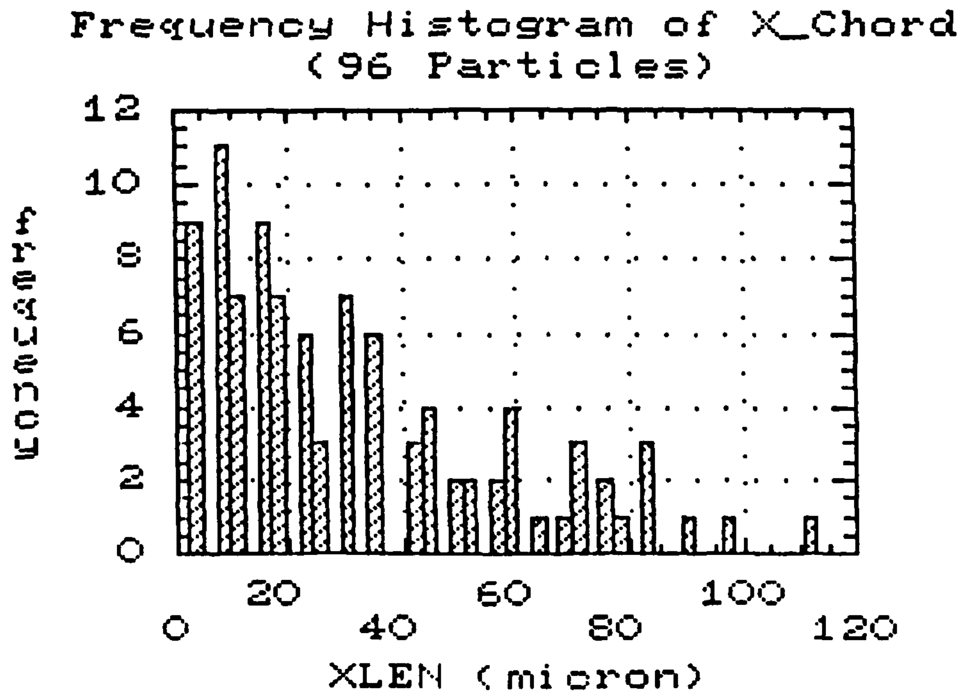


Figure 2.7: Histogram of particle sizes (in μm) for 96 particles contained in 1 field of view.

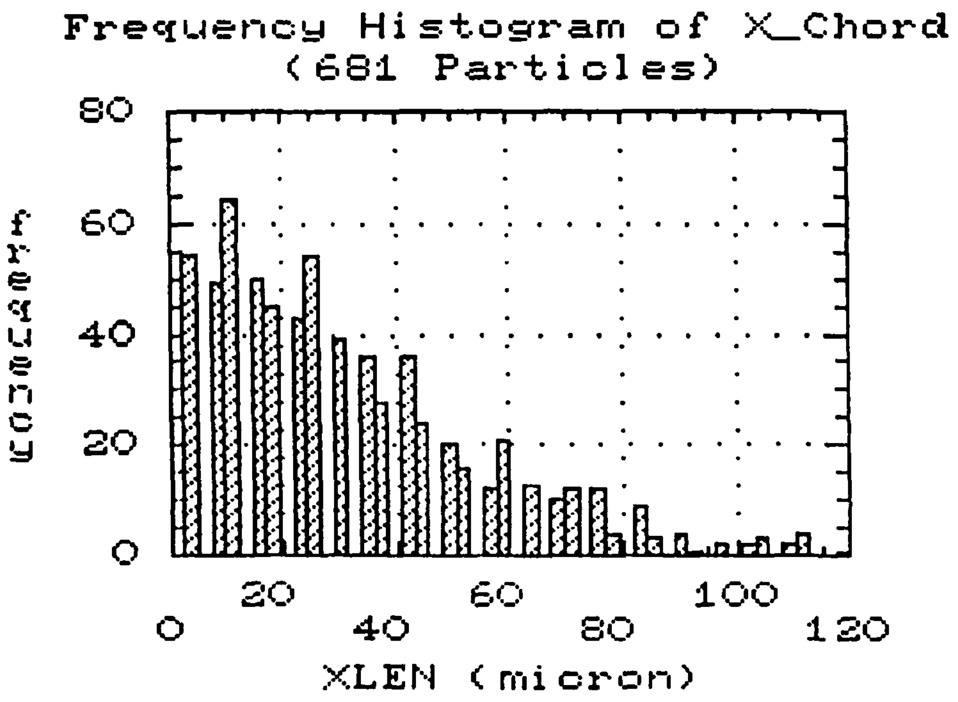


Figure 2.8: Histogram of particle sizes (in μm) for 681 particles contained in 7 fields of view.

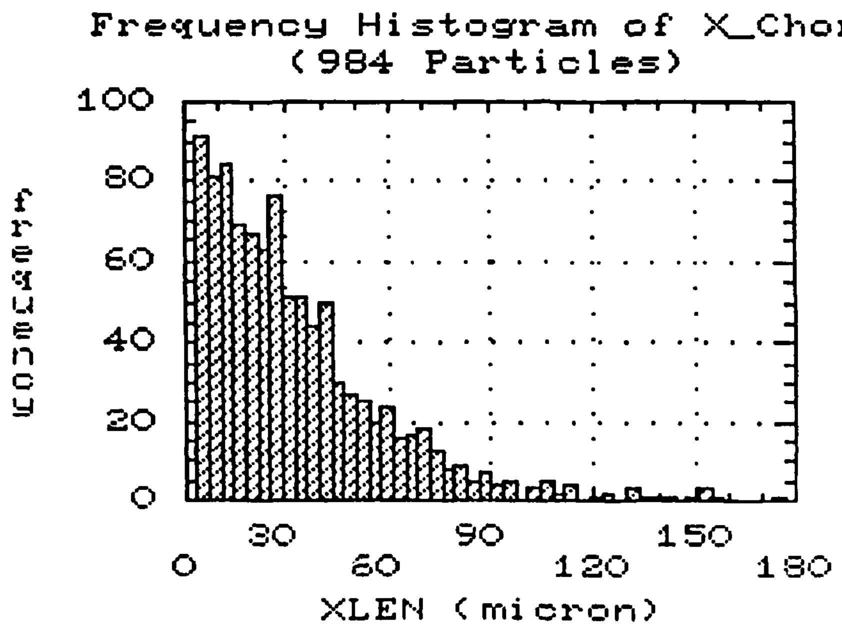


Figure 2.9: Histogram of particle sizes (in μm) for 984 particles contained in 10 fields of view.

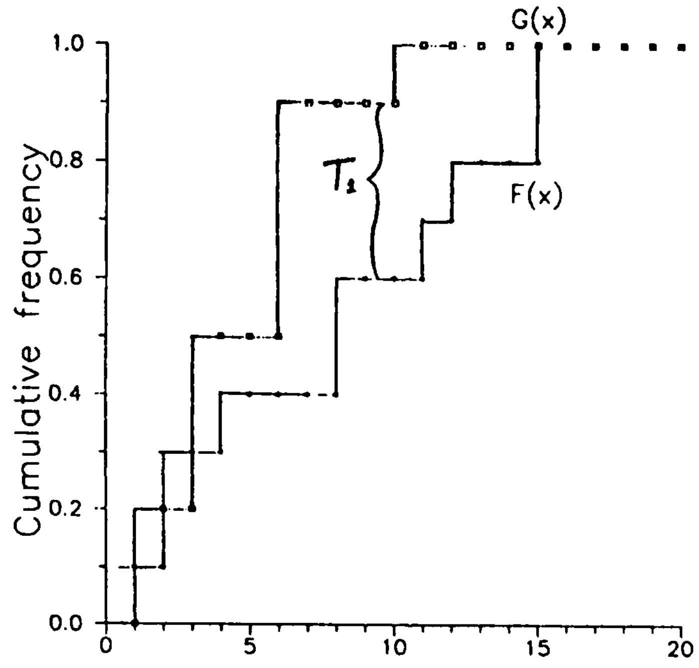


Figure 2.10: Two independent cumulative frequency distributions and the Kolmogorov statistic T_1.

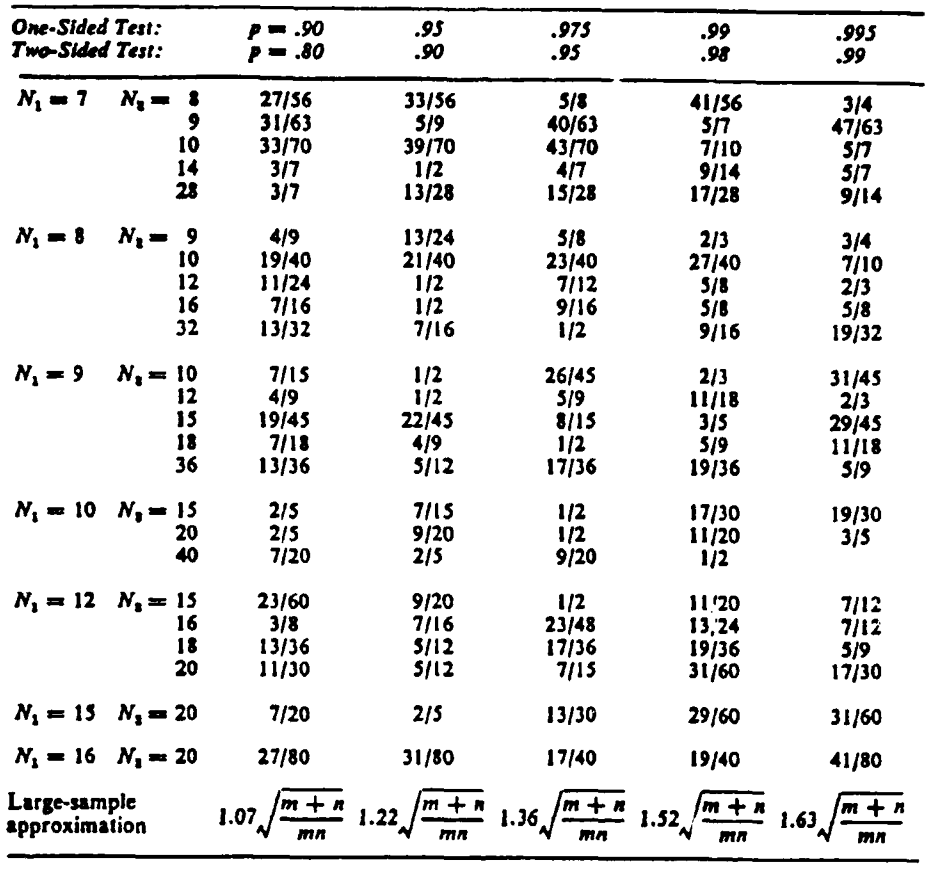
X_i	Y_i	$\|S_1(x) - S_2(x)\|$	X_i	Y_i	$\|S_1(x) - S_2(x)\|$
	5.2	$\|0 - 1/15\| = 3/45$		9.8	$\|5/9 - 8/15\| = 1/45$
	5.7	$\|0 - 2/15\| = 6/45$	9.9		$\|6/9 - 8/15\| = 6/45$
	5.9	$\|0 - 3/15\| = 9/45$	10.1		$\|7/9 - 8/15\| = 11/45$
	6.5	$\|0 - 4/15\| = 12/45$	10.6		$\|8/9 - 8/15\| = 16/45$
	6.8	$\|0 - 5/15\| = 15/45$		10.8	$\|8/9 - 9/15\| = 13/45$
7.6		$\|1/9 - 5/15\| = 10/45$	11.2		$\|9/9 - 9/15\| = 18/45$
	8.2	$\|1/9 - 6/15\| = 13/45$		11.3	$\|9/9 - 10/15\| = 15/45$
8.4		$\|2/9 - 6/15\| = 8/45$		11.5	$\|9/9 - 11/15\| = 12/45$
8.6		$\|3/9 - 6/15\| = 3/45$		12.3	$\|9/9 - 12/15\| = 9/45$
8.7		$\|4/9 - 6/15\| = 2/45$		12.5	$\|9/9 - 13/15\| = 6/45$
	9.1	$\|4/9 - 7/15\| = 1/45$		13.4	$\|9/9 - 14/15\| = 3/45$
9.3		$\|5/9 - 7/15\| = 4/45$		14.f	$\|9/9 - 15/15\| = 0/45$

Table 2.1: Two–sample distributions. (After W.J. Conover, *Practical Non-parametric Statistics*, John Wiley and Sons, 1971.)

We decide whether to reject or accept the hypothesis with a by comparing T_1 with a tabular value found in various statistical texts (see Figure 2.11). The table shows the quantile p for the one–sided and two–sided test at the top. The hypothesis is accepted with a significance of $1 - p$ if the value of T_1 is less than the table entry for the values of m and n of the samples. The *significance value* of a conclusion has the value of $1 - p$ if the hypothesis is accepted as a result of the test.

To illustrate the use of the table, we consider a simple example from Ref. [16] where we have two independent samples, one of nine particles ($X_i, i = 1 \ldots 9$) and one of fifteen particles ($Y_i, i = 1 \ldots 15$). The samples are given in Table 2.1. Scanning the table we observe that the largest difference is 18/45 (at the 11.2 value of the samples), so $T_1 = 18/45 = 0.40$. For this set of data we enter the Kolmogorov table at $N_1 = 9$ and $N_2 = 15$. The top heading gives the confidence level desired (we want to use the "Two-Sided Test" values).

We can accept the hypothesis with a significance of 5% (= 1 − 0.95, i.e., $p = 0.95$) if the measured value of T_1 is less than the table value of $8/15 = 24/45$. Since our measured value of 18/45 is smaller than the table value of 24/25, we conclude that we have met the test and that the hypothesis tha the distributions are the same is true with a significance value of 5%. We now want to see if we can raise the significance value to a higher number.



One-Sided Test:		$p = .90$	.95	.975	.99	.995
Two-Sided Test:		$p = .80$	.90	.95	.98	.99
$N_1 = 7$	$N_2 = 8$	27/56	33/56	5/8	41/56	3/4
	9	31/63	5/9	40/63	5/7	47/63
	10	33/70	39/70	43/70	7/10	5/7
	14	3/7	1/2	4/7	9/14	5/7
	28	3/7	13/28	15/28	17/28	9/14
$N_1 = 8$	$N_2 = 9$	4/9	13/24	5/8	2/3	3/4
	10	19/40	21/40	23/40	27/40	7/10
	12	11/24	1/2	7/12	5/8	2/3
	16	7/16	1/2	9/16	5/8	5/8
	32	13/32	7/16	1/2	9/16	19/32
$N_1 = 9$	$N_2 = 10$	7/15	1/2	26/45	2/3	31/45
	12	4/9	1/2	5/9	11/18	2/3
	15	19/45	22/45	8/15	3/5	29/45
	18	7/18	4/9	1/2	5/9	11/18
	36	13/36	5/12	17/36	19/36	5/9
$N_1 = 10$	$N_2 = 15$	2/5	7/15	1/2	17/30	19/30
	20	2/5	9/20	1/2	11/20	3/5
	40	7/20	2/5	9/20	1/2	
$N_1 = 12$	$N_2 = 15$	23/60	9/20	1/2	11/20	7/12
	16	3/8	7/16	23/48	13/24	7/12
	18	13/36	5/12	17/36	19/36	5/9
	20	11/30	5/12	7/15	31/60	17/30
$N_1 = 15$	$N_2 = 20$	7/20	2/5	13/30	29/60	31/60
$N_1 = 16$	$N_2 = 20$	27/80	31/80	17/40	19/40	41/80
Large-sample approximation		$1.07\sqrt{\frac{m+n}{mn}}$	$1.22\sqrt{\frac{m+n}{mn}}$	$1.36\sqrt{\frac{m+n}{mn}}$	$1.52\sqrt{\frac{m+n}{mn}}$	$1.63\sqrt{\frac{m+n}{mn}}$

Figure 2.11: Quantiles of the Smirnov test statistic for two samples. (From W.J. Conover, *Practical Nonparametric Statistics*, John Wiley and Sons, 1971.)

Two-sided Test	$p = 0.80$	0.90	0.95	0.98	0.99
Value	$1.07\sqrt{\frac{m+n}{mn}}$	$1.22\sqrt{\frac{m+n}{mn}}$	$1.36\sqrt{\frac{m+n}{mn}}$	$1.52\sqrt{\frac{m+n}{mn}}$	$1.63\sqrt{\frac{m+n}{mn}}$

Table 2.2: Large–sample approximation for the Smirnov test statistic.

We can accept the hypothesis with a significance of 20% ($= 1 - 0.80$, i.e., $p = 0.80$) if the measured value of T_1 is less than the table value of 19/45. Since our measured value of T_1 is 18/45, we conclude that we have met the test and that the hypothesis that the distributions are the same is true with a significance value of 20%. Unfortunately the table does not contain values for higher levels of significance; we cannot try to raise the significance value any higher.

For our samples we will be dealing with many more particles in a sample (on the order of 1,000 particles). For this large–sample case we use the large–sample approximation on the bottom line of of the Table 2.1, repeated in Table 2.2.

We now proceed to the use of this test to our particle distributions. Carrier [9] sized the particles in two independent samples consisting of 909 particles and 1,059 particles. (As described before, previous intuitive measurements of particle distributions had shown that about 1,000 particles were needed to minimize changes in the size distribution.) Two cumulative frequency distributions were plotted with the aid of the Statgraphics statistical analysis program. (Statgraphics is supposed to be able to do the Kolmogorov–Smirnov test directly, but bugs in the program prevented us from using it.) Manipulation of the distributions in a spreadsheet program showed that $T_1 = 0.142$. Using $m = 909$ and $n = 1059$, the values in the test table are as shown in Table 2.3 Using the 5% significance value as an arbitrary desired level of significance, we see that our measured value of $T_1 = 0.142$ exceeds the table value of 0.0615, so our conclusion is that the samples do *not* have the same distribution (to within a 5% significance level) and larger sample sizes are required.

The experiment was repeated with sample sizes of 1,059 and 1,179 samples. The measured value of T_1 was 0.020. The large–sample table was recalculated for $m = 1059$ and $n = 1179$ with the results shown in Table 2.4. Again using the 5% significance as an arbitrary desired level of

Two–sided Test	$p = 0.80$	0.90	0.95	0.98	0.99
Value	0.0484	0.0522	0.0615	0.067	0.0737

Table 2.3: Large–sample approximation for the Smirnov test statistic for $m = 909$ and $n = 1059$.

Two–sided Test	$p = 0.80$	0.90	0.95	0.98	0.99
Value	0.0453	0.0517	0.0576	0.0644	0.0690

Table 2.4: Large–sample approximation for the Smirnov test statistic for $m = 1059$ and $n = 1179$.

confidence, we see that our measured value of $T_1 = 0.020$ is less than the table value of 0.0576, so our conclusion is that the samples *do* have the same distribution (to within a 5% significance level) and larger sample sizes are not required. In fact, inspection of the values in Table 2.4 shows that we can raise the significance factor to 20% or more for our small value of T_1.

Hence we conclude that we need at least 1100 particles in the sample. More particles will raise the significance, but this is a diminishing return (i.e., equal increases in sample size bring diminishing increases in the significance level).

Two points should be made about the test that was performed. First, we assumed that the distributions obtained were independent even though they were from the same hologram. Strictly speaking, the samples should be taken from different holograms made from different firings. Second, we have assumed that the two distributions were "representative" of all distributions of the same sample size (i.e., there are no large anomalies in the distribution). This assumption should be checked by accumulating and comparing many distributions from the hologram data. Also, the 5% significance level was arbitrarily chosen; more information needs to be developed to verify that this significance level (or the 20% significance level of our samples) is appropriate.

2.4 Summary

In this Chapter we have presented the sizing results obtained from calibration objects and a hologram made during a motor firing. The sizing algorithms are checked against a white–light image of the rectangular portion of the LEOS calibration array whose sizes are known (see Figure 1.7 on page 10). Initial results by Redman showed that resolution was a potential problem as he could resolve only to 20 μm. Since the hologram was predicted to have better resolution capability than that (and was observed to have better resolution when the hologram reconstruction was observed by eye), we felt that Redman's crude speckle–reduction filter was severely degrading the resolution. Efforts on speckle reduction techniques followed and are reported in the next chapter.

Following Edwards' work on speckle reduction, Orguc returned to the particle–sizing problem and produced measured results for both the LEOS calibration object and a hologram reconstruction of a motor firing. His histogram results have been presented in this chapter.

To minimize the amount of data required to reach a representative distribution, Carrier used the Kolmogorov–Smirnov two–sided test to show that about 1,200 particles were necessary to achieve a stable distribution in his data set. This corroborated an intuitive estimate of 1,000 particles that had been made in our prior work.

Chapter 3

SPECKLE REDUCTION

During laser reconstruction of the hologram, speckle noise is introduced due to the optical diffuser that was required to avoid imaging the thermal gradients present in the combustion process. This speckle noise can be of the same size as some of the smaller particles of interest, and so it can cause a faulty count in the number of features present. In an attempt to reduce the speckle, Redman [6] investigated a imaging the hologram reconstruction on a moving screen (a spinning mylar disk), averaging techniques, blurring filters, and lowpass filtering. Twenty microns was the best resolution that was achieved after the application of these filters. Some of the loss in resolution may have been due to the optics involved; nevertheless, it became necessary to explore other means of filtering out the speckle to get the best possible resolution.

Edwards [7] explored three speckle reduction algorithms suggested by work in the field of synthetic aperture radars. The initial step was to write computer programs to support each algorithm. Once the programs were running, a comparison was done to find which of the three was best at reducing speckle while retaining as much resolution as possible.

Carrier [9] investigated the possibility of digitally averaging images with different speckle patterns to reduce the speckle. He also quantified the speckle reduction due imaging the reconstruction on the spinning mylar disk for comparison with the other speckle–reduction techniques.

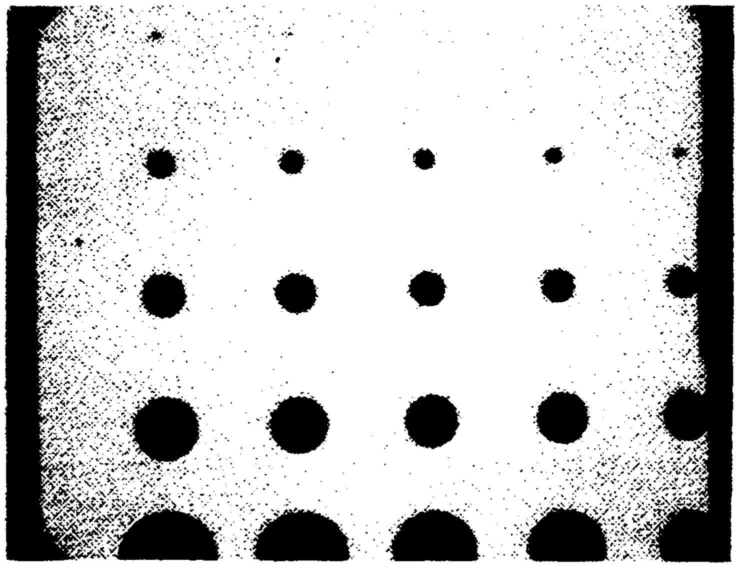


Figure 3.1: Image without speckle present.

3.1 Speckle index

Speckle has the characteristics of random multiplicative noise in the sense that the noise level increases with the average gray level of a local area [17]. The presence of speckle reduces one's ability to resolve fine detail. Figures 3.1 and 3.2 show images without and with speckle present. It can be seen how the speckle is easily confused with the smaller feature particles.

As a figure of merit in determining the amount of speckle reduction, the "speckle index" was used. In Reference [18], Crimmins showed that the ratio of local deviation to local mean was a reasonable quantitative measure of speckle, due to the multiplicative nature of speckle noise.

To compute the *speckle index*, a FORTRAN subroutine was written. The algorithm used was [19]:

$$\text{speckle index} = \frac{1}{MN} \sum_{x=1}^{M} \sum_{y=1}^{N} \frac{\text{dev}}{\text{mean}} \tag{3.1}$$

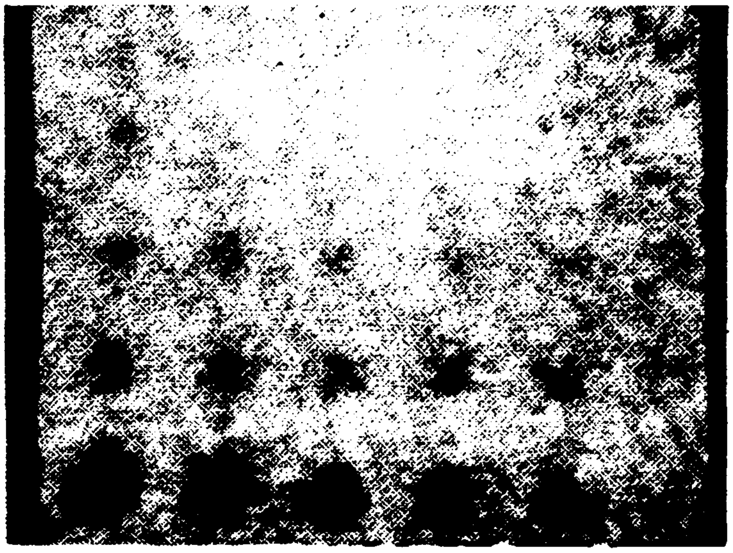


Figure 3.2: Image with speckle present.

where:

$$\text{dev} = \max_{-1\leq a,b\leq 1} p(x+a, y+b) - \min_{-1\leq a,b\leq 1} p(x+a, y+b) \tag{3.2}$$

$$\text{mean} = \frac{1}{9} \sum_{a,b=-1}^{1} p(x+a, y+b) \tag{3.3}$$

In the above equations, M and N are the dimensions of the array in the x and y directions, and $p(x, y)$ is the gray level of an individual pixel located at x, y.

In simple terms, the *local deviation* is found by subtracting the smallest gray level of a particular pixel and its eight immediate neighbors from the largest gray level of the same nine pixels. The *local mean* is the average of the nine gray levels. The subroutine was run with various values of M and N to determine values for the minimum-size region that needs to be used for the speckle index calculation. From Figure 3.3, it can be seen that a small array can distort the calculation if there is a sharp local disturbance.

Based on the results of Figure 3.3, the values $M = N = 240$ were chosen as representative array dimensions for speckle index calculations.

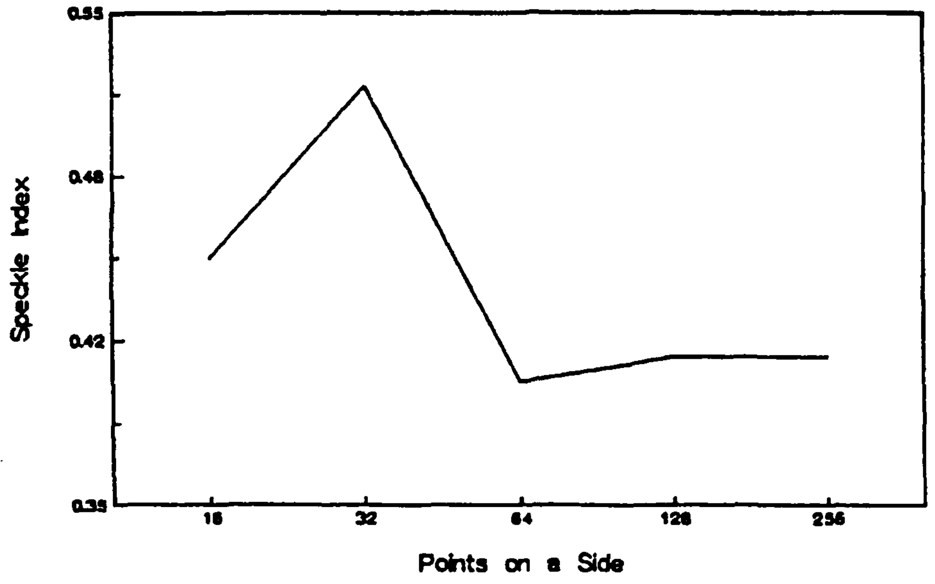


Figure 3.3: Speckle index as a function of the area of computation.

This size represents about 23% of the visible screen, yet does not take an excessive amount of time to calculate. A 16 MHz IBM PC/AT with the math coprocessor returns the speckle index of an image to the calling program in 75 seconds. Values for the speckle index can range from over 1.0 (rarely) to a theoretical limit of zero. The value of zero could only be achieved if the entire image was filtered to the point where every pixel had the same gray level. This is, of course, an uninteresting image.

3.2 Speckle reduction filters

Three filters were implemented in software and will now be discussed. Two filters depend to a certain degree on the statistics of the image. The third filter uses a geometric hulling algorithm.

3.2.1 The sigma filter

The first speckle reduction filter [20] is based on the standard deviation of a Gaussian distribution. By definition, 95.5% of all the pixels fall within two standard deviations on either side of the mean. Any pixels within two standard deviations of a given pixel's gray level are included in an averaging scheme, whereas those outside the "2–sigma" range are excluded. Obviously, the standard deviation of the image must be known beforehand. ImageAction was used to find standard deviations and histogram plots very quickly and easily. (The later version of the ImageAction*plus* software did not calculate the standard deviation of the image, so a software routine was written to find this value.)

If a particular pixel is considerably different from its neighbors, perhaps none of the neighbors will be within the 2–sigma range. This would indicate a very sharp feature. To avoid the possibility that this sharp feature will not be subject to the averaging process at all, a cutoff is established. If the total number of pixels inside the 2–sigma range is less than this minimum cutoff number (2 was chosen), the four–neighbor average then replaces the central pixel's gray level value.

The sigma filter program uses the algorithm as follows [20]:

$g(x, y) =$

$$\begin{cases} \dfrac{\sum_{i=x-2}^{x+2}\sum_{j=y-2}^{y+2}\delta(i,j)p(i,j)}{\sum_{i=x-2}^{x+2}\sum_{j=y-2}^{y+2}\delta(i,j)} \\ \qquad \text{if } \sum_{i=x-2}^{x+2}\sum_{j=y-2}^{y+2}\delta(i,j) > 2 \\ \\ \dfrac{p(x-1,y)+p(x+1,y)+p(x,y)+p(x,y-1)+p(x,y+1)}{5} \\ \qquad \text{otherwise,} \end{cases} \tag{3.4}$$

where

$p(x,y)$ = gray level of a pixel in 5x5 local array,

$g(x,y)$ = gray level of the filtered central pixel,

σ = standard deviation, and

$$\delta(x,y) = \begin{cases} 1 & \text{if } (1-2\sigma)p(x,y) \le p(i,j) \le (1+2\sigma)p(x,y) \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

By altering the value of σ, varying degrees of filtering result. If σ is increased, the sigma range is increased and so more pixels are included in the average. For this work, however, the true standard deviation of each image was used when it was filtered.

For illustration of the sigma filter's effectiveness, images of the Air Force Resolution Target are presented. Figures 3.4 and 3.4 shows an image before and after two iterations of the filter. Visually, the speckle has been reduced considerably, although the edges have been blurred and the resolution has decreased somewhat. Numerically, the speckle index has decrease from 0.305 to 0.103.

When a threshold is applied at a value of 140 (iteratively determined as the best value) the effectiveness of the sigma filter is most dramatically displayed. (See Figures 3.6) and 3.7.) Now it is much more evident that good data can be retrieved from the filtered image, whereas the excessive noise in the unfiltered image would lead to faulty data detection.

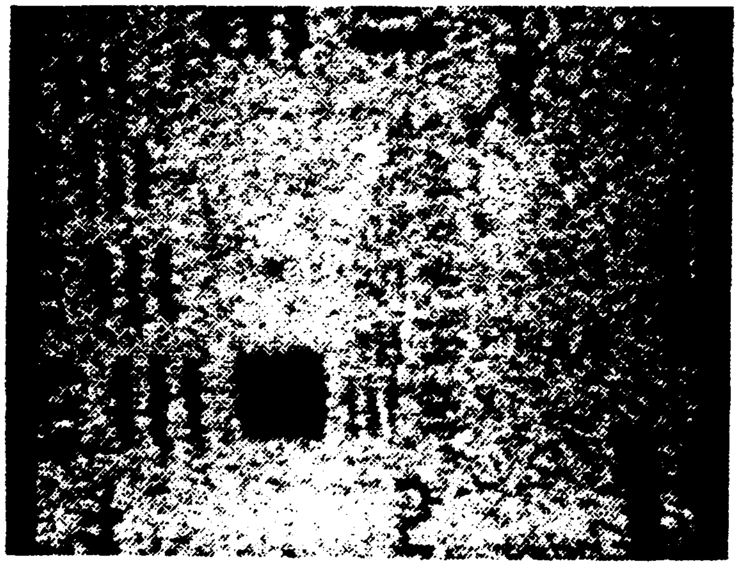


Figure 3.4: Image of Air Force resolution chart before filtering ($SI = 0.305$).

3.2.2 The local statistics filter

This filter [20] uses local estimates of the mean and variance in a 5x5 window about the central pixel in question.

These local statistics are used to calculate a weight k. The k value then determines where the new gray level of the central pixel will be placed: near the original value of the central pixel, near the linear average of all pixels in the 5x5 array, or somewhere in-between. Throughout this algorithm, $p(x,y)$ represents unfiltered pixels and $g(x,y)$ represents the filtered central pixel. To compute the filtered gray level of a pixel using the local statistics method, the local mean $E\{g(x,y)\}$ and variance $\text{var}\{g(x,y)\}$ must first be estimated [20]:

$$E\{g(x,y)\} \approx \frac{1}{25} \sum_{i=x-2}^{x+2} \sum_{j=y-2}^{y+2} p(i,j) \tag{3.6}$$

$$\text{var}\{g(x,y)\} \approx \frac{\text{var}\{p(x,y)\} + E^2\{g(x,y)\}}{\sigma^2 + 1} - E^2\{g(x,y)\}. \tag{3.7}$$

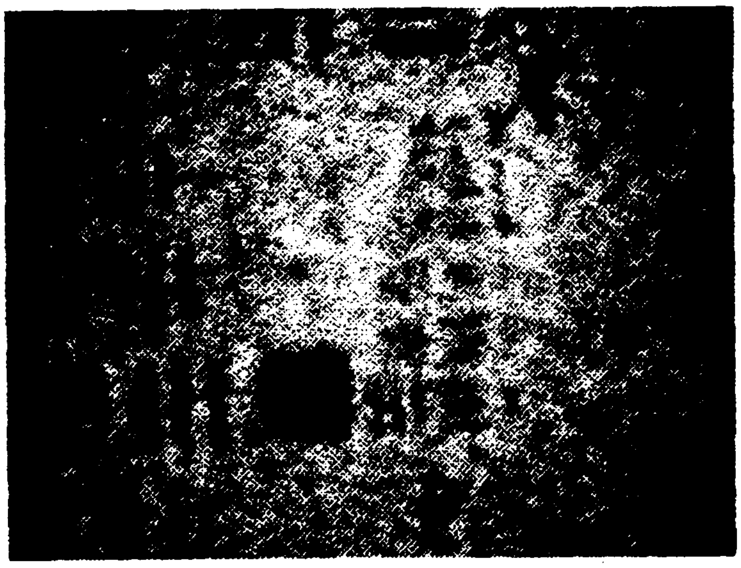


Figure 3.5: Image of Air Force resolution chart after two iterations of the sigma filter ($SI = 0.103$).

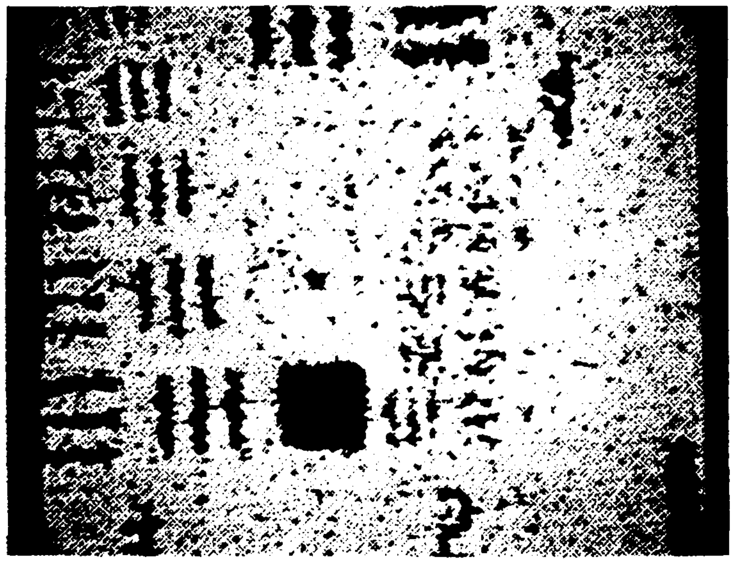


Figure 3.6: Thresholded image without any filtering applied. (Threshold value = 140.)

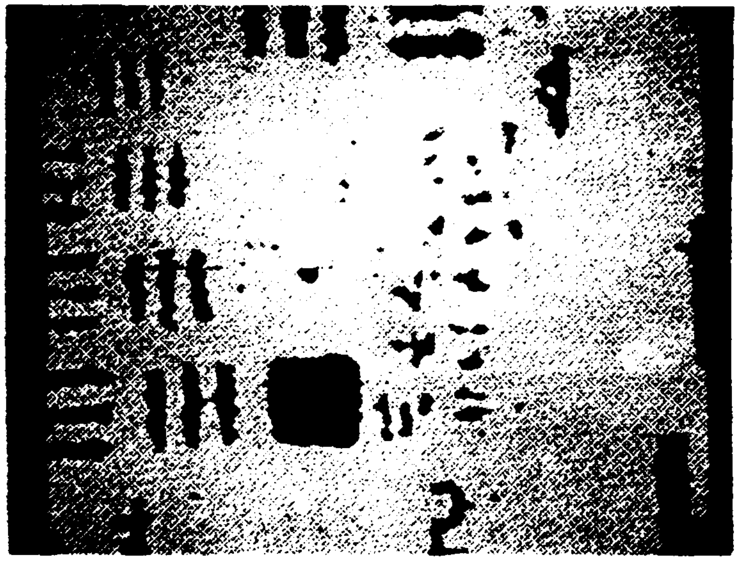


Figure 3.7: Thresholded image after application of Sigma filter. (Threshold value = 140.)

where

$$\mathrm{var}\{p(x,y)\} \approx \frac{1}{25}\sum_{i=x-2}^{x+2}\sum_{j=y-2}^{y+2}[p(i,j) - E\{g(x,y)\}]^2 . \tag{3.8}$$

The weight k may then be calculated as

$$k = \frac{\mathrm{var}\{g(x,y)\}}{\sigma^2 E^2\{g(x,y)\} + \mathrm{var}\{g(x,y)\}} . \tag{3.9}$$

Finally, the estimate of $g(x,y)$ is

$$g(x,y) = E\{g(x,y)\} + k\,[p(x,y) - E\{g(x,y)\}\,] . \tag{3.10}$$

From close scrutiny of the algorithm, some of the equations may be better understood. Equation 3.6 states that the local mean is the average of the 25 pixels in the 5x5 local array. Equation 3.8 is the variance of the unfiltered pixels in the 5x5 array; the mean value of the local array from Equation 3.6 is subtracted from each of the pixels in turn. These differences are squared and then summed over the entire array. For more detail on how the remaining equations were developed, see Ref. [20].

Consider a region of an image which is flat, meaning adjacent pixels have about the same gray level. Here, the variance approaches zero, and so k approaches zero. The new estimate of $g(x,y)$ is therefore close to the mean of the local 5x5 array. Consider now a region of high contrast, say the edge of a particle. Here, k approaches one and so $g(x,y)$ is close to the value of the central pixel. The filter therefore has a marked tendency toward the retention of high contrast edges [20]. Figure 3.8 and Figure 3.9 show the Air Force Resolution Target (of Figure 3.4 on page 37) after two iterations of the local statistics filter, before and after thresholding at 140.

3.2.3 The geometric filter

Whereas the previously discussed filters depend to a certain degree upon the statistics of the image to be filtered, the geometric filter is based on nonlinear geometric concepts. The algorithm was developed in Reference [18] and [19]. It is a one-dimensional routine which is run horizontally, vertically and then in the two diagonal directions. In each direction it applies a geometric hulling algorithm to the image and then to the image's complement. The algorithm [18] for this filter follows:

1. Let $a = 1$, $b = 0$. (This sets the values for a horizontal run.)

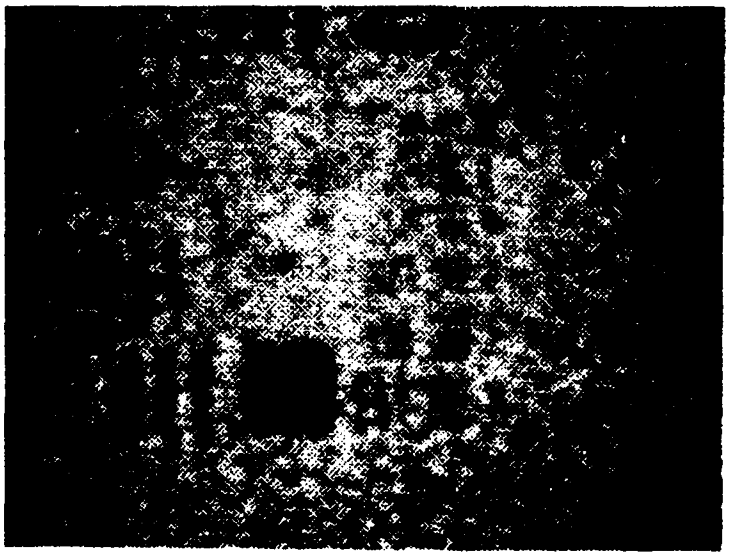


Figure 3.8: Image (before threshold operation) after two iterations of local statistics filter ($SI = 0.115$).

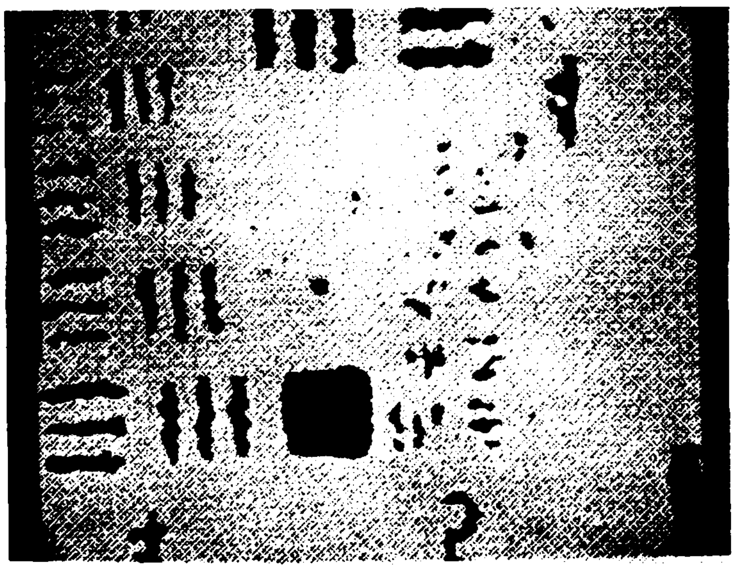


Figure 3.9: Thresholded image of Air Force resolution chart after two iterations of local statistics filter. (Threshold value = 140.)

2. Let $c = 3$. (This is a counter to determine when to change directions.)

3. Let $d = 1$. (This controls whether the image or its complement is being filtered.)

4. Compute

$$g(x,y) = \max[p(x,y), \min\{p(x-a, y-b) - 1, p(x,y) + 1\}] \quad \text{for } 1 \leq x \leq M, 1 \leq y \leq N. \tag{3.11}$$

5. Compute

$$p(x,y) = \max[g(x,y), \min\{g(x-a, y-b), g(x,y) + 1, g(x+a, y+b) + 1\}] \quad \text{for } 1 \leq x \leq M, 1 \leq y \leq N. \tag{3.12}$$

6. If $d = 1$, let $a = -a$, $b = -b$, $d = 0$ and go to Step 4.
   If $d = 0$, let $d = 1$ and go to Step 7.

7. Compute

$$g(x,y) = \min[p(x,y), \max\{p(x-a, y-b) + 1, p(x,y) - 1\}] \quad \text{for } 1 \leq x \leq M, 1 \leq y \leq N. \tag{3.13}$$

8. Compute

$$p(x,y) = \min[g(x,y), \max\{g(x-a, y-b), g(x,y) - 1, g(x+a, y+b) - 1\}] \quad \text{for } 1 \leq x \leq M, 1 \leq y \leq N. \tag{3.14}$$

9. If $d = 1$, let $a = -a$, $b = -b$, $d = 0$ and go to Step 7.
   If $d = 0$, let $d = 1$ and go to Step 10.

10. If $c = 3$, let $a = 0$, $b = 1$, $c = 2$ and go to Step 4. (This sets up a vertical run.)
    If $c = 2$, let $a = 1$, $b = 1$, $c = 1$ and go to Step 4. (This sets up a diagonal run.)
    If $c = 1$, let $a = 1$, $b = -1$, $c = 0$ and go to Step 4. (This sets up the other diagonal.)

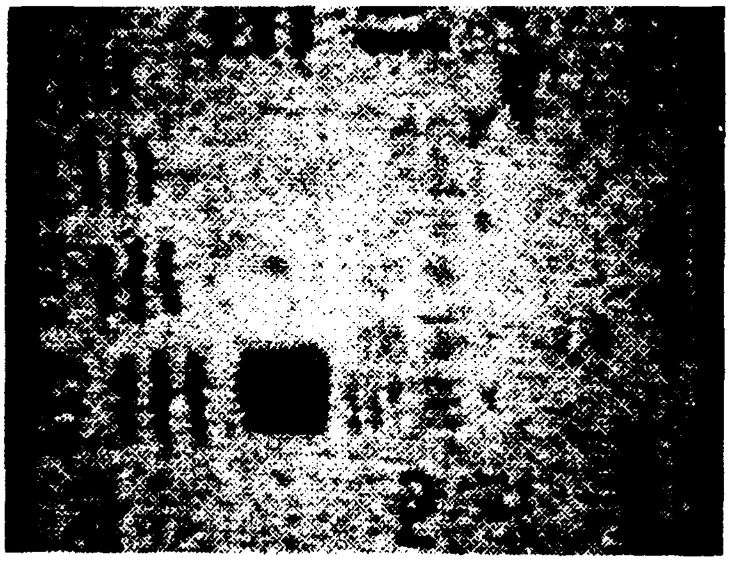


Figure 3.10: Image after two iterations of geometrical filter ($SI = 0.129$) before application of threshold operation.

11. If $c = 0$, stop.

The algorithm used exhibits the interesting property that it will allow a certain curvature in the terrain, but will tear down anything more drastic. With a few exceptions [19], a curve sharper than 45 degrees at any vertex will be filtered. What this means is that the narrow valleys comprising speckle will be filled in and the walls torn down after sufficient iterations of the filter. Of course, the objects of interest that should be preserved are also filled in and torn down, but at a much slower rate. Generally, the larger and darker the feature, the more slowly it will be degraded.

If an object of interest is almost as narrow as the speckle, they both will be reduced at about the same rate. Hopefully, the object is much darker than the speckle, so the speckle will be beaten down after a number of iterations leaving the object of interest largely intact.

Figure 3.10 and Figure 3.11 show the Air Force Resolution Target (of Figure 3.4 on page 37) after three iterations of the geometrical filter, both before and after thresholding at 140.

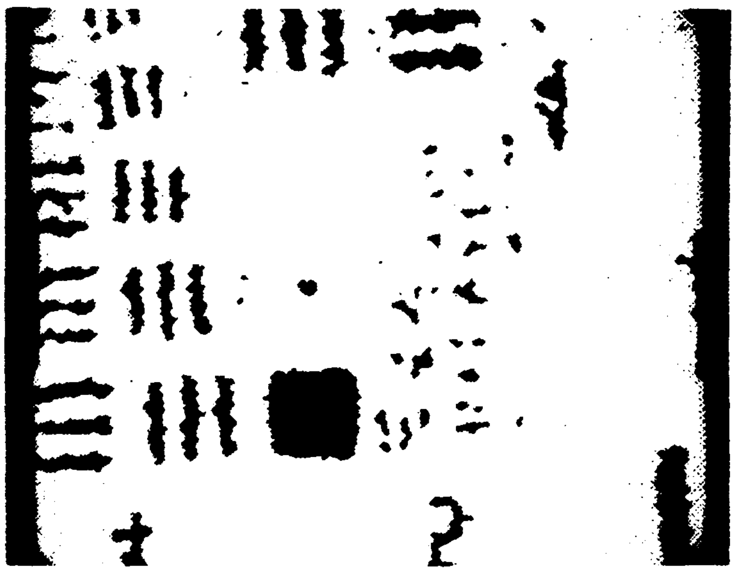


Figure 3.11: Thresholded image of Air Force resolution chart after three iterations of geometrical filter. (Threshold level = 140.)

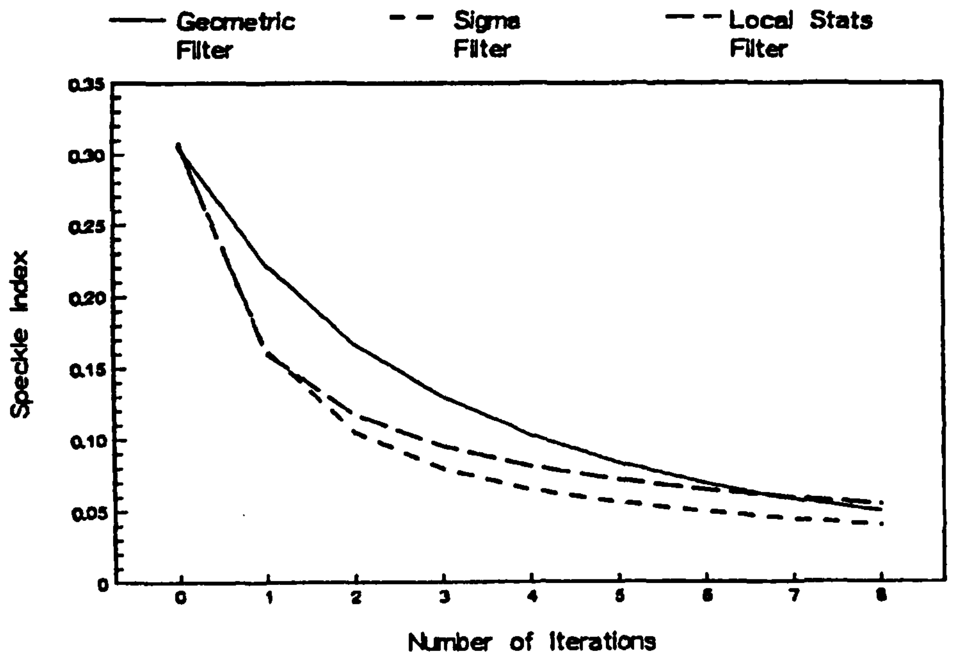


Figure 3.12: Reduction in speckle vs. the number of iterations for the three speckle reduction filters.

3.3 Comparison of the filters

This section will compare the three filters using the speckle index as a figure of merit in gauging how much speckle has been removed. The filters will be compared in how much speckle they remove per iteration, the computation time per iteration, histogram differences, and, of course, visual differences.

3.3.1 Speckle index comparison

All three filters reduce speckle at their own rate. Figure 3.12 shows this fact. The sigma and local statistics filters both reduce speckle by almost a factor of two after only one iteration, but then dramatically "slow down" with increased iterations. This may be attributed to the fact that both filters are actually various themes on the blurring technique. More iterations serve to

FILTER	TIME
Sigma	7 min, 20 sec
Geometric	18 min, 50 sec
Local Statistics	39 min, 45 sec

Table 3.1: Time per iteration (512x512 image, 16 MHz PC/AT)

merely increase the blur. Figure 3.12 shows that the geometric filter's curve is more gradual, allowing the user to stop at a less severe level of filtering. This is a particularly desirable trait, because resolution degradation becomes a problem after only a few iterations. It should be noted that filtering cannot stop at an arbitrary value of speckle index, only at discrete levels depending on how many iterations have occurred. It is foreseeable that, in general, a given filter may allow a degree of speckle reduction not achievable by the others although our three particular filters all approach approximately the same value of speckle index after several iterations.

On the basis of Figure 3.12, the geometric filter ranks first in its ability to control the amount of speckle reduction, followed by the local statistics filter.

3.3.2 Resolution comparison

Speckle reduction is a fundamental tradeoff against image resolution. Figure 3.13 shows a comparison of the measured resolution for each of the filter types. The horizontal axis shows the number of iterations of the filter. (Each iteration reduces the speckle further.) It is observed that the resolution degrades from about 12 μm to almost 18 μm after several iterations of the filters. The geometric and local statistics filters are comparable to each other up through seven iterations, and are superior to the sigma filter in preserving the resolution of the image.

3.3.3 Processing time comparison

The times for an iteration of each filter are in Table 3.1 The local statistics filter is slow due to the nested do–loops, frequent calls to subroutines, and a large number of calculations required inside each do-loop. It should be noted that speed is of secondary importance since there is no requirement in

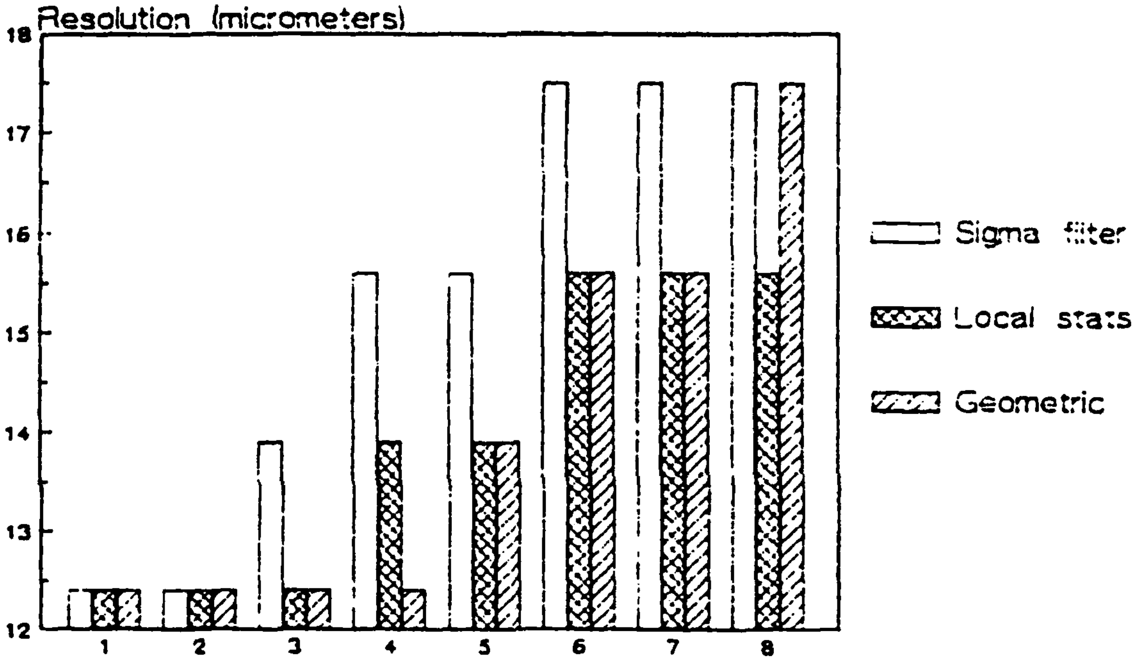


Figure 3.13: Image resolution (measured from Air Force resolution target) vs. number of iterations for the three speckle reduction filters.

this application to achieve results within a given time limit. Also, dedicated image–processing equipment can do the calculations much more efficiently and faster. For example, Reference [19] reports times of 14 seconds per iteration for the geometric filter using a VAX 11/780 and DeAnza IP–5500 digital video processor.

3.3.4 Visual comparison

The ultimate test of any procedure dealing with images is how the image looks to the viewer. Comparisons of this type are difficult to justify, for each person sees an image slightly differently. Nevertheless, based on numerous runs of the filters, the geometric filter is judged best at retaining the edges, basic shape and size of objects of interest while beating down the speckle. This can be seen by close comparison of Figures 3.5, 3.8, and 3.10. All three figures are close in speckle index, yet the local statistics and sigma filters tend to smear and spread particle edges more than does the geometric filter. This observation has been verified in runs using other images as well. Once the threshold at 140 is applied (Figures 3.7, 3.9, and 3.11), the sigma filter is definitely seen as the inferior of the three. The other two are very close in performance. Close examination reveals a few small particles of speckle that the geometric filter failed to eliminate which the local statistics filter successfully handled. This is a consequence of the filtering stopping at discrete levels. In most other cases, the geometric filter is superior.

3.3.5 Histogram comparison

All three filters produced histograms generally similar in shape and distribution. Figures 3.14 and 3.15 shows the histograms of two filtered images. Notable differences include the geometric filter's reduction of the spike at 255, the way it retained virtually intact the darker features, and its more pronounced "valley" between the two "humps".

The claim was made earlier that the geometric filter had a tendency to reduce smaller (hopefully speckle) particles much faster than the dark features. The retention of the darker pixels in the histogram is evidence that this claim is valid. All the filtered histograms pointed to a threshold at about 140. Thus, this was the value used on all the images, including the original, for comparison purposes.

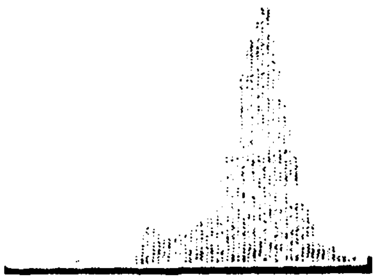


Figure 3.14: Histogram of image filtered with two iterations of the local statistics filter.

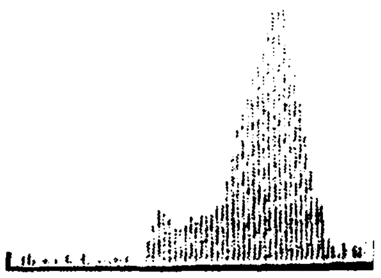


Figure 3.15: Histogram of image filtered with three iterations of the geometric filter.

3.4 Moving–screen averaging

Another speckle–reduction technique is to record the desired image when focused on a moving screen. If the screen moves, then the speckle pattern will change while the particle images will remain constant. If the speckle pattern changes during the integration time of the detector (1/30th of a second for a TV camera), the averaging effect of the detector integration will cause the speckle to "wash out" while the particle images remain the same. This method combines an easily–implemented technique with one that achieves good speckle reduction while maintaining resolution of small features. Our measured value of speckle index for this technique is 0.0593 and the measured resolution is 12.4 microns. This value of speckle index and resolution are comparable to the best of the nonlinear speckle–reduction filters after several iterations. In terms of ease of implementation and speed, this moving–screen averaging technique is far superior to the nonlinear filters. It is recommended that all hologram reconstruction images be recorded from images focused on a moving screen and that the speckle–reduction filters be used only to further reduce the speckle.

3.5 Multiple–image averaging

We wanted to find out just how good the speckle reduction to a speckle index value of 0.0593 was and how the achieved resolution of 12 micrometers resolution was, compared to a simple averaging technique. In the simple averaging technique [14], N separate images with the same resolution chart image but different speckle patterns were recorded and digitized. Then N digitized images were then successively registered and averaged in the computer with the result displayed on the screen. (The number of average images ranged from 1 to N.) The speckle index was calculated from the averaged data and the resolution was measured by an observer looking at the image of the resolution chart. This method is known to theoretically produce a reduction in speckle that is proportional to $1/\sqrt{N}$.

Figure 3.16 displays the relative value of speckle index as a function of the number of averaged images N. Since we are interested in the reduction of speckle index, we have normalized the speckle index to the value for $N = 1$. The dashed curve shows the theoretical reduction in speckle as $1/\sqrt{N}$. Relatively good agreement is shown. Differences are probably attributable to small registration problems with the images to be averaged. The solid

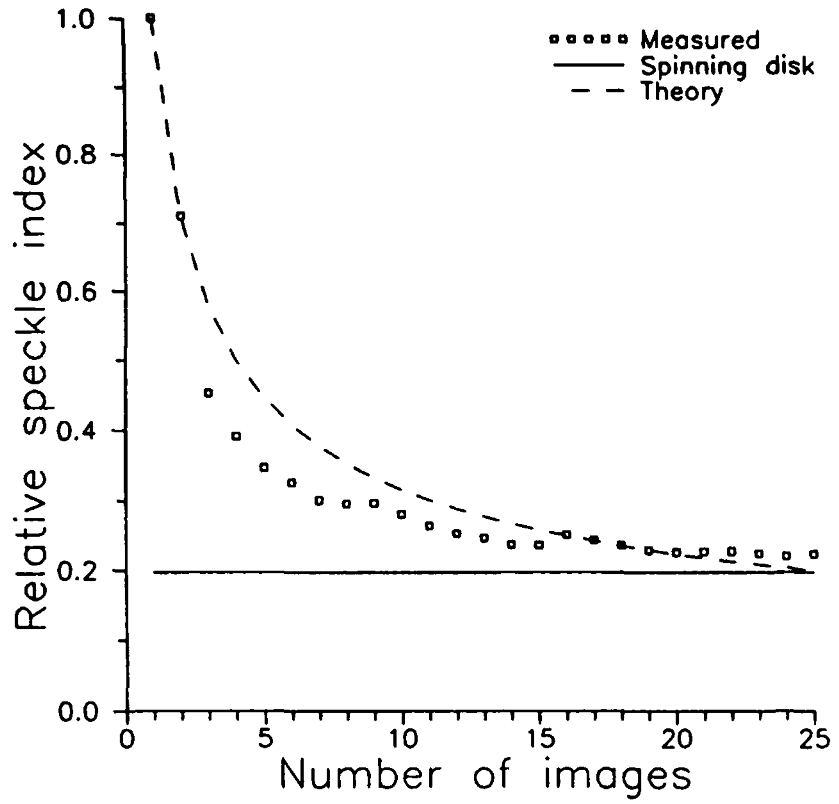


Figure 3.16: Reduction in relative speckle index vs. number of images that are averaged. Also shown for reference is the speckle index of an image recorded from a moving screen.

line on the graph shows the speckle index measured from an image taken with the moving screen. (In our experimental setup, the moving screen is made by recording the images formed on a spinning mylar disk.) The data show that the moving screen technique is better than the averaging of over 25 separate images. Further reduction in the speckle by averaging even more images is not advised, since the knee of the curve has been passed and the averaging of many more images would be required to achieve only minimal reduction in speckle.

Figure 3.17 shows the measured resolution of the averaged images. Contrary to the application of the convolution filters or nonlinear speckle reduction filters, the resolution of the result improves rather than degrades. This improvement is due to the removal of the overlying speckle that masks the smaller features. (The ultimate resolution is determined by the limitations of the hologram recording and reconstruction geometry and the ability of the technique to exactly align the multiple images being averaged.) As the speckle is reduced, the resolution improves by almost a factor of two. (The measured resolution is quantized to the values of the AF Resolution chart.) Also shown in Figure 3.17, for reference, is the measured resolution of the image recorded with the moving screen. It is seen that the resolution of the image from the moving screen has better resolution than the best of the averaged images.

3.6 Conclusions

The IBM PC/AT with dedicated software and hardware is a viable system to reduce speckle in reconstructed holograms. The ITEX/PC software with the PC/Vision frame-grabber board may be used in conjunction with FORTRAN programs to achieve the filtering desired. The best method of reducing the speckle is to record the image off of a moving screen. In our reconstruction system, we focused the reconstructed real image on a rotating circular mylar disk. (See Figure 1.1 on page 3). The screen must rotate fast enough to allow sufficient integration of the image over the 1/30th second frame time of the imaging tube. Modest rotation rates proved capable of providing enough velocity for our disk.

While all three nonlinear filters are superior to current linear blurring digital techniques, the geometric filter has been found to be the best of the three. It combines the ability to hold the edges and shape of objects with the desirable trait of less filtering per iteration. In this way, the user has

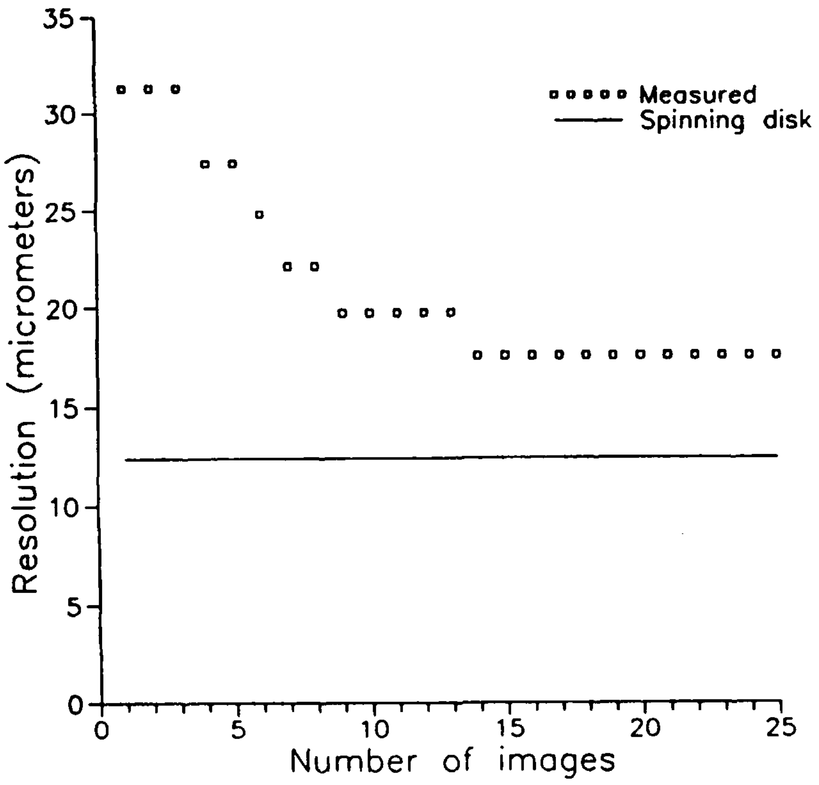


Figure 3.17: Measured image resolution after averaging images. Also shown for reference is the resolution of an image recorded from a moving screen.

more control over the amount of filtering to be done and to what degree the speckle will be reduced. While the geometric filter has been found to be the best overall filter, it may not be superior in every particular circumstance. It has been noted that due to the discrete jumps in speckle index which occur after every iteration, one filter may be able to reach the region of optimum filtering when another cannot. For this reason, the local statistics filter should be used along with the geometric filter and the results after 2 or 3 iterations compared.

Resolution of the original image was about 12 microns. This was degraded to about 14 microns in the filtered images. It is doubtful that much more improvement can be made toward reducing the speckle degradation due to filtering. Any further resolution improvements will most likely have to occur in the recording and reconstruction process, and better optics. Some improvements have been made recently in this area. Reference [7] reports resolutions typically of 8 microns, and in certain cases, 4 microns. Taking into account the degradation after filtering, a conservative estimate of 10 microns resolution is now possible in the finished filtered and thresholded image.

Digital averaging of images is not recommended when the moving-screen averaging technique can be used. Digital averaging requires large amounts of disk storage for the images recorded and a fairly long time to perform the average (due mainly to the disk reading operations required). It does serve as a useful benchmark, however, for comparison of the other imaging techniques as the mathematical description of the frame-averaging technique is well understood.

Chapter 4

CODE OPTIMIZATION

Significant effort was spent in this project to improve the speed and efficiency of the programs. Early efforts were limited to a 256x256 picture due to processing times (on the order of 4 hours) and memory constraints. We can now handle 512x512 images in approximately 10 minutes from frame digitization to production of the data table.

Improvements have been due to the commercial availability of software that manipulates the fast-access video memory board, improved software revisions on locally-produced programs, and the substitution of a faster microprocessor. Further reductions in processing time, if desired, can be made by using more powerful and more expensive computer hosts for the image processing board. The final time required for our computations is on an appropriate scale for the feasibility studies that we are carrying out. Several iterations have been made in improving the execution of the code as new optimizing compilers became available and as new image-processing subroutine packages became available from Imaging Technology.

4.1 FORTRAN programs

4.1.1 Original FORTRAN program

Redman [6] began the study on the PC after investigations concluded that the Quantimet system was too unreliable for use. He used Microsoft FORTRAN together with the ImageAction software for his routines. ImageAction is a mouse-driven package that allows the user to manipulate the image. (The software-callable routines of ITEX/PC were not commercially available at the time of Redman's work.) He used the ImageAction software to retrieve and save images and to apply speckle–reduction convolution filters of his own design. The main feature identification and sizing algorithms were written into separate programs, in modular form. The typical frame contains 512x512 pixels, each represented by an 8–bit word, so each frame requires 262 kilobytes (kB) of memory for storage or manipulation. This large frame size, together with the Redman's programs, exceeded the 640 kB DOS limitation. (All of the Image Technology software that we used was operated under DOS. UNIX versions later appeared, but our hardware investment had already been made in the DOS–based machine.) Redman's solution was to manipulate only one–fourth of an image at a time (i.e., 64 kB). This was easily implemented since the one–fourth frame is an available option for all of the ImageAction manipulations. Also an operation called a "quad squish" would retain every fourth pixel and reduce the size of a frame to one–fourth of its original size (accompanied by a loss of three–fourths of the data). Redman also included a test object of his own devising to test out his algorithms as they were developed. Redman's programs performed the following operations:

- read the image frame into a 256x256 array in the computer memory, (optionally) presented the data values of the upper left corner of the image, and thresholded the image, by setting the array values to zero if their value exceeded the upper threshold limit or their values were less than the lower threshold limit specified;
- performed a feature identification algorithm that looked for connected non–zero pixels and labeled the connected pixels with a unique feature identification number (the algorithm is described in more detail in Ref. [6]); and
- sized the features by measuring the maximum horizontal width (the x–chord), the maximum vertical height (the y-chord), and the total number of pixels in the feature (the area of the feature). The results were each written to a data table for later statistical processing.

Each of these steps required a separate FORTRAN program that worked on the results of the previous step. Each step required that the array of values be saved at the end of the step and be retrieved at the beginning of the following step for subsequent processing.

Once the measurement data table was available, conversion from pixels to micrometers was done by the measurement of a known object that was

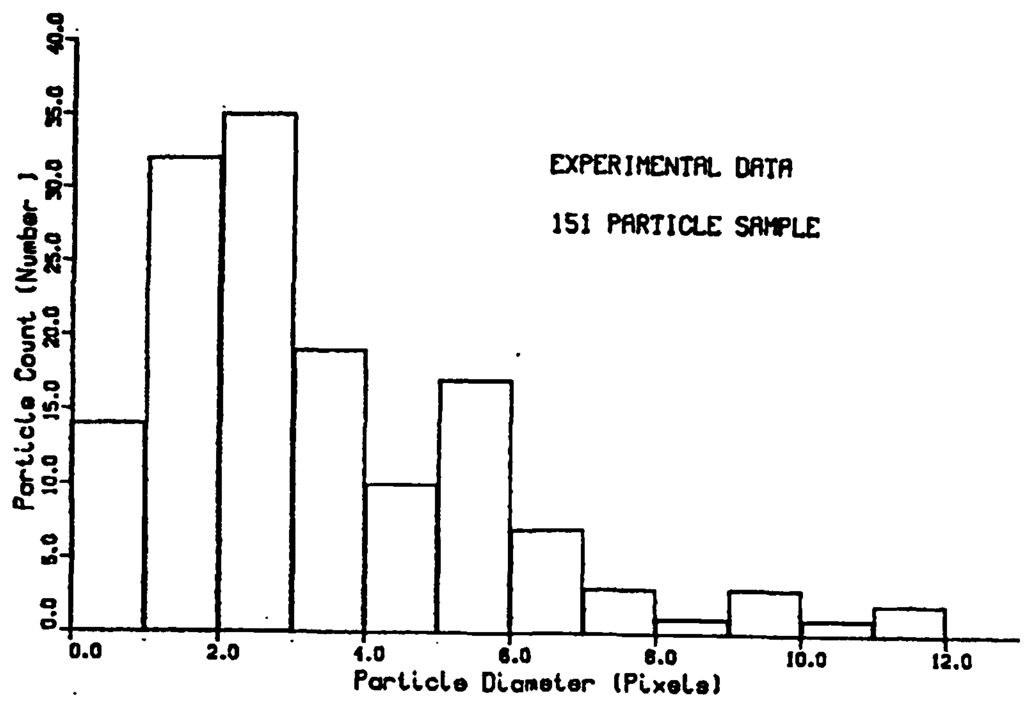


Figure 4.1: Measured histograms of x–chord length from 151 particles in the LEOS calibration test object.

recorded in the hologram. (The object was a screw with a known thread period.) The data was processed to produce histograms of the desired parameter. Sample results are shown in Figure 4.1 and Figure 4.2.

The memory limitations of the computer meant that a maximum of 250 particles could be measured in a single one–fourth frame. This was not very limiting in practice as there were never more particles than this in the field of view of the microscope. For a one–fourth frame to be processed it took 2–1/2 hours for the programs to run from start to finish on a 8 MHz PC/AT. (Redman also demonstrated the capability to perform the processing calculations on a mainframe computer, but the reduction in computing time was canceled by the time required to transfer the image data over slow telephone lines.)

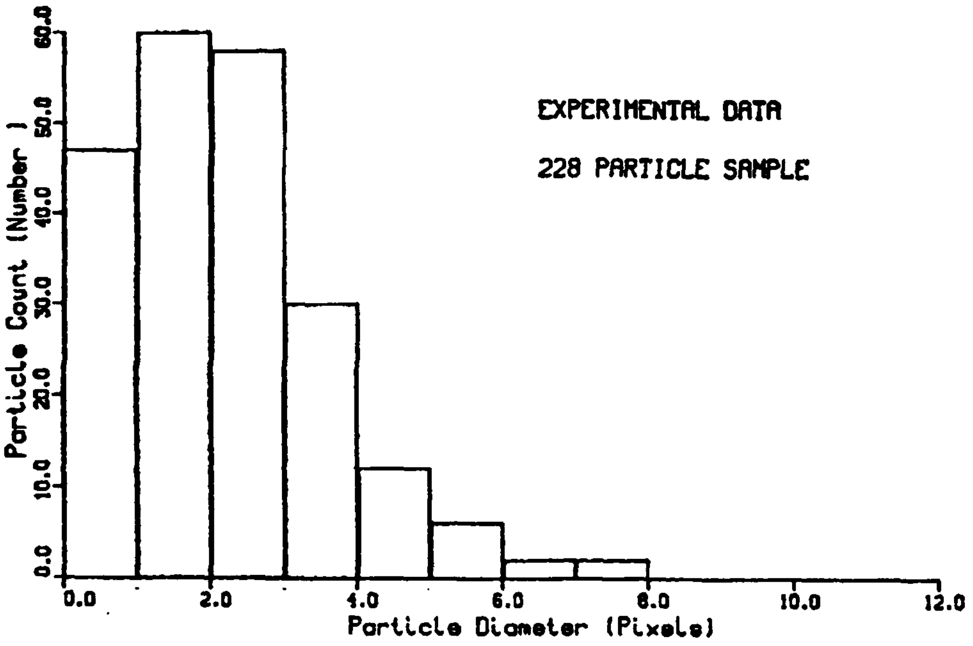


Figure 4.2: Measured histograms of x–chord length from 228 particles in the LEOS calibration test object.

The inability to process a full image and the long time required for the image processing resulted in a major effort to improve the calculation speed. A variety of techniques including the use of fast video memory, the use of optimizing compilers that later became available, an increase of the computer speed to 16 MHz, the use of virtual arrays, and the use other programming techniques all combined to vastly improve the speed and capacity of the processing system. These improvements are described in the following.

4.1.2 Modified FORTRAN program

Orguc [8] continued the particle measurement work begun by Redman. The ITEX/PC software had become commercially available and the computer processor had been upgraded to a 16 MHz 386-based system.

The ITEX/PC package unbundled the routines that had been built into the ImageAction software. The routines were written in Microsoft Pascal and were included in a library that could be linked to a user program written in either Microsoft FORTRAN, Pascal, or C. The routines were called by the master program with the input and output parameters passed as arguments to the routine. The key routines that were used in improving the performance of the processing program were:

1. pixel read and write routines – These routines allowed the user to read the value in a particular pixel location in the image processing board memory, manipulate the value within the computer, and to write a modified value back into the pixel location. These routines offered a dual advantage. The data memory was no longer in the operating memory of the computer, but was retained on the image processing memory board. No longer did an array equal to the frame size have to be declared within the computer memory for the image. Additionally, the image memory on the computer board was very fast and access was optimized to allow the board to handle real-time video images (i.e., 512x512 bytes of data every 1/30th of a second).

2. image save and retrieve – These routines allowed simple calls to replace the writing of the array to memory. Again the speed of the image processing memory made these read and write operations very fast. (The images were stored on Bernoulli disks with a 28 ms access time.)

3. threshold – The threshold operation was done with a call to the thresholding routine.

Orguc modified Redman's code to improve the efficiency of the algorithms, as well as to incorporate the new subroutine calls, and to decrease the processing time. The revised program consisted of three modular sections that had to operated sequentially (because the three combined programs still exceeded the 640 kB limit of DOS). The sections performed the following operations (details of these processing steps are found in Ref. [8].):

1. thresholded of the image. (This thresholding could best be done under the ImageAction program, but was also written as a FORTRAN program for consistency in presenting a unified user interface.)

2. identified the features in the thresholded image and assigned each feature a unique gray-scale. (This step would limit the maximum number of features in a frame to 256 due to the 8-bit frame memory, but a special group count register was incorporated to allow the feature count to exceed this value. This removed another limitation that was present in Redman's work.)

3. sized the identified features and wrote a data table. This program finds the x-chord, the y-chord, and the area of the feature. It incorporated a conversion from pixels into micrometers by requesting the value of the microscope objective magnification that was used to record the reconstructed hologram images. The measured data was written into a data table for subsequent postprocessing.

Orguc used the Statgraphics statistics program to process the data. This program is a (relatively) user-friendly multi-function statistics program that accepts data in a simple table format and allows the calculation and plotting of the histograms with simple menu choices. The program also allows more sophisticated processing of the data, if desired.

Orguc was able to process full 512x512 images rather than the one-quarter images that Redman was limited to. The processing time required for a full-frame image was reduced from five hours for Redman to 10 minutes of time. Typical results of Orguc's work are shown in Figure 4.3 and Figure 4.4.

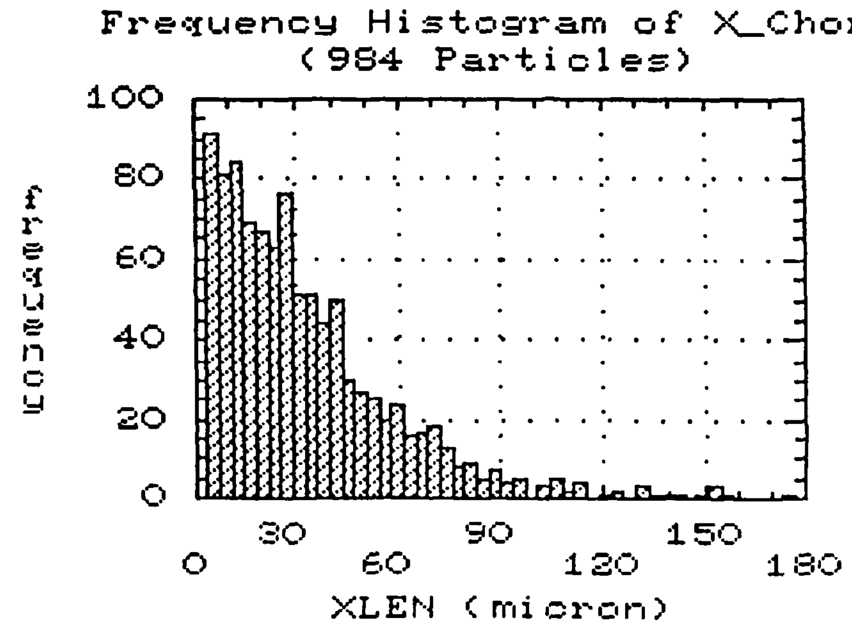


Figure 4.3: Measured histograms of x-chord length from 984 particles contained in 10 frames.

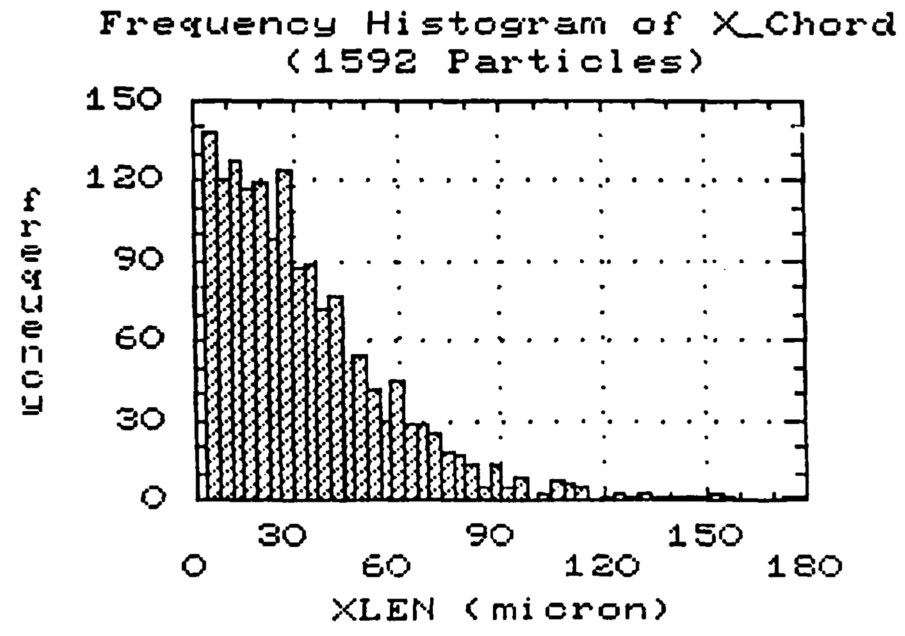


Figure 4.4: Measured histograms of x-chord length from 1,592 particles contained in 16 frames.

4.1.3 Final FORTRAN program

Kaeser [10] used an optimizing FORTRAN compiler (Microsoft FORTRAN Compiler, Ver. 4.1) to improve the performance of the program written by Orguc. Several subtle changes to the code were required to make the type casting (e.g., INTEGER, CHARACTER) agree between the main program and the ITEX/PC subroutines. The earlier compiler was not fussy about differences in type casting between the main program and the subroutines (e.g., a CHARACTER*21 variable in the main program could work with a CHARACTER*22 variable in a subroutine). The later compiler would not accept this difference, as it is incompatible with the FORTRAN 77 ANSI standard that Ver. 4.1 adhered to. Also the order of the array declarations had to be reversed to follow a change in syntax (e.g., a integer*2 jarray(9) [REFERENCE,NEAR] declaration became integer*2 jarray[REFERENCE,NEAR] (9)).

The improvements in the program included a reduction in the program size and the ability to run the program in a DOS 4.0 environment. (Previous versions of the executable program would not run in the presence of any other drivers loaded in the configuration program or in the presence of any Terminate–and–Stay–Resident (TSR) programs. Two of the programs are still large enough to require that the TSR programs be remove (or loaded into high memory), but would operate in the presence of a standard set of drivers needed to control the Bernoulli storage, the multi–partitioned hard disk in the computer, and the 386 add–in board.)

The striking reduction in the executable program size, when optimized, is noted by comparing the second ("Original") and third ("MSFORT") columns of Table 4.1.

The optimized FORTRAN program execution times are shown in Table 4.2. No significant improvement in execution time was observed due to the optimizing FORTRAN compiler.

4.2 C programs

4.2.1 Original C program

Kaeser [10] initiated the conversion of the FORTRAN programs into the C programming language. The C language offers several advantages for our application. The language offers dynamic memory allocation during runtime, eliminating the need to preallocate space for arrays. In addition, the availability of a virtual array capability that allowed the use of hard–disk

Filename	Original	MSFORT	MSC
THRESH.EXE	131,882	44,795	31,467
SPECKLE.EXE	122,180	33,569	—
NEWSIZE.EXE	556,038	52,575	46,281
NEWID.EXE	569,830	46.655	32,509
SIGMA.EXE	379,704	48,293	49,483
STAT.EXE	544,626	44,143	49,475
GEOFIL.EXE	364,382	41,375	49,969

Table 4.1: Program sizes (in kilobytes) for particle sizing programs and speckle–reduction filter programs in original form, after use of optimizing FORTRAN compiler (MSFORT), and after conversion to C language (MSC).

Program	MSFORT	MSC
Feature ID	1 m 11 s	1 m 0 s
Feature sizing	1 m 15 s	1 m 35 s
2sigma filter	7 m 20 s	6 m 33 s
Local stat filter	39 m 45 s	13 m 0 s
Geometric filter	18 m 50 s	6 m 17 s

Table 4.2: Program execution times for particle sizing programs and speckle–reduction programs after use of optimizing FORTRAN compiler (MSFORT) and after conversion to C language (MSC). Programs work with only one–fourth of an image frame to allow comparison with Redman;s results.

space for array storage and manipulation freed the computer memory of the storage requirements for large arrays (at some sacrifice in processing speed due to the read/write overhead). The program structures were also improved by the elimination of numerous GO TO statements that had appeared in the FORTRAN programs.

The programs were again arranged in a modular fashion, as the combined program would have exceeded the computer's memory capability. The modules were:

1. a header program that contained files and parameters that were common to all modules and the INCLUDE statements that were necessary to ensure compatibility with the libraries used. In this way redefinition of constants was easily accomplished in one location without having to work with multiple files.

2. a general function program that contained the functions used throughout the modules (again to add portability and to ease redefinition of a function),

3. a threshold program to threshold the screen image (and allow the user to change it) until the desired thresholded image is achieved,

4. a program to measure the speckle index of an image for the purpose of quantifying the speckle level,

5. a virtual array program that contained the virtual array functions as in Ref. [21] (with minor modifications),

6. a program to identify the features (based on the programs of Redman and Orguc),

7. a program to size the features identified in the prior program (based on the algorithms developed by Redman and Orguc) and to write a data table suitable for use with the Statgraphics analysis program, and

8. programs to implement the speckle reduction filters studied by Edwards, based on his FORTRAN programs.

The programs were verified by reproducing the results that had been made in Refs. [7] and Orguc:87.

The reduction in the executable program size is noted by comparing the third ("MSFORT") and fourth ("MSC") columns of Table 4.1 on page 66.

The sizing programs were slightly smaller than the optimized FORTRAN programs. The speckle–filter programs were larger than the optimized FORTRAN programs due to the added speckle index calculation code and the added virtual array code that was incorporated into the C–language version of these programs.

The program execution times for the C–language programs are shown in the last column of Table 4.2 on page 66. Modest improvements in execution time were observed for most programs with significant improvements occurring for the local statistics speckle–reduction filter and the geometric speckle–reduction filter programs. The improvements in program logic and execution speed overcame the additional overhead of the virtual array operations.

4.2.2 Modified C program

Hockgraver [11] followed up Kaeser's work seeking to improve the C program. She implemented a new image processing board (a PCVISION*plus* board from Imaging Technology) with a new software library, ITEX/PC*plus*. The new board has sufficient memory to store two frames of an image (although this feature was not used), implements a memory architecture that makes the board more compatible with other commercially–available image processing boards, and contains more image processing capability than the prior–generation board.

The modular arrangement of the programs is shown in Figure 4.5.

- The support programs speckle.c, vir_arry.c, and genfunc.c are shown to side of the main flow.

- The images are processed with one of the speckle–reduction filters (if desired) and passed to the thresholding program.

- The thresholded image is passed to the feature identification module which uses the video memory to record the features that are found by the algorithm.

- The feature image is passed to the sizing module for feature sizing and data table generation.

- The data table is passed to the commercially–available Statgraphics software for data analysis.

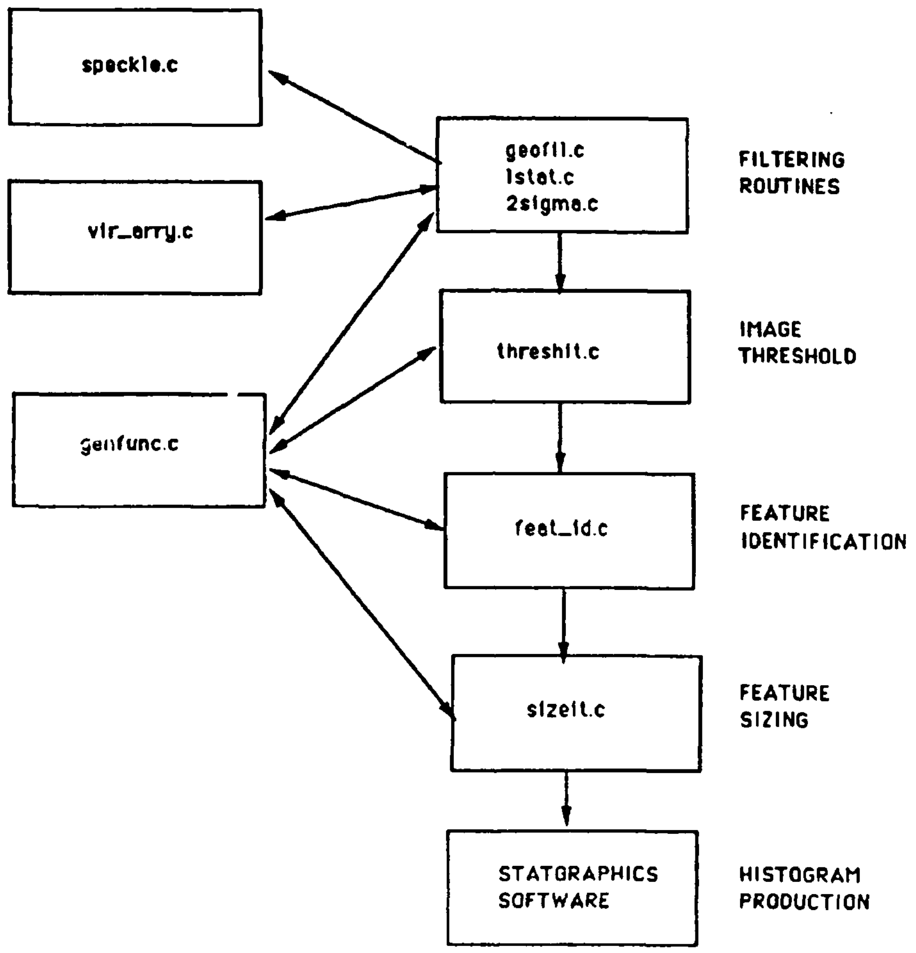


Figure 4.5: Modular program structure of C programs.

At the completion of each module the user is asked whether the program should save the output to disk (for safekeeping). After answering the prompt, the user initiates the next step in the process from the control menu.

The following summarizes the changes that were made at stage of the study:

- All of the calls to the subroutines in the ITEX/PC*plus* library were modified to follow the revised syntax of the new package.

- Programming improvements included more use of the general function program module that Kaeser had included, but underutilized. In addition, all remaining GO TO statements were eliminated by a rewriting of the code, allowing agreement with structured programming concepts and increased portability.

- An over-arching menu program was incorporated to provide a user-friendly interface that allows the user to run the modules of the routines in sequence by selection from a menu, rather than with command-line inputs.

- Much of Kaeser's C-language code was revised to improve the modularity of the programs.

- A subroutine was added in the local statistics filter program and the sigma filter program to compute the standard deviation of the gray levels in the image. In previous versions, this value was computed by the ImageAction software and was supplied by the user as an input at the beginning of the local statistics filter program. The new version of ImageAction*plus* does not compute this value, so a routine was needed to perform the calculation.

- Another change to all of the programs was required by a change in the handling of the file name and the comment associated with an image that is saved to disk. In the new software, pointers were used to indicate these quantities, rather reserved arrays. The code was changed accordingly.

The executable program size is shown in Table 4.3. The programs have all increased in size (but are still much smaller than their FORTRAN counterparts). This increase in program is attributed to the increased complexity of the subroutine calls to the ITEX/PC*plus* subroutines over the calls to the

Filename	ITEX/PC	ITEX/PC*plus*
THRESHIT.EXE	31,467	63,101
FEAT_ID.EXE	32,509	64,407
SIZE_IT.EXE	46,281	65,429
2SIGMA.EXE	49,483	65,491
LSTAT.EXE	49,475	65,499
GEOFIL.EXE	49,969	66,317

Table 4.3: Program sizes (in kilobytes) for particle sizing programs and speckle–reduction filter programs in original C language form for use with ITEX/PC software package and after modification for use with ITEX/PC*plus* software package.

ITEX/PC subroutines and to the fact that the entire ITEX/PC*plus* library was linked to the programs rather than the individual subroutines as in Kaeser's programs. A comparison of the source code, shown in Table 4.4, reveals that the original code was greatly shortened and that the expansion occurred in the compiling and linking process.

The modified C–program execution times using a hologram image are shown in Table 4.5. The processing times were significantly longer than the previous C program processing times noted in Table 4.5 on page 72. Most of this increase in execution time is attributed to the fact that the modified programs treat the full frame image rather than the one–fourth frame that was the basis of the timing measurements made by Kaeser. (He used the one–fourth frame to compare his results against Redman's results.) Due to the new frame storage structure used in ITEX/PC*plus*, timing measurements could not be done on one–fourth of a frame, so direct comparison of the timing results cannot be done. It should also be noted that the processing times for a full frame should increase more than four times the processing time of the quarter–frame, due to the nesting of the DO lops in the programs. These processing times for the full frame are still too long and work is currently proceeding to reduce them by trying to further improve the code efficiency.

Filename	Original C	Modified C
THRESHIT.C	5,807	1,336
FEAT_ID.C	10,130	6,003
SIZE_IT.C	11,011	7,426
2SIGMA.EXE	14,623	5,396
LSTAT.EXE	14,434	5,396
GEOFIL.EXE	14,888	7,916

Table 4.4: Source program sizes (in kilobytes) for particle sizing programs and speckle-reduction filter programs in original C language form for use with ITEX/PC software package and after modification for use with ITEX/PC*plus* software package.

Program	Modified C
Feature ID	13 min 30 s
Feature sizing	29 m 40 s
2sigma filter	12 m 41 s
Local stat filter	22 m 4 s
Geometric filter	56 m 43 s

Table 4.5: Program execution times for particle sizing programs and speckle-reduction programs after use of optimizing FORTRAN compiler (MSFORT) and after conversion to C language (MSC). Programs work with a full image frame.

Chapter 5

CONCLUSIONS

5.1 Summary

This work has shown that computer image processing can be used to locate and size particles in a hologram reconstruction.

Algorithms were developed for the Imaging Technology, Inc., line of image processing boards made for installation in AT–compatible computers. These boards and their associated software offer a cost–effective way to digitize, store, and process the images. Images of test objects recorded in white illumination and as holograms were successfully processed and served to validate the algorithms. Data from reconstructions of holograms made of a test motor firing were also successfully processed. Comparison with other measurement techniques is required to validate the measured results from the test firings.

Speckle–reduction was required due to the pronounced speckle that overlay our reconstructed images. Removal of the speckle comes at the cost of reduced resolution capability and increased processing time. The speckle index can be used to quantitatively compare the amount of speckle in an image. The resolution was best measured from holograms and images of the Air Force standard resolution chart. The best speckle–reduction technique for our images was to record the images from a moving diffuse screen. The integration of the image by the TV tube reduced the speckle considerably. Of the nonlinear filters used to reduce speckle in synthetic aperture radar images, the geometric filter proved to be the best combination of speckle reduction, resolution maintenance, and processing time. Techniques for digitally averaging the images were successful but required considerable computer storage and processing time to produce an image that was inferior to the moving-screen images in terms of the amount of speckle present. The nonlinear filters and image-averaging techniques might prove useful, however, in reducing the residual speckle found in the moving-screen images.

The original code that processed only one-fourth of an image in four hours has been optimized and sped up to the point where we can process an entire image (without speckle-reduction filtering) in approximately ten minutes. Further increases in speed can be accomplished with faster hardware and more the development of better sizing algorithms.

5.2 Acknowledgements

The author is pleased to acknowledge the contributions of Professor David Netzer of the Aeronautics and Astronautics Department of the Naval Postgraduate School and his students in providing the holograms of the various test objects and the rocket motor firing. The support of the Air Force Astronautical Laboratory is also gratefully acknowledged.

Bibliography

[1] T.D. Edwards, K.G. Horton, D.N. Redman, J.S. Rosa, J.B. Rubin, S.C. Yoon, J.P. Powers, and D.W. Netzer, "Measurements of Particulates in Solid Propellant Rocket Motors," *Proceedings of the 21st JANNAF Combustion Meeting*, Chemical Propulsion Information Agency, Johns Hopkins University, Laurel, Maryland, pp. 1–14, 1987.

[2] T.D. Edwards, R.K. Harris, K.G. Horton, M.G. Keith, A. Kertadidjaja, Y.S. Lee, D.N. Redman, J.S. Rosa, J.B. Rubin, S.C. Yoon, J.P. Powers, and D.W. Netzer, "Measurements of Particulates in Solid Propellant Rocket Motors (U)," Air Force Astronautics Laboratory Technical Report AFAL–TR–87–029, Edwards AFB, CA, October, 1987.

[3] Imaging Technology Incorporated, *The ITEX/PC Programmer's Manual*, Part Number 47–S00005–01, Revision 1.1, September 1985.

[4] CPT L.A. Klooster, *Image processing of solid propellant combustion holograms using a Quantimet 720*, MSEE Thesis, Naval Postgraduate School, Monterey, California, December 1983.

[5] LT M.P. Shook, *Computer-Controlled Image Analysis of Solid Propellant Combustion Holograms Using a Quantimet 720 and a PDP-11*, MSEE Thesis, Naval Postgraduate School, Monterey, California, September 1985.

[6] MAJ D.N. Redman, *Image Analysis of Solid Propellant Combustion Holograms Using an ImageAction Software Package*, MSEE Thesis, Naval Postgraduate School, Monterey, California, June 1986.

[7] CPT T.D. Edwards, *Implementation of three speckle reduction filters for solid propellant combustion holograms*, MSEE Thesis, Naval Postgraduate School, Monterey, CA, December 1986

[8] LT(jg) E.S. Orguc, *Automatic data retrieval from rocket motor holograms*, MSEE Thesis, Naval Postgraduate School, Monterey, CA, December 1987

[9] MAJ D.J.G. Carrier, *Automatic measurement of particles from holograms taken in the combustion chamber of a rocket motor*, MSEE Thesis, Naval Postgraduate School, Monterey, CA, December 1988

[10] LCDR D.S. Kaeser, *Code optimization of speckle reduction algorithms for image processing of rocket motor holograms,* MSEE Thesis, Naval Postgraduate School, Monterey, CA, December 1988

[11] LT V.R. Hockgraver, *Implementation of ImageActionplus software for image analysis of solid propellant combustion holograms,* MSEE Thesis, Naval Postgraduate School, Monterey, CA, September 1989

[12] LT Yeong-Lip Lee, *Title to be determined,* MSEE Thesis, Naval Postgraduate School, Monterey, CA, March 1991

[13] K.D. Ahlers and D.R. Alexander, "Microcomputer based digital image processing system developed to count and size laser-generated small particle images," *Optical Engineering,* Vol. 24, No. 6, pp. 1060-1065, 1985

[14] J.S. Lim and H. Nawab, "Techniques for speckle noise removal," *Optical Engineering,* Vol. 20, No. 3, pp. 472-480, 1981

[15] Netzer, D.W. and Powers, J.P., "Particle Sizing in Rocket Motor Studies Utilizing Hologram Image Processing," Presented at NASA Workshop on Data Reduction From Images and Interferograms, NASA/Ames Research Center, January 1985.

[16] W.J. Conover, *Practical Nonparametric Statistics,* John Wiley and Sons, 1971.

[17] J.W. Goodman, "A random walk through the field of speckle," *Optical Engineering,* Vol. 25, No. 5, pp. 610-612, 1986.

[18] Crimmins, T.R., "Geometric Filter for Speckle Reduction," *Applied Optics,* Vol. 24, No. 10, May 1985.

[19] Crimmins, T.R., "Geometric Filter for Reducing Speckle," *Optical Engineering,* Vol. 25, No. 5, May 1986.

[20] Lee, J.S., "Speckle Suppression and Analysis for Synthetic Aperture Radar," *Optical Engineering,* Vol. 25, No. 5, May 1986.

[21] Mark Tichenor, "Virtual arrays in C," *Dr. Dobb's Journal of Software Tools,* No. 138, May 1988.

DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Department Chairman, Code EC Naval Postgraduate School Monterey, CA 93943-5004	1
4.	Director of Research Administration, Code 81 Naval Postgraduate School Monterey, CA 93943-5000	1
5.	Professor John Powers, Code EC/Po Naval Postgraduate School Monterey, CA 93943-5002	10
6.	Professor David Netzer, Code AA/Nt Naval Postgraduate School Monterey, CA 93943-5002	2
7.	Dr. Michael Holmes, AL/LSNE Air Force Astronautics Laboratory Edwards AFB, CA 93523-5000	4