Reconstruction of computer simulated, atmospheric turbulence-degraded, astronomical objects by application of the Knox-Thompson and triple-correlation phase recovery techniques

Lackemacher, James M.

Monterey, California: Naval Postgraduate School
THESIS

RECONSTRUCTION OF COMPUTER SIMULATED, ATMOSPHERIC TURBULENCE-DEGRADED, ASTRONOMICAL OBJECTS BY APPLICATION OF THE KNOX-THOMPSON AND TRIPLE-CORRELATION PHASE RECOVERY TECHNIQUES

by

James M. Lackemacher

December, 1990

Thesis Advisor: Donald L. Walters
Thesis Co-Advisor: Charles L. Matson
Thesis Co-Advisor: Michael C. Roggemann

Approved for public release; distribution is unlimited
Atmospheric turbulence severely degrades images of astronomical objects. Providing images that accurately reflect the true nature of these objects is essential to their understanding. Several object recovery techniques exist within the field of speckle imaging that produce accurate representations of astronomical objects. This thesis provides an in-depth comparison of two such techniques, Knox-Thompson and triple-correlation.

Through computer simulation, this thesis accurately compares the abilities of both recovery techniques to enhance turbulence degraded objects by exploiting the diffraction-limited information contained in short exposure, or "speckle", images. The simulation produced these images by creating an object and several phase screens which simulated the effects of turbulence. Together, the object and the appropriate quantity of phase screens yielded the required short exposure images. Application of the Knox-Thompson and triple-correlation techniques to identical sets of these degraded images produced the resulting reconstructed objects, their signal-to-noise ratios and their azimuthal RMS phase errors. Comparison of these three factors over several imaging criteria concluded that the superior phase recovery technique was triple-correlation.
Approved for public release; distribution is unlimited.

Reconstruction of Computer Simulated, Atmospheric Turbulence-
Degraded, Astronomical Objects by Application of the Knox-
Thompson and Triple-Correlation Phase Recovery Techniques

by

James M. Lackemacher
Lieutenant, United States Navy
B.A., University of California, Berkeley

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN PHYSICS

from the

NAVAL POSTGRADUATE SCHOOL
December 1990

Author:  

James M. Lackemacher

Approved by:  

Donald L. Walters, Thesis Advisor  
Charles L. Matson, Thesis Co-Advisor

Michael C. Roggeaman, Thesis Co-Advisor  
David S. Davis, Second Reader

Approved by:  

Karlheinz E. Woehler, Chairman
Department of Physics
ABSTRACT

Atmospheric turbulence severely degrades images of astronomical objects. Providing images that accurately reflect the true nature of these objects is essential to their understanding. Several object recovery techniques exist within the field of speckle imaging that produce accurate representations of astronomical objects. This thesis provides an in-depth comparison of two such techniques, Knox-Thompson and triple-correlation.

Through computer simulation, this thesis accurately compares the abilities of both recovery techniques to enhance turbulence degraded objects by exploiting the diffraction-limited information contained in short exposure, or "speckle", images. The simulation produced these images by creating an object and several phase screens which simulated the effects of turbulence. Together, the object and the appropriate quantity of phase screens yielded the required short exposure images. Application of the Knox-Thompson and triple-correlation techniques to identical sets of these degraded images produced the resulting reconstructed objects, their signal-to-noise ratios and their azimuthal RMS phase errors. Comparison of these three factors over several imaging criteria concluded that the superior phase recovery technique was triple-correlation.
# TABLE OF CONTENTS

## I. INTRODUCTION ................................................................. 1

## II. BACKGROUND ................................................................... 3
   A. RESOLUTION .................................................................. 3
   B. TURBULENCE EFFECTS .................................................. 4
   C. HISTORICAL PERSPECTIVE ............................................ 4

## III. THEORY ........................................................................ 7
   A. TURBULENCE MODEL .................................................... 7
   B. PHASE SCREEN PRODUCTION ......................................... 9
      1. Fast-Fourier-Transform Method .................................. 9
      2. Karhunen-Loeve Expansion Method .......................... 10
   C. IMAGE RECOVERY ....................................................... 13
      1. Power Spectrum Recovery ......................................... 13
      2. Phase Spectrum Recovery ......................................... 14
         a. Technique Analysis .............................................. 14
         b. Phasor Spectrum Recovery ..................................... 16
c. Photon Noise effects ................................ 18

d. Phasor Spectrum Weighting .......................... 20

e. Recovery Process ................................... 21

D. PHASOR RECOVERY TECHNIQUE COMPARISON .... 21

E. SUMMARY ........................................ 23

IV. COMPUTER SIMULATION ................................ 24

A. COMPUTER REQUIREMENTS .......................... 24

B. SIMULATION PROCESS ............................... 25

1. Object Production .................................... 26

   a. Object Creation .................................. 26

   b. Object Scaling .................................. 28

   c. Fourier Spectrum Determination ................... 30

   d. Fourier Spectrum Normalization .................... 30

2. Turbulence Phase Screen Production .................. 31

   a. Gaussian Random Number Array .................... 31

   b. Filter Function .................................. 31

   c. Karhunen-Loeve Functions ........................ 32

   d. Incoherent Transfer Function ....................... 33

3. Object Degradation ................................... 33

4. Centroiding ......................................... 34

5. Image Recovery ...................................... 34
a. Power Spectrum Recovery .................................. 35
b. Phasor Spectrum Recovery ................................. 35
6. Azimuthal Signal-To-Noise Ratio ......................... 37
7. Fourier Spectrum Filtering ................................. 37
8. Azimuthal RMS Phase Error ............................... 38
C. SUMMARY ................................................. 38

V. SIMULATION RESULTS ..................................... 39
   A. RECOVERY TECHNIQUE COMPARISON CRITERIA ...... 39
   B. RECOVERY TECHNIQUE COMPARISONS ............... 40
      1. Short Exposure Image Quantity ....................... 41
      2. Photon Count ......................................... 41
      3. Turbulence Magnitude and Offset Value ............. 42
      4. Tilt .................................................. 43
      5. Centroiding .......................................... 44
      6. Point Objects ........................................ 45
      7. Recovery Technique Comparison Findings .......... 45

VI. CONCLUSION .............................................. 47
    A. OVERVIEW ............................................. 47
    B. OPTIMUM IMAGE RECOVERY APPROACH ............... 47
    C. FURTHER STUDY ....................................... 48

vi
LIST OF REFERENCES ........................................... 49

APPENDIX A. PLOTS AND GRAPHS .............................. 51

APPENDIX B. KNOX-THOMPSON MAIN PROGRAM .......... 107

APPENDIX C. TRIPLE-CORRELATION MAIN PROGRAM ..... 112

APPENDIX D. UNIVERSAL IMAGE RECOVERY SUBROUTINES .... 118

APPENDIX E. SPECIFIC KNOX-THOMPSON SUBROUTINES .... 131

APPENDIX F. SPECIFIC TRIPLE-CORRELATION SUBROUTINES ... 136

APPENDIX G. ATMOSPHERIC DEGRADATION SUBROUTINES .... 141

APPENDIX H. PHASE ERROR AND LOW PASS FILTER PROGRAM . 164

INITIAL DISTRIBUTION LIST .................................. 179
ACKNOWLEDGEMENTS

I would like to extend my sincerest gratitude to all those who, through their assistance and support, made this work possible. To Captain Mike Roggemann, I would like to express my heartfelt thanks for arranging my experience tour to the Weapons Lab and providing me with the broad theoretical information base that was the foundation for this work. To Captain Chuck Matson, I would like to express my genuine appreciation for providing essential assistance in the area of speckle imaging with the uncompromising determination required to make this computer simulation operate. To Ms. Ida Drunzer of Rockwell Power Systems, I would like to extend many thanks for affording the essential computer help during my stay at the Weapons Lab. Their invaluable assistance granted me the knowledge and courage to complete this thesis. Finally, to my wife, Lynn-Ann and sons, Kent and Wesley, I would like to dedicate this thesis. They provided me endless support and encouragement, and sacrificed untold hours of my company to allow its completion. Thank you.
I. INTRODUCTION

Scientific research of astronomical imagery has been continual since the invention of the telescope at the beginning of the seventeenth century. Initially, the limiting factor on astronomical image resolution was the quality and size of the telescope optics. As technology progressed, larger telescopes improved to the point where atmospheric turbulence (hereafter, simply turbulence) became the limiting factor on image resolution.

Turbulence severely limits the resolution of long exposure images produced by ground-based telescope imaging systems. Removal of turbulence corruption to improve image resolution is an area of extensive research. Optimal telescope site location minimized the effect of turbulence, but did not produce the near diffraction-limited images desired. Removal of turbulence distortion occurred through the development of statistical methods, called speckle imaging techniques, that produced near diffraction-limited resolution. The basis of these techniques was the assumption that turbulence remains essentially stationary during a short exposure image of the desired object. Though distorted, these short exposure images retain diffraction-limited information.

Determination of the object's power and phase spectra are separate operations. Although the power spectrum can be directly averaged while retaining diffraction-limited information, the phase spectrum cannot. Instead, either the cross-spectrum
The Knox-Thompson method or the bispectrum (the triple-correlation method) are averaged because diffraction-limited information is retained. Then the phase spectrum is recovered from either the cross-spectrum or the bispectrum. The combination of the power and phase spectra generates the object's Fourier spectrum which, when inverse Fourier transformed, provides the recovered image.

Several factors affect the quality of image reconstruction. These factors influence both phase recovery techniques and include the amount of turbulence, the size of the telescope, and the light level of the object. The randomness of turbulence and the effect of random photon noise, make exact image reconstruction impossible, however, increasing the number of short-exposure images in the averaging process recovers a better image. This thesis compares the Knox-Thompson and triple-correlation phase recovery techniques under several different imaging conditions to determine their ability to improve the signal-to-noise ratio (SNR) of a degraded object.
II. BACKGROUND

A. RESOLUTION

In the image reconstruction process, resolution determines image quality. Resolution is defined as:

the process or capability of making distinguishable the individual parts of an object, closely adjacent optical images, or sources of light [Ref. 1].

From a Fourier optics perspective, resolution is proportional to the high spatial frequency content of the imaging system. The standard for resolution is based on the image of two point objects (binary star) viewed through a telescope. The image consists of two overlapping Airy patterns with intensity

\[ I(\psi) = I(0) \cdot \left( \frac{2J_1(\psi)}{\psi} \right)^2, \tag{2.1} \]

where \( J_1(\psi) \) is the first order Bessel function and \( \psi \) is the normalized spatial frequency. The separation of the two first order fringes of the two Airy patterns constitutes the so-called Rayleigh limit of image resolution

\[ \Delta \theta = \frac{1.22 \cdot \lambda}{D}, \tag{2.2} \]

(in radians) for a circular aperture where \( \lambda \) is the wavelength of light and \( D \) is the telescope diameter [Ref. 2]. Telescope imperfections and turbulence prevent attainment of the theoretical limit for large apertures. Speckle imaging techniques
remove distortions due to these factors and produce images with resolution near the theoretical limit.

B. TURBULENCE EFFECTS

Turbulence produces temporal and spatial variations in atmospheric density, temperature, and index of refraction. The turbulence in the atmosphere perturbs an image, such as an Airy pattern from a point object (star), producing an extended image referred to as a seeing disk. The perturbation randomizes the electromagnetic phase front of the object. This randomization produces angular spreading, image wandering about its centroid, and scintillation or "twinkling". Turbulence effectively reduces the telescope's resolving power by randomly attenuating the high spatial frequencies.

C. HISTORICAL PERSPECTIVE

Until roughly 1970, attempts at solving the turbulence distortion problem were limited to finding the ideal telescope site. Generally, the sites were high in elevation and at locations regarded as having long periods of atmospheric stability. Even with great care in site selection, the typical angular resolution obtained was approximately one arcsecond, the maximum resolution attainable with a 12 centimeter telescope. Although phase distortions from turbulence constrained resolution, construction of large diameter telescopes provided enhanced light gathering capability.

In 1970, a technique developed by Labeyrie enabled recovery of near diffraction-limited image Fourier moduli [Ref. 3]. The concept that a long
exposure image was blurred by turbulence fluctuations due to phase spectrum blurring and not power spectrum blurring, provided the basis of Labeyrie's technique. Labeyrie determined that a short exposure image, about 10 milliseconds, would freeze the disturbance yet still contain near diffraction-limited information of the object. Taking many short exposure images, calculating their power spectra, then averaging, enabled a diffraction-limited estimate of the object's power spectrum to be made. Labeyrie's technique allowed high resolution measurements of binary star separations. However, lack of object phase information prevented faithful image reconstruction.

In 1974 Knox and Thompson developed a technique for retrieving the object's phase spectrum [Ref. 4]. The method uses the Knox-Thompson (KT) algorithm to provide an estimate of the object’s phase spectrum using the same short exposure images required for the estimate of the object’s power spectrum. The KT method calculates the average cross-spectrum of the object in Fourier space to determine the object's phase spectrum. Calculation of the cross-spectrum involves determining the average correlation of spatial frequency pairs displaced from each other by a small frequency differential. The average provides a statistical phase difference approximation from which the object's phase spectrum is obtained.

In 1983 Lohmann, Weigelt, and Wirnitzer developed another technique for retrieving the object's phase spectrum [Ref. 5]. This method, referred to as triple-correlation (TC), also uses short exposure images to estimate the object’s phase spectrum. The TC method calculates the average bispectrum of the object in Fourier space to determine the object's phase spectrum. Calculation of the bispectrum
involves determining the average of a third order correlation that consists of a frequency point, a point shifted by an offset, and a difference term. As with the KT technique, the average provides a statistical phase difference approximation from which the object's phase spectrum is obtained.

Either the KT or the TC technique determines the object's phase spectrum which is necessary to produce accurate recovered images. Combining the reconstructed power spectrum and phase spectrum produces the reconstructed Fourier spectrum, which when Fourier transformed, yields the recovered image.
III. THEORY

A. TURBULENCE MODEL

Both the KT and the TC techniques utilize short exposure images to recover the object's phase spectrum. A method developed by Tyler and Fried of the Optical Sciences Company, simulates turbulence to produce the short exposure images required to test these recovery techniques [Ref. 6]. This method requires the following three assumptions: [Ref. 7]

1. Turbulence is represented by a single phase screen in the pupil plane of the telescope.

2. Turbulence is isoplanatic, that is, the distortion from turbulence and the imaging system is considered shift invariant over the entire image plane.

3. The images are quasi-monochromatic.

With these assumptions, a single short exposure image becomes a convolution in image space

\[ i(\vec{x}) = o(\vec{x}) * s(\vec{x}) , \]

(3.1)

where \( i(\vec{x}) \) is the short exposure image intensity, \( o(\vec{x}) \) the object intensity, and \( s(\vec{x}) \) is the instantaneous point spread function. The vector \( \vec{x} \) represents the two-dimensional orthogonal spatial coordinates \( x \) and \( y \). Using the convolution theorem, equation (3.1), becomes a product in Fourier space
\[ I(\vec{u}) = O(\vec{u}) \cdot S(\vec{u}) \quad , \] (3.2)

where \( I(\vec{u}) \) is the Fourier transform of the short exposure image intensity, \( O(\vec{u}) \) is the Fourier transform of the object intensity, and \( S(\vec{u}) \) is the instantaneous incoherent transfer function. The vector \( \vec{u} \) represents the two-dimensional orthogonal spatial frequency coordinates \( u \) and \( v \).

The point spread function, \( S(\vec{r}) \), and thereby the incoherent transfer function, \( S(\vec{u}) \), represent distortions from both turbulence and the imaging system. The assumption of stationary turbulence is accurate for short exposure images. Consequently, an instantaneous distribution of random phases (phase screen) approximates the instantaneous distortion of an image by turbulence. An array of random numbers filtered by a power spectral density function and corrected for low spatial frequency under-representation can simulate this phase screen [Ref. 6]. Therefore,

\[ H(\vec{u}) = P(\lambda F \vec{u}) e^{i \Theta(\lambda F \vec{u})} \quad , \] (3.3)

represents the instantaneous coherent transfer function, where \( P(\lambda F \vec{u}) \) is the transfer function of the telescope, \( e^{i \Theta(\lambda F \vec{u})} \) is the instantaneous turbulence phase screen, \( F \) is the focal length of the telescope, and \( \lambda \) is the wavelength of light [Ref. 7]. Finally, the auto-correlation of the coherent transfer function, \( H(\vec{u}) \),

\[ S(\vec{u}) = H(\vec{u}) \ast H(\vec{u}) \quad . \] (3.4)

yields the instantaneous incoherent transfer function required for equation (3.2).
B. PHASE SCREEN PRODUCTION

Three common techniques for producing turbulence phase screens exist. One technique, the Fast-Fourier-transform (FFT) method, creates an array of filtered white noise and inverse Fourier transforms the array to real space providing the phase screen. A second technique, referred to as the Karhunen-Loeve (KL) expansion method, uses the KL expansion with a basis of Zernike polynomials to represent the phase screen. The third, hybrid, technique referred to as the Karhunen-Loeve-Fast-Fourier-Transform (KLFFT) method, combines the best properties of both techniques and manufactures phase screens which most accurately represent turbulence distortions.

1. Fast-Fourier-Transform Method

The FFT method provides a rapid means of generating a phase screen. Initially, creation of a square array of Gaussian-distributed random numbers of unity variance provides a representation of the phase over the entire aperture of the imaging system being evaluated. The array amplitudes are filtered in Fourier space radially outward from the origin by the square root of the Kolmogorov power spectral density function

\[ F_n(\bar{u}) = 0.1517 r_0^{-5/6} |\bar{u}|^{-11/6} , \]  

where \( |\bar{u}| \) is the radial distance from the origin in frequency units, and \( r_0 \) is the coherence diameter [Ref 7]. The origin is set to zero removing the constant (DC) term from the phase screen before applying the inverse Fourier transform. Two
phase screens result since complex numbers in the array consist of real and imaginary parts which are entirely distinct and statistically independent.

Though the FFT method generates phase screens rapidly, it has deficiencies. The FFT uses a finite number of discrete points, and consequently, high and low spatial frequency cutoff occurs. High spatial frequency cutoff is minor since most of the wave front error induced by turbulence is of low spatial frequency. Low spatial frequency cutoff is more serious as it induces under-representation of low spatial frequencies producing wave front tilt, or centroid position errors. Therefore, the associated structure function of the FFT-produced phase screen does not completely represent the $5/3$ power law turbulence structure function.

2. Karhunen-Loeve Expansion Method

The KL expansion method provides an accurate method of generating a phase screen. Random phases associated with turbulence can be expanded in terms of a series of orthogonal functions $\xi_k(t)$,

$$x(t) = \sum_j r_j \xi_k(t).$$  

(3.6)

The expansion coefficients $r_j$, are uncorrelated Gaussian random variables which represent turbulence statistics. The orthogonal functions provide the proper spatial dependence, thereby allowing the random phase to be evaluated anywhere within the aperture. The above expansion is referred to as the KL expansion. For a finite value of $j$, the KL expansion is the optimum basis whose eigenvalues represent the energy.
content of the expansion coefficients, \( r_k(t) \), and the total energy is the sum of these eigenvalues. [Ref. 8].

Zernike expansion coefficients, for a random phase screen, are Gaussian random variables. Unfortunately, these expansion coefficients are correlated and cannot be used as a KL basis set directly. However, the Zernike covariance matrix (the matrix of expansion coefficients) is useful in determining the KL expansion for turbulence.

Three properties justify the use of Zernike polynomials as a basis set for the KL expansion to determine wave front turbulence. Use of the Zernike covariance matrix provides the necessary random variables required for the phase screen. Additionally, each eigenvector of the Zernike covariance matrix is the representation of the KL function in terms of the Zernike polynomials. Further, each eigenvalue of the Zernike covariance matrix is the variance associated with the corresponding KL expansion coefficient. Wave front error induced by turbulence is an outcome of the these properties. [Ref. 6]

The eigenvectors and the corresponding eigenvalues of the normalized Zernike covariance matrix are found which obey the relation

\[
C e_i = \lambda_i e_i ,
\]

(3.7)

where \( C \) is the covariance matrix, \( e_i \) is the \( i \)th eigenvector and \( \lambda_i \) is the corresponding normalized eigenvalue. The eigenvectors are normalized so each
element of the eigenvector indicates the amount of the corresponding Zernike polynomial that is contained in the $i^{th}$ KL function expressed as

$$K_i(\rho) = \sum_p e_{ip} Z_p(\rho),$$

(3.8)

where $K_i(\rho)$ is the $i^{th}$ KL function, $e_{ip}$ is the $p^{th}$ component of the $i^{th}$ eigenvector, and $Z_p(\rho)$ is the $p^{th}$ Zernike polynomial. Hence, the random wave front error produced by turbulence, $\phi(\vec{r})$, is

$$\phi(\vec{r}) = \sum_i \gamma_i K_i \left( \frac{\vec{r}}{D/2} \right),$$

(3.9)

where $\gamma_i$ is a set of Gaussian-distributed random numbers, $\vec{r}$ is the distance from the origin, and $D$ is the telescope diameter.

Though the KL expansion method is accurate, deficiencies exist, and care must be taken in its use. Calculating wave front distortion using the KL method requires an enormously large number of Zernike polynomials to achieve enough accuracy. The required number is proportional to $(D/x_0)^2$. Additionally, numerical inaccuracies exist in evaluating Zernike polynomials of high radial order. Therefore, to achieve the accuracy desired, avoidance of Zernike polynomials of high radial order is necessary.


The KLFFT method combines the fast computational speed of the FFT method with the optimum low spatial frequency representation of the KL functions. This technique requires a phase screen to be developed by the FFT method. The
first five KL functions are produced and applied to this phase screen to compensate for the under-representation of low spatial frequencies. With this compensation, this combined technique closely represents atmospheric turbulence.

The KLFFT method is quite powerful. Since only five KL functions are produced and applied, the total number of operations per phase screen is approximately double that of the FFT method alone. Therefore, this phase screen production technique is relatively fast. [Ref. 6]

C. IMAGE RECOVERY

The image recovery techniques represent methods for providing the reconstructed image from a turbulence-distorted object by utilizing several short exposure images of the object and a nearby star. The Labeyrie technique recovers the object’s power spectrum [Ref. 3]. The power spectrum provides the modulus of the Fourier transform of the object, the first part of the complex quantity required. Both the KT and TC techniques recover the object’s phase spectrum. Inverse Fourier transforming the product of the modulus and the phase provides a reconstructed image of the original object.

1. Power Spectrum Recovery

Equation (3.2) represents the image intensity for a single short exposure image in Fourier space. The time average power spectrum of several short exposure images in Fourier space is
\[ \langle |I(\vec{u})|^2 \rangle = |O(\vec{u})|^2 \cdot \langle |S(\vec{u})|^2 \rangle . \]  

(3.10)

where the term on the left is the time average power spectrum of the image, the first term on the right is the object's power spectrum and the second term on the right is the incoherent transfer function. Several images of a star under similar imaging conditions as the object determine this transfer function, which is the time average of the instantaneous transfer functions. The object and the star need not be in the same isoplanatic patch as long as the second order statistics of the transfer function are the same for both sets of exposures [Ref. 9]. Solving for the object power spectrum, equation (3.10) becomes

\[ |O(\vec{u})|^2 = \frac{\langle |I(\vec{u})|^2 \rangle}{\langle |S(\vec{u})|^2 \rangle} . \]  

(3.11)

which recovers the object's Fourier modulus.

2. Phase Spectrum Recovery

Both the KT and TC phase recovery techniques recover the object's phase spectrum by using the cross-spectrum and bispectrum averaging processes respectively. To recover the object's phase spectrum, the KT technique calculates the average cross-spectrum while the TC technique calculates the average bispectrum.

a. Technique Analysis

The KT technique is the simpler of the two phase recovery methods. Determining the cross-spectrum is at the heart this algorithm. The cross-spectrum is defined as the product of two quantities in Fourier space: an array point and the
complex conjugate of an array point which is shifted, in frequency, from the original
array point by a small offset, $\Delta \bar{u}$. The cross-spectrum, $CS(\bar{u})$, is

$$CS(\bar{u}) = X(\bar{u}) \cdot X^* (\bar{u} + \Delta \bar{u}) ,$$  \hspace{1cm} (3.12)

where $X(\bar{u})$ is the array point and $X^* (\bar{u} + \Delta \bar{u})$ is the complex conjugate of the array
point shifted by the offset vector $\Delta \bar{u}$.

The TC technique is a more complicated form of phase recovery. The
bispectrum is defined as the product of three quantities in Fourier space: an array
point, the complex conjugate of an array point which is shifted, in frequency, from
the original array point by an offset, and an array point which is a function of the
offset only. The bispectrum, $BS(\bar{u})$, is

$$BS(\bar{u}) = X(\bar{u}) \cdot X^* (\bar{u} + \Delta \bar{u}) \cdot X(\Delta \bar{u}) ,$$  \hspace{1cm} (3.13)

where $X(\Delta \bar{u})$ is the array point which is a function of offset only and the other
terms are the same as defined for equation (3.12).

From equation (3.2), the average cross-spectrum is

$$<I(\bar{u}) \cdot I^* (\bar{u} + \Delta \bar{u})> =$$

$$[O(\bar{u}) \cdot O^* (\bar{u} + \Delta \bar{u})] \cdot <S(\bar{u}) \cdot S^* (\bar{u} + \Delta \bar{u})> .$$  \hspace{1cm} (3.14)

where the first term on the right is the object's cross-spectrum and the second term
on the right is the average incoherent transfer function cross-spectrum. The average
bispectrum is
where the first term on the right is the object's bispectrum and the second term on the right is the average incoherent transfer function bispectrum.

The average incoherent transfer function cross-spectrum and bispectrum contain distortions from both the turbulence and the imaging system. The average phases resulting from these calculations for turbulence are zero. Imperfections in the imaging system produce phases which are negligible for both the cross-spectrum [Ref. 10] and the bispectrum [Ref. 5] calculations. Therefore, the average cross-spectrum and bispectrum of the incoherent transfer function is assumed to be unity. Equations (3.14) and (3.15) reduce to simpler forms

\[ O(\bar{u}) \cdot O^* (\bar{u} + \Delta \bar{u}) = <I(\bar{u}) \cdot I^* (\bar{u} + \Delta \bar{u})> \quad (3.16) \]

and

\[ O(\bar{u}) \cdot O^* (\bar{u} + \Delta \bar{u}) \cdot O(\Delta \bar{u}) = <I(\bar{u}) \cdot I^* (\bar{u} + \Delta \bar{u}) \cdot I(\Delta \bar{u})> \quad (3.17) \]

b. Phasor Spectrum Recovery

Direct recovery of the object's phase spectrum is not possible since the cross-spectrum and bispectrum phases are only known modulo $2\pi$. The recursion algorithm fails when the cross-spectrum or bispectrum phase estimates are not equal to their principle arguments if multiple estimates of a single object phase spectrum
point are used. To avoid this problem without losing information, the reconstructed object's phasor spectrum is determined instead. The recovery process will henceforth be referred to as phasor reconstruction.

An arbitrary complex number $N$ in phasor notation is

$$N = |N| e^{i\phi},$$

(3.18)

where $|N|$ is the modulus of the complex number and $e^{i\phi}$ is the phasor in which $\phi$ represents the phase of the complex number. Solving for the phasor, equation (3.18) becomes

$$e^{i\phi} = N_{ph} = \frac{N}{|N|},$$

(3.19)

where the subscript $ph$ denotes phasor. Therefore, solving for the phasors of equations (3.16) and (3.17) by dividing each by their respective moduli gives

$$O_{ph}(\bar{u}) \cdot O_{ph}^*(\bar{u} + \Delta \bar{u}) = I_{ph}^{RT}(\bar{u}, \Delta \bar{u}),$$

(3.20)

and

$$O_{ph}(\bar{u}) \cdot O_{ph}^*(\bar{u} + \Delta \bar{u}) \cdot O_{ph}(\Delta \bar{u}) = I_{ph}^{TC}(\bar{u}, \Delta \bar{u}),$$

(3.21)

where

$$I_{ph}^{RT}(\bar{u}, \Delta \bar{u}) = \frac{\langle I(\bar{u}) \cdot I^*(\bar{u} + \Delta \bar{u}) \rangle}{|\langle I(\bar{u}) \cdot I^*(\bar{u} + \Delta \bar{u}) \rangle|},$$

(3.22)

and

$$I_{ph}^{TC}(\bar{u}, \Delta \bar{u}) = \frac{\langle I(\bar{u}) \cdot I^*(\bar{u} + \Delta \bar{u}) \cdot I(\Delta \bar{u}) \rangle}{|\langle I(\bar{u}) \cdot I^*(\bar{u} + \Delta \bar{u}) \cdot I(\Delta \bar{u}) \rangle|}.$$
The terms $I_{ph}^{RT}(\bar{u}, \Delta \bar{u})$ and $I_{ph}^{TC}(\bar{u}, \Delta \bar{u})$ are four-dimensional quantities, since several offset values may be used to determine estimates of the object's cross-spectrum and bispectrum. Solving for the offset-shifted object phasor and applying the complex conjugate operator to both sides, equations (3.20) and (3.21) become

$$O_{ph}(\bar{u} + \Delta \bar{u}) = \left( \frac{I_{ph}^{RT}(\bar{u}, \Delta \bar{u})}{O_{ph}(\bar{u})} \right)^*,$$

(3.24)

and

$$O_{ph}(\bar{u} + \Delta \bar{u}) = \left( \frac{I_{ph}^{TC}(\bar{u}, \Delta \bar{u})}{O_{ph}(\bar{u}) \cdot O_{ph}(\Delta \bar{u})} \right)^*.$$

(3.25)

Equations (3.24) and (3.25) provide the phasor spectrum necessary for image reconstruction. However, use of these equations is limited to infinite photon count, short exposure images which are unrealistic and useful for computer simulations only.

**c. Photon Noise effects**

Compensating for photon noise allows image reconstruction from low photon count images, though more of these images are required in the process to resolve the object. During the recovery process of low light-level objects, the introduction of photon bias occurs. This bias corresponds to the cross-spectrum or bispectrum averaging of photon events with themselves and contributes no useful information to the average. With the bias removed, equations (3.11) and (3.12) become

18
\[ CS(\vec{u}) = X(\vec{u}) \cdot X^*(\vec{u} + \Delta \vec{u}) - X^*(\Delta \vec{u}) \, , \]  
\[ (3.26) \]

and

\[ BS(\vec{u}) = X(\vec{u}) \cdot X^*(\vec{u} + \Delta \vec{u}) \cdot X(\Delta \vec{u}) + 2N_p - |X(\vec{u})|^2 + |X(\vec{u} + \Delta \vec{u})|^2 - |X(\Delta \vec{u})|^2 \, , \]  
\[ (3.27) \]

where \(-X^*(\Delta \vec{u})\) removes the cross-spectrum photon noise bias, \(-|X(\vec{u})|^2\), \(-|X(\vec{u} + \Delta \vec{u})|^2\), and \(-|X(\Delta \vec{u})|^2\) remove the bispectrum photon noise bias, and \(N_p\) is the photon count [Ref. 11]. Therefore, for realistic image reconstruction, equations (3.24) and (3.25) become

\[ O_{ph}(\vec{u} + \Delta \vec{u}) = \left( \frac{I^{TBIAS}_{ph}(\vec{u}, \Delta \vec{u})}{O_{ph}(\vec{u})} \right)^* \, , \]  
\[ (3.28) \]

and

\[ O_{ph}(\vec{u} + \Delta \vec{u}) = \left( \frac{I^{TBIAS}_{ph}(\vec{u}, \Delta \vec{u})}{O_{ph}(\vec{u}) \cdot O_{ph}(\Delta \vec{u})} \right)^* \, , \]  
\[ (3.29) \]

where

\[ I^{TBIAS}_{ph}(\vec{u}, \Delta \vec{u}) = \frac{\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta \vec{u}) - I^*(\Delta \vec{u}) \rangle}{|\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta \vec{u}) - I^*(\Delta \vec{u}) \rangle|} \, , \]  
\[ (3.30) \]

and

\[ I^{TBIAS}_{ph}(\vec{u}, \Delta \vec{u}) = \frac{\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta \vec{u}) - |I(\vec{u})|^2 - |I(\vec{u} + \Delta \vec{u})|^2 - |I(\Delta \vec{u})|^2 + 2N_p \rangle}{|\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta \vec{u}) - |I(\vec{u})|^2 - |I(\vec{u} + \Delta \vec{u})|^2 - |I(\Delta \vec{u})|^2 + 2N_p \rangle|} \, . \]  
\[ (3.31) \]
Equations (3.28) and (3.29), allow realistic image reconstruction.

d. Phasor Spectrum Weighting

Phasor spectrum weighting provides a means for enhancing desired phasor estimates, increasing reconstructed image quality. Both phasor recovery techniques benefit from phasor spectrum weighting. Weighting techniques suppress higher frequencies and enhance the reconstructed image's SNR. The method presented is the weighted least squares estimation approach. Matson showed that this method obtained the best results of four approaches analyzed [Ref. 12]. The method weights the object’s phasor spectrum with the SNR determined from the variance of the cross-spectrum or bispectrum as follows:

\[ \sigma^2(Re[SS(\bar{u})]) = \left| \frac{\sum_{i=1}^{n} \left( Re[SS_i(\bar{u})]^2 - \frac{Re[<SS(\bar{u})>]^2}{n} \right)}{n-1} \right| \]  

\[ \sigma^2(Im[SS(\bar{u})]) = \left| \frac{\sum_{i=1}^{n} \left( Im[SS_i(\bar{u})]^2 - \frac{Im[<SS(\bar{u})>]^2}{n} \right)}{n-1} \right| \]  

\[ \sigma = \sqrt{\sigma^2(Re[SS(\bar{u})]) + \sigma^2(Im[SS(\bar{u})])} \]  

\[ SNR = \frac{|<SS(\bar{u})>|}{\sigma} \cdot \sqrt{n} \]  

where \( SS(\bar{u}) \) represents either the cross-spectrum or the bispectrum.
e. **Recovery Process**

Determination of the object's phasor spectrum is afforded by the recursive method. An estimate of the image cross-spectrum or bispectrum for each short exposure image spectrum element provides the necessary information to determine the object's phasor spectrum. The estimates for each array element are determined recursively, from the origin radially outward to the diffraction limit of the imaging system. This method capitalizes on the inherent property of the phasor spectrum SNR which decreases radially as a function of increasing spatial frequency. These estimates are averaged over all the short exposure images and are weighted utilizing equation (3.35). This process results in the object's phasor array determined by equations (3.28) or (3.29). Once the reconstructed phasor array is found in this manner, it is multiplied by the square root of the power spectrum provided by equation (3.11). Inverse Fourier transforming this spectrum to image space provides the final reconstructed image.

D. **PHASOR RECOVERY TECHNIQUE COMPARISON**

There are several distinctions between the two phasor recovery techniques, aside from the obvious differences of equations (3.28) and (3.29). Since the recursion technique utilizes all the values from unity to the offset value in the averaging process, the greater the offset, the better the reconstructed image. For the KT technique, the modulus of the offset vector, \( |d\bar{u}| \), is restricted to be less than the seeing limit, \( r_o/\lambda \), whereas for the TC technique, the offset is unlimited. This
restriction allows the TC technique to surpass the KT technique in image quality by using a greater offset that includes more phase information. [Ref. 13]

Shift invariance becomes important for low photon count images. The KT technique is not shift invariant and requires shifting of the degraded short exposure images to the center of the image array by,

\[
\bar{x} = \frac{\sum_j x_j \cdot I_j(x)}{\sum_j I_j(x)},
\]  

and

\[
\bar{y} = \frac{\sum_j y_j \cdot I_j(y)}{\sum_j I_j(y)},
\]  

prior to the recovery process. Image centroiding minimizes linear phase ramping in Fourier space, however, centroiding accuracy decreases with lower light levels resulting from the randomness of the image. Consequently, since the KT technique is not shift invariant, it performs poorly for low photon count images because the centroiding accuracy is decreased. The TC technique is shift invariant, eliminating the centroiding requirement. Since the TC technique does not require centroiding, the inaccuracy of the centroiding process at low light levels does not enter into TC image reconstruction. [Ref. 11]
E. SUMMARY

The short exposure image is the key to effective image recovery. The high spatial frequencies within the short exposure image contain diffraction-limited information, though the object's phase spectrum is randomized. Recovery of the phasor spectrum provides the information required for true image reconstruction.

Three techniques for image reconstruction provide the recovered image. The Labeyrie technique recovered the object's power spectrum. Both the Knox-Thompson and the Triple-Correlation techniques recovered the object's phasor spectrum. Each phasor recovery technique yielded an independent array of phasors that, when combined with their corresponding power spectrum and inverse Fourier transformed, produced reconstructed images for comparison. Simulation of turbulence-degraded short exposure images by the Karhunen-Loeve-Fast-Fourier-Transform method allowed computer simulated comparison.
IV. COMPUTER SIMULATION

A. COMPUTER REQUIREMENTS

Turbulence simulation and image recovery processing, by their nature, require enormous amounts of calculations. The simulation presented in this thesis creates an object and a turbulence phase screen. The phase screen distorts the object producing a short exposure image. Several of these short exposure images are utilized in the image reconstruction process by using either the KT or TC recovery techniques. The phase screens and short exposure images are presented in the form of two-dimensional square arrays. The arrays produced are of dimension $64 \times 64$ consisting of 4096 elements. Several operations are performed on each element in these arrays throughout the entire simulation process. As a result, the requirement arose for a fast computer to process the arrays and for a large random access memory (RAM) to store them during the process.

A personal computer was used for the simulation process and the data reduction. The computer, a Compaq Deskpro 80386/20 with 16 megabytes of RAM and a Weitek 1167 coprocessor, provided ample speed and convenience as long as array sizes did not exceed $64 \times 64$. Standard Fortran 77 was the language used throughout the simulation. A Microway 32 bit NDP Fortran-386 compiler provided the speed, precision, and array processing capabilities required for the simulation. The construction of each short exposure image and its subsequent cross-spectrum or
bispectrum estimation, required approximately 28 to 34 seconds to process with this computer-compiler combination, depending on the imaging parameters (coherence length, photon count, short exposure image quantity). Surfer and Grapher software from Golden Software provided plots of image data, phasor spectrum SNR comparison data, and azimuthal RMS phase error (henceforth, simply phase error) comparison.

B. SIMULATION PROCESS

The simulation process compared the KT and TC phasor recovery techniques. The necessity for two programs, one using the KT technique and the other the TC technique, arose from RAM limitations. To ensure accuracy, both programs were identical except for the individual phasor recovery subroutines. Additionally, the imaging parameters and the random number seeds were identical for each comparison run to ensure production of the same phase screens and, hence, the same short exposure images. With identical short exposure images and object power spectra, the only distinguishable difference between reconstructed images resulted from the utilization of different phasor recovery techniques.

The simulation utilized the speckle imaging procedure. This procedure involved the use of several short exposures, from 25 to 1600, to remove the effects of turbulence by means of an averaging process. This process determined the object's power and phasor spectra by calculating the autocorrelation and the cross-spectrum or bispectrum respectively, for each short exposure image. At the end, these values
were averaged and combined, providing the object's Fourier spectrum. A separate program filtered and transformed this spectrum to image space, yielding the recovered image. The simulation process is shown in Figure 4.1.

1. **Object Production**

Construction of an object that provided adequate detail to test the resolution of the two phasor recovery techniques was essential. Three objects were designed to compare the two techniques over several different image parameters. These objects were created, scaled, transformed to Fourier space, and normalized.

   a. **Object Creation**

   The first step in the process created the object. The option to construct one of three objects was provided. The first object resembled a finite-dimensional astronomical body centered in the array. The body was a convolution of a Gaussian function and a circular pupil function, giving it the appearance of a smooth planet. Since phasor spectrum SNR declines radially with spatial frequency, round objects provide a less than ideal choice for image recovery comparison. However, increasing the detail on the body provided a means to test the resolution capabilities of the two phasor recovery techniques. Seven Gaussian functions of various size and depths at random locations on the body provided craters. These craters gave the body the appearance of an asteroid. The randomness of the craters on the asteroid provided the additional detail to test resolution.
Figure 4.1 Simulation Process

- BEGIN SIMULATION
- CREATE OBJECT
- SCALE OBJECT
- NORMALIZE OBJECT
- FFT
- CREATE GAUSSIAN RANDOM ARRAY
- FILTER ARRAY WITH PSDF
- IFFT
- APPLY KL FUNCTIONS
- APPLY RYTOV APPROXIMATION
- FILTER WITH TPF
- FFT
- DETERMINE SQUARE MODULUS
- FFT
- NORMALIZE PHASE SCREEN

- COMPUTE PHASOR SPECTRUM
- FOURIER MOMENTS
- LAST SHORT EXPOSURE IMAGE?
- YES
- CALCULATE PHASOR SPECTRUM
- COMPUTE AZIMUTHAL SNR
- COMBINE POWER AND PHASOR SPECTRA
- LOW PASS FILTER FOURIER SPECTRUM
- CALCULATE RMS PHASE ERROR
- IFFT
- PLOT RECONSTRUCTED IMAGE
- NO

- LAST SHORT EXPOSURE IMAGE?
- YES
- COMBINE OBJECT AND PHASE SCREEN
- APPLY POISSON RANDOM PHOTON NOISE
- CENTROID IF DESIRED
- COMPUTE POWER SPECTRUM FOURIER MOMENT
- LAST SHORT EXPOSURE IMAGE?
- YES
- CALCULATE OBJECT POWER SPECTRUM
- SQUARE MODULUS
- FFT
- IFFT
- NORMALIZE PHASE SCREEN
The second object resembled a binary star. This object consisted of two delta functions placed symmetrically about the center. Initially, this object provided the ability to troubleshoot the program since the phase spectrum of an equal intensity binary star involves a square wave pattern that was easily recognizable. After the completion of troubleshooting, the binary star allowed a test of the ability of the phasor recovery techniques to resolve point objects at very low photon counts. One point had twice the intensity of the other to reduce any ambiguities brought about by symmetry.

The third object resembled a star. The object consisted of a delta function at the center of the array. The short exposure image of a point source yields the instantaneous incoherent transfer function representing distortion from the turbulence and the imaging system. The transfer function allowed recovery of the object's power spectrum.

b. Object Scaling

The object had to be scaled before use. Object scaling assured that the imaging parameters retained complete frequency information within the given array size and ensured that the size of the object was within acceptable limits. One such parameter was \( r_0 \), the coherence length, which was a measure of the amount of turbulence present in the atmosphere [Ref. 14]. For the simulation, values of 0.206 and 0.103 meters were chosen. Another parameter, \( |\mathcal{U}| \), was the offset value. This value represented the number of pixels (array elements), in Fourier space, contained within the coherence length. The offset maximized the averaging
of cross-spectrum and bispectrum while preventing loss of high frequency information. Offset values of two and four were chosen for the simulation. The corresponding width of each pixel in terms of coherence length was determined and divided into the telescope diameter to calculate the number of pixels retained by this telescope under the conditions of the above parameters. The number of pixels is synonymous with the frequency cutoff of the diffraction-limited telescope incoherent transfer function \((D/\lambda)\). The image array size limits the frequency cutoff value to

\[ f_c \geq \frac{n}{2} - 1 , \quad (4.1) \]

where \(f_c\) is the frequency cutoff and \(n\) is the array size. If the frequency cutoff is too large, frequency information is lost.

After constraining the imaging parameters, the field of view and the object size were ascertained. The field of view (FOV) is equivalent to the reciprocal of the fundamental spatial frequency

\[ \text{FOV} = |\mathcal{U}| \cdot \left( \frac{\lambda}{r_0} \right) , \quad (4.2) \]

where \((\lambda/r_0)\) is the seeing disk. In general, the allowance of one seeing disk width between the object and each side of the array provided for object distortion effects. The size of the asteroid was then maximized to provide the most visual detail while satisfying the above criteria. Verification that the binary star and the star obeyed the above criteria was sufficient for those objects.
c. **Fourier Spectrum Determination**

Production of phase screens occurred in Fourier space, while object production transpired in image space. Proper image production required transformation to frequency space. A two-dimensional Fast-Fourier-transform (2D-FFT) algorithm transformed the object to Fourier space. The FFT provides a fast and accurate method of transforming a discrete function to Fourier space. The 2D-FFT employs a 1D-FFT provided by Gonzalez and Wintz [Ref. 15]. This 1D-FFT determines the discrete Fourier transform of a complex one-dimensional array of numbers. The 2D-FFT simply calls the 1D-FFT for each row then each column of the two-dimensional object array. Testing of the 2D-FFT by transforming a normalized pupil function then inverse transforming enabled comparison between the results and the original function. With double precision complex numbers, the 2D-FFT provided accuracy to ten significant figures. In addition to determining the object's Fourier spectrum, this 2D-FFT provided the means for transformation, from image space to Fourier space and back, extensively throughout the simulation.

d. **Fourier Spectrum Normalization**

Normalization of the object's Fourier spectrum produced the correct number of photons in the short exposure image. In reality, each short exposure image furnishes a photon count, however for the simulation, the same value was chosen for all short exposures used for each image reconstruction run. Dividing the object's Fourier spectrum by the photon count normalized the spectrum. Since the phase map was normalized by its DC value, the photon count was the same for the
object and for the short exposure image. The normalization was important for Poisson noise generation introduced later in the process.

2. Turbulence Phase Screen Production

Construction of a turbulence phase screen that correctly resembled true atmospheric turbulence allowed accurate phasor recovery technique comparison. Testing the KLFFT method of phase screen production showed it represented the actual 5/3 power law structure function closely [Ref. 16]. Therefore, this method was adapted to produce the phase screens in this simulation.

a. Gaussian Random Number Array

Each phase screen involved an array of random numbers. The random numbers represented the random phases produced by turbulence. A Gaussian distributed random number generator, provided by the subroutine Gauss, produced the required random numbers that represented the randomness of turbulence statistics.

b. Filter Function

The simulation used a filter function that represented the square root of the Kolmogorov power spectral density function, equation (3.5). This function represented turbulence statistics and filtered each array of Gaussian distributed random numbers to provide the turbulence structure function. The resulting array elements $\phi_{j}^{FPT}$, when put in terms of the Rytov approximation, modelled the 5/3
power law structure function, and hence provided a model for turbulence, though the low spatial frequencies were under-represented.

c. Karhunen-Loeve Functions

The simulation used the first five KL functions. Each filtered array was inverse Fourier transformed and then the five KL functions were applied in image space to compensate for the low spatial frequency under-representation. This application involved more than simple multiplication. The inner product of the KL functions and the filtered array provided scalars which expressed the amount of each individual KL function contained in the array. These amounts were subtracted from the array. The technique involved random numbers with variances equal to the eigenvalues associated with their respective KL function. Multiplying the random numbers by their corresponding KL functions and adding the result to the array gave the corrected phase screen. The resulting array elements $\phi_j^{KLFFT}$, when put in terms of the Rytov approximation, accurately depicted the 5/3 power law structure function.

An effect of the KLFFT method was the inclusion of tilt in the short exposure image. An option was given in the simulation which allowed the removal of tilt by setting the first two KL functions to zero. This option allowed phasor recovery without the presence of tilt and was utilized as a criterion for comparison of the phase recovery techniques.
d. Incoherent Transfer Function

Use of the Rytov approximation determined the incoherent transfer function of the turbulence and imaging system. The Rytov approximation, 

\[ A \cdot e^{i\phi_j}, \]  

provides the coherent transfer function of the turbulence and imaging system, where \( A \) represents the amplitude (set to one) and \( \phi_j \) represents the realization of the phase screen determined above. Calculating the autocorrelation of the coherent transfer function produced the incoherent transfer function. This calculation first required multiplying the phase screen array produced by the Rytov approximation by the pupil function of the telescope. Squaring the modulus of the Fourier transform of this array and Fourier transforming and normalizing the result yielded the incoherent transfer function of the true phase screen.

3. Object Degradation

The product of the phase screen and the object’s Fourier spectrum yielded the short exposure image spectrum. The inverse Fourier transform of this spectrum resulted in the required short exposure image. Since the object was normalized to the photon count of the short exposure image and the phase screen was normalized to unity, the final step in the degradation process was to apply photon noise to the short exposure image. The photon noise effect was added to the short exposure image by entering each image element into a Poisson distributed random function generator provided by the Poisson function in the simulation. The generator returned
a random number at each image point drawn from a Poisson distribution with mean equal to that value. As the photon count decreased from infinity, the randomness of the returned values increased which caused the grainy, photon noise effect. With photon noise included, the short exposure image was complete.

4. Centroiding

The simulation supported centroiding the degraded short exposure in image space. If desired, the image array was shifted to its true centroid first by column, then by row, based on the centroids determined from equations (3.32) and (3.33). Determining the centroid of the binary star object modified with equal intensity stars checked the accuracy of the centroid subroutine. The object initially had an arbitrary translation from its centroid. Centroiding this translated object using the subroutine then analyzing its phase spectrum ensured the subroutine performed correctly. Image centroiding worked well on high light-level images of the asteroid above $10^4$ photons. Images below this level had unevenly distributed intensities and centroiding was ineffective.

5. Image Recovery

The simulation separated image recovery into two distinct parts. First, the Labeyrie technique provided the object's power spectrum. Use of this technique for both programs ensured uniformity in comparison. Second, the KT and TC techniques reconstructed the object's phasor spectrum, each in separate programs.
a. **Power Spectrum Recovery**

Recovery of the object's power spectrum provided the modulus of the object's Fourier spectrum. Estimates of the autocorrelation were calculated for each short exposure image of the object and the point source. Averages of these results yielded the average power spectrum. Normalizing both power spectrum arrays by their respective DC values provided an equivalent intensity basis. Dividing the object's average power spectrum by the point source's average power spectrum removed imaging system errors. The square root of this result multiplied by the photon count furnished the object's modulus.

b. **Phasor Spectrum Recovery**

The object's phasor spectrum, with its modulus, determined the object's Fourier spectrum. Each short exposure image provides estimates of the either the cross-spectrum or the bispectrum, for various offset values. Several of these estimates, each with a different offset value, determined a specific phasor array element by averaging these estimates over all short exposure images, then over all offset values.

Recursive calculation, outward from the origin of the image array, supplied the estimates, and hence the object's phasor spectrum. The number of estimates which existed within an estimation circle of integer pixel radius equal to the whole part of the pixel distance of the desired point determined the maximum number of estimates and the offset value for each phasor estimated. The estimation circle began at the origin, and the estimates of the phasor spectrum points began one
radial pixel distance beyond that. The phasor at the origin had a value of one, since it had no imaginary part, and it was normalized. Each phasor spectrum point included more estimations as the recursive process proceeded, out to the maximum radial offset value. The same estimates of the phasor spectrum were found for every short exposure image and then averaged. Averaging a set of estimates associated with a phasor spectrum point produced the corresponding phasor value for that point.

Using the phasor spectrum of the uncorrupted binary star object modified with equal intensity stars verified the phasor recovery process. Comparing the phasor spectrum of the binary star before and after the recovery process using complex double precision numbers provided a match for all points to ten significant figures.

The phasor recovery process required an extensive amount of calculation. The time for phase reconstruction was approximately 30 seconds for every short exposure image. The process was made less time consuming by invoking Hermitian symmetry. Hermitian symmetry dictates

$$O_{i,j} = O_{j,i}^*$$  \hspace{1cm} (4.4)

thereby allowing half the number of calculations to determine the full object's phasor spectrum.

At the end of the phasor recovery operation, the variance of the cross-spectrum or bispectrum estimates provided an SNR value for each phasor element from equations (3.34) through (3.37). The square of the SNR for each estimate was
then multiplied by the corresponding estimate when the phasor spectrum points were calculated. This calculation provided a weighted least squares estimation of the object's phasor spectrum. When combined with the recovered modulus, the object's Fourier spectrum resulted.

6. Azimuthal Signal-To-Noise Ratio

Averaging the SNR values of the cross-spectrum and bispectrum provided an average SNR value for each phasor element. This average SNR array was then averaged azimuthally, one radial pixel value at a time, from the origin out to the cutoff frequency to provide a SNR as a function of radial spatial frequency. This radial SNR provided one means to compare the two phasor recovery techniques as well as to determine the frequency at which noise overcame signal to enable proper filtering.

7. Fourier Spectrum Filtering

The final result of the simulation process was a weighted least squares estimate of the object's Fourier spectrum. Before inverse Fourier transformation, the object's Fourier spectrum required filtering. A simple rectangular low-pass filtering method, which truncated spatial frequencies beyond the radial frequency where the azimuthal SNR was unity, determined the object's Fourier spectrum. This filtering process was provided by a separate program which included a method to determine the phase error of the recovered image's Fourier spectrum.
8. **Azimuthal RMS Phase Error**

Measuring the phase error of the two phasor recovery techniques produced another means for comparison. The object's phasor spectrum determined its phase spectrum. Computing the phase spectrum modulo $2\pi$, then subtracting the result from the object phase spectrum of the true object representation provided the array point phase error. The square of this error was determined then averaged azimuthally one radial pixel value at a time, from the origin out to the cutoff frequency. The square root of this average provided the azimuthal RMS phase error.

**C. SUMMARY**

The simulation process obtained the reconstructed image from several short exposures images. The process produced the desired object and the phase screens to make the short exposure images required for speckle imaging. With several short exposure images of the object, the Labeyrie technique recovered its power spectrum and the Knox-Thompson and Triple-Correlation techniques recovered its phasor spectrum. Low-pass filtering removed noise and the SNR and phase error calculations presented means for comparison of the two phasor recovery techniques. The inverse Fourier transform of the filtered spectrum yielded the recovered image presented in the form of a contour plot.
V. SIMULATION RESULTS

A. RECOVERY TECHNIQUE COMPARISON CRITERIA

Seven criteria provided a basis for comparison of the KT and TC phasor recovery techniques. Each criterion tested the reconstructed image's resolution produced by both algorithms over a range of values for a specific imaging parameter. A baseline of imaging parameters (Table A.I) was established. Typically one parameter was varied within an individual criterion. Each criterion used a range of two to four imaging parameter values. The criteria included reconstructed image evaluation based on the quantity of short exposure images, the short exposure image photon count, and the amount of turbulence. Additionally, the effects of short exposure image tilt on reconstruction as well as centroiding in image space to remove it, were weighed. Further, the effect of offset value on cross-spectrum and bispectrum estimates and object size on image resolution, were rated.

The techniques were judged in terms of image resolution, phasor spectrum SNR, and phase error. Each comparison included both the KT and TC image reconstructions. The evaluation included two-dimensional graphs of the SNR values for both the KT and TC image reconstruction as well as their phase error values. The reconstructed images appear with normalized intensities on two-dimensional contour plots with hachure marks indicating the direction of minima. The SNR and phase error values for each comparison were plotted against spatial frequency. The
maximum frequency value was the last point at which the SNR had a value of one or greater. The reconstructed image plots were visually compared to the object’s true image. By contrast, their SNR and phase error values were compared to each other.

B. RECOVERY TECHNIQUE COMPARISONS

Appendix A maintains the results of the recovery technique comparisons within the seven criteria. Table A.I delineates the imaging parameters involved in the simulation process and whether these values are variable over these criteria. Figures A.1 and A.2 represent the true representations of the asteroid and binary star respectively. These figures show the results of the actual computer generated objects without turbulence corruption, filtering, or modification resulting from the imaging system including aperture effects, which all other figures include. They were the reference figures for the reconstructed images and the phase error calculations. Figures A.3 through A.22 are plots of single short exposure images of the asteroid, the binary star and the star that show the effects of varying coherence lengths and photon counts. They show the level of distortion the objects realize in the imaging process. Figures A.23 through A.25 are plots of long exposure images that consisted of an average of 100 short exposure images with tilt. The inclusion of these images provides an appreciation for the necessity of image reconstruction to acquire an image that more closely resembles the truth.
1. **Short Exposure Image Quantity**

   Image resolution increases with the quantity of short exposure images used in the reconstruction process. Varying the quantity of images with no tilt present provided the first comparison criterion for the recovery techniques. For the four comparisons, the values of $N_f$ were 400 (baseline), 25, 100, and 1600 short exposure images. All other baseline parameters remained unchanged. The relevant plots and graphs are Figures A.26 through A.41. In all cases, the visual image quality of the TC recovered images was superior to those recovered by the KT process. This image quality distinction was especially noticeable at the lower $N_f$ values of 25 and 100. As the value of $N_f$ increased, the distinction decreased to the point where it was only slightly noticeable at the $N_f$ of 1600. Analysis of the phasor spectrum SNR graphs showed, in general, that the TC SNR curves were offset toward approximately ten to 20 percent greater SNR values than those of the KT SNR curves. Further, the phase error graphs showed the error curves resulting from the TC method were offset toward approximately five to 15 percent lesser error than those of the KT method. These two series of graphs confirmed that in all cases, especially at low $N_f$, the TC technique outperformed the KT technique.

2. **Photon Count**

   Image resolution increases with the amount of short exposure image photons present. Varying the quantity of photons in the short exposure images with no tilt present provided the second comparison criterion for the recovery techniques. For the four comparisons, the photon count values were $10^5$ (baseline), $10^6$, $10^4$, and
$10^3$ photons where all other baseline parameters remained unchanged. Figures A.26 through A.29 and A.42 through A.53 are applicable for this criterion. The visual quality of the TC and KT reconstructed images for photon counts of $10^6$ and $10^5$ was almost identical. For photon counts of $10^4$ and $10^3$, the KT technique produced slightly better reconstructed images. This photon count distinction of recovery technique performance was confirmed by both the SNR and phase error graphs. For low photon count short exposure images, the KT technique SNR curves were offset toward approximately ten to 20 percent greater SNR values than those of the TC technique, and the phase error curves were generally offset toward five to 25 percent lesser error values. Therefore, for low photon count image recovery without tilt, the KT phasor recovery technique provides slightly better resolution.

3. **Turbulence Magnitude and Offset Value**

As the amount of turbulence increases, the resolution of the reconstructed image decreases. Short exposure images with no tilt of coherence lengths 0.103 and 0.206 meters (baseline), provided the third and fourth comparison criteria for the recovery techniques. For the turbulence magnitude criterion, the 0.103 meter coherence length images required an offset value of two, to ensure inclusion of all spatial frequencies. For consistency, the 0.206 meter coherence length images used the same value. Comparison between offset values of two and four satisfied the offset criterion for TC and KT reconstructed images. Offset values effect the quantity of cross-spectrum or bispectrum averaging and the reconstructed image quality increases with increasing offset value. All other baseline parameters remained
unchanged. Figures A.54 through A.61 showed the comparisons of the turbulence criterion and Figures A.26 through A.29 and A.58 through A.61 showed the comparisons of the offset criterion. For the coherence length case, the KT technique provided a slightly better image with greater turbulence. The TC SNR curve was offset toward approximately five to ten percent greater SNR values relative to the KT curve. However, the KT phase error curve was offset toward approximately five percent lesser error values providing the more resolved image. For the offset value case, the TC technique produced a slightly better image, though the SNR and phase error curves were almost coincident. This apparent contradiction arose from TC techniques having more frequency values above the unity SNR value, thereby providing higher frequencies for the filtering process.

4. Tilt

The resolution of the reconstructed image declines with the inclusion of tilt in the short exposure images. The previous criterion comparisons were conducted without the presence of tilt. The addition of tilt provides a more realistic comparison of the phasor recovery techniques as tilt is always present in true short exposure images. Varying the photon count of the short exposure images with tilt present provided the fifth criterion for comparison of the recovery techniques. For the three comparisons, the photon count values were $10^2$, $10^4$, and $10^3$ photons and all other baseline parameters remained unchanged. Figures A.62 through A.77 were applicable for this criterion. In all photon count cases, the TC recovered image were superior. The KT SNR curves were generally offset toward 30 to 50 percent lesser
SNR values and dropped below the unity value at lower spatial frequencies than the TC curves. Hence, fewer high frequency values were included in the filtering process, producing a poorer resolution. The phase error curves for the TC recovered images were offset toward approximately 20 to 50 percent lesser error values than for those recovered using the KT process except at the $10^5$ photon count. At this value, however, the KT SNR curve was offset toward much lower SNR values. Consequently, the TC technique was found to be superior when tilt was included in the short exposure images.

5. Centroiding

Centroiding the short exposure images prior to image reconstruction enhances both the TC and KT recovery techniques for high photon counts. Short exposure image centroiding provided the sixth criterion for comparison of the recovery techniques by again varying the photon count of the short exposure images. With tilt present, the images were centroided prior to reconstruction. For the three comparisons, the photon count values were $10^5$, $10^4$, and $10^3$ photons and all other baseline parameters remained unchanged. Additionally, a comparison with $10^5$ photons and 1600 short exposures with and without tilt tested the effects of centroiding at high $N_f$ values. Figures A.78 through A.93 were relevant for this criterion. Centroiding offered a two to five percent improvement to both recovery methods with higher photon count. At low photon counts such as $10^3$ photons, centroiding actually offset the phase error curves toward higher error values. For the TC method, centroiding should have no effect or the effect should disappear with
large enough $N_r$ values since the method is shift invariant. However, at $10^4$ photons and with 1600 short exposure images, a minor improvement in phase error occurred with centroiding. Centroiding did not improve the reconstructed image to the point of those recovered images having no tilt, nor did centroiding bring the KT method results in line with that of the TC method. However, a minor improvement in phase error occurred for both methods.

6. Point Objects

The resolution of the reconstructed image depends upon its size and detail. Resolution of an object such as a binary star occurs more easily because of the requirement for less short exposure image photons. A binary star with one star having twice the intensity as its counterpart, provided the seventh comparison criterion for the phasor recovery techniques. The binary star was corrupted by turbulence of coherence length 0.103 and 0.206 meters and with short exposure image photon counts of $10^2$ and $10^3$ photons. Figures A.94 through A.109 are germane. The two techniques produced identical results at the higher photon count and lower turbulence values. With lower photon count and greater turbulence, the TC technique provided greater resolution of the stars. Both the SNR and phase error curves supported this fact.

7. Recovery Technique Comparison Findings

For the eventual use of real objects in future applications of the recovery techniques where tilt is inherent in short exposure images, triple-correlation was the
superior reconstruction technique. The triple-correlation technique exceeded the Knox-Thompson technique by far in the comparisons where tilt was a concern. The shift invariance of the triple-correlation technique provided the ability to resolve low light-level objects where the Knox-Thompson technique failed. Though the Knox-Thompson technique required eight percent less time for image reconstruction, the resolution improvement obtained by the TC approach outweighed the computational time efficiency of the competing technique.
VI. CONCLUSION

A. OVERVIEW

This thesis compared the Knox-Thompson and triple-correlation phasor recovery techniques. Since speckle imaging involves extensive calculations, comparison of the two techniques required a powerful computer. The simulation produced an object and a phase screen for each short exposure image. The diffraction-limited information in these images allowed reconstruction of the object's power and phasor spectra. Combining these spectra produced the reconstructed image. Image reconstruction comparison under seven imaging criteria permitted the ability to determine the superior technique. The triple-correlation technique provided the best overall image resolution. This judgement stems from its superiority with regard to realistic short exposure images which included tilt.

B. OPTIMUM IMAGE RECOVERY APPROACH

This thesis found that, of the two phasor recovery techniques compared, the triple-correlation technique was the optimum approach for real short exposure image recovery. From the shift invariance of the triple-correlation technique, attainment of 20 to 50 percent less azimuthal phase error values occurred when compared with the Knox-Thompson technique. As a result, use of the triple-correlation phasor recovery technique is essential. Specifically, removal of the noise bias compensates
for the random photon noise that is intrinsic to real images. The use of phasor vice
phase recovery is key to avoid the phase $2\pi$ wrap-around problem. The weighting
of the phasors determined from their cross-spectrum or bispectrum estimates by the
least squares estimation approach, is critical. The recursion method used to provide
the phasors is not imperative, and other techniques may be used, such as the method
of least squares or the method of steepest decent, not discussed in this thesis. With
regard to imaging parameters, the triple-correlation technique allowed larger
maximum offset values than the Knox-Thompson technique; $D/\lambda$ instead of $x_0/\lambda$.
This provides maximum estimate averaging. Based on the results with centroiding,
it is helpful for relatively high light level short exposure images. Finally, peak
recovered image resolution requires the maximum amount of short exposure images
practicable.

C. FURTHER STUDY

This thesis provided simulated results for comparison of the two recovery
techniques. Application of the triple-correlation technique to actual, turbulence-
degraded images provides an avenue of research. Development and testing of other
methods of extracting the phasor spectrum beyond the recursion method demands
analysis. Reconstructed Fourier spectrum filtering processes beyond the simple
rectangular low pass filtering approach used herein require exploration.


APPENDIX A. PLOTS AND GRAPHS

The following table, plots, and graphs were referenced in the text.

Figure A.I  Imaging Parameters.

<table>
<thead>
<tr>
<th>Imaging Parameter</th>
<th>Parameter Abbrev.</th>
<th>Parameter Baseline</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short Exposure Image Quantity</td>
<td>N_f</td>
<td>400</td>
<td>Variable</td>
</tr>
<tr>
<td>Coherence Length</td>
<td>r_0</td>
<td>0.206 (m)</td>
<td>Variable</td>
</tr>
<tr>
<td>Short Exposure Image Photon Quantity</td>
<td>N_p</td>
<td>10^5</td>
<td>Variable</td>
</tr>
<tr>
<td>Offset Value</td>
<td>OS</td>
<td>4 (pixels)</td>
<td>Variable</td>
</tr>
<tr>
<td>Telescope Primary Diameter</td>
<td>D</td>
<td>1.6 (m)</td>
<td>Constant</td>
</tr>
<tr>
<td>Telescope Secondary Diameter</td>
<td>D_s</td>
<td>0.33 (m)</td>
<td>Constant</td>
</tr>
<tr>
<td>Aperture Radius</td>
<td>a</td>
<td>31 (pixels)</td>
<td>Constant</td>
</tr>
<tr>
<td>Light Wavelength</td>
<td>( \lambda )</td>
<td>5.5 \times 10^{-7} (m)</td>
<td>Constant</td>
</tr>
<tr>
<td>Cutoff Frequency</td>
<td>( f_c )</td>
<td>68.3 (1/arcsec)</td>
<td>Constant</td>
</tr>
<tr>
<td>Random Number Seed</td>
<td>s</td>
<td>123456789</td>
<td>Constant</td>
</tr>
</tbody>
</table>

51
Figure A.1 True Asteroid Representation With No Aperture Affects.

Figure A.2 True Binary Star Representation With No Aperture Affects.
Figure A.3  Asteroid Short Exposure Image: 
\( r_0 = 0.206, N_p = 10^6 \).

Figure A.4  Asteroid Short Exposure Image, 
\( r_0 = 0.206, N_p = 10^5 \).
Figure A.5 Asteroid Short Exposure Image, \( r_0 = 0.206, N_p = 10^4 \).

Figure A.6 Asteroid Short Exposure Image, \( r_0 = 0.206, N_p = 10^3 \).
Figure A.7 Asteroid Short Exposure Image, $r_0 = 0.103$, $N_p = 10^6$.

Figure A.8 Asteroid Short Exposure Image, $r_0 = 0.103$, $N_p = 10^5$. 
Figure A.9  Asteroid Short Exposure Image, $r_0 = 0.103, N_p = 10^4$.

Figure A.10  Asteroid Short Exposure Image, $r_0 = 0.103, N_p = 10^3$. 
Figure A.11 Binary Star Short Exposure Image, \( r_0 = 0.206, N_p = 10^5 \).

Figure A.12 Binary Star Short Exposure Image, \( r_0 = 0.206, N_p = 10^2 \).
Figure A.13  Binary Star Short Exposure Image, \( r_0 = 0.103, N_p = 10^2 \).

Figure A.14  Binary Star Short Exposure Image, \( r_0 = 0.103, N_p = 10^2 \).
Figure A.15 Star Short Exposure Image, 
\( r_0 = 0.206, N_p = 10^5 \).

Figure A.16 Star Short Exposure Image, 
\( r_0 = 0.206, N_p = 10^4 \).
Figure A.17 Star Short Exposure Image, \( r_0 = 0.206, N_p = 10^3 \).

Figure A.18 Star Short Exposure Image, \( r_0 = 0.206, N_p = 10^2 \).
Figure A.19  Star Short Exposure Image, $r_0 = 0.103, N_p = 10^3$.

Figure A.20  Star Short Exposure Image, $r_0 = 0.103, N_p = 10^4$.
Figure A.21 Star Short Exposure Image, \( r_0 = 0.103, N_p = 10^3 \).

Figure A.22 Star Short Exposure Image, \( r_0 = 0.103, N_p = 10^2 \).
Figure A.23 Asteroid Long Exposure Image, 
$r_0 = 0.206, N_p = 10^5$.

Figure A.24 Binary Star Long Exposure Image, 
$r_0 = 0.206, N_p = 10^5$. 
63
Figure A.25 Star Long Exposure Image, $r_0 = 0.206, N_p = 10^5$. 
Figure A.26 Baseline KT Recovered Asteroid, No Tilt, $r_0 = 0.206$, $N_p = 10^5$, $N_t = 400$, OS = 4.

Figure A.27 Baseline TC Recovered Asteroid, No Tilt, $r_0 = 0.206$, $N_p = 10^5$, $N_t = 400$, OS = 4.
Figure A.28 Asteroid Baseline Phasor Spectrum SNR.

Figure A.29 Asteroid Baseline Azimuthal RMS Phase Error.
Figure A.30 KT Recovered Asteroid, No Tilt, $r_0 = 0.206, N_p = 10^5, N_t = 25, OS = 4.$

Figure A.31 TC Recovered Asteroid, No Tilt, $r_0 = 0.206, N_p = 10^5, N_t = 25, OS = 4.$
Figure A.32  Asteroid Phasor Spectrum SNR, $N_f = 25$.

Figure A.33  Asteroid Azimuthal RMS Phase Error, $N_f = 25$. 
Figure A.34 KT Recovered Asteroid, No Tilt, 
r_0 = 0.206, N_p = 10^5, N_t = 100, OS = 4.

Figure A.35 TC Recovered Asteroid, No Tilt, 
r_0 = 0.206, N_p = 10^5, N_t = 100, OS = 4.
Figure A.36 Asteroid Phasor Spectrum SNR, $N_f = 100$.

Figure A.37 Asteroid Azimuthal RMS Phase Error, $N_f = 100$. 
Figure A.38 KT Recovered Asteroid, No Tilt, 
\( r_0 = 0.206, N_p = 10^5, N_f = 1600, OS = 4 \).

Figure A.39 TC Recovered Asteroid, No Tilt, 
\( r_0 = 0.206, N_p = 10^5, N_f = 1600, OS = 4 \).
Figure A.40 Asteroid Phasor Spectrum SNR, $N_t = 1600$.

Figure A.41 Asteroid Azimuthal RMS Phase Error, $N_t = 1600$. 
Figure A.42 KT Recovered Asteroid, No Tilt, $r_0 = 0.206, N_p = 10^6, N_f = 400, OS = 4$.

Figure A.43 TC Recovered Asteroid, No Tilt, $r_0 = 0.206, N_p = 10^6, N_f = 400, OS = 4$. 
Figure A.44 Asteroid Phasor Spectrum SNR, $N_p = 10^6$.

Figure A.45 Asteroid Azimuthal RMS Phase Error, $N_p = 10^6$. 
Figure A.46  KT Recovered Asteroid, No Tilt, \( r_0 = 0.206, N_p = 10^4, N_r = 400, \text{OS} = 4. \)

Figure A.47  TC Recovered Asteroid, No Tilt, \( r_0 = 0.206, N_p = 10^4, N_r = 400, \text{OS} = 4. \)
Figure A.48 Asteroid Phasor Spectrum SNR, $N_p = 10^4$.

Figure A.49 Asteroid Azimuthal RMS Phase Error, $N_p = 10^4$. 

76
Figure A.50  KT Recovered Asteroid, No Tilt, $r_0 = 0.206, N_p = 10^3, N_f = 400$, OS = 4.

Figure A.51  TC Recovered Asteroid, No Tilt, $r_0 = 0.206, N_p = 10^3, N_f = 400$, OS = 4.
Figure A.52 Asteroid Phasor Spectrum SNR, $N_p = 10^3$.

Figure A.53 Asteroid Azimuthal RMS Phase Error, $N_p = 10^3$. 

78
Figure A.54 KT Recovered Asteroid, No Tilt, $r_0 = 0.206$, $N_p = 10^5$, $N_f = 400$, OS = 2.

Figure A.55 TC Recovered Asteroid, No Tilt, $r_0 = 0.206$, $N_p = 10^5$, $N_f = 400$, OS = 2.
Figure A.56 Asteroid Phasor Spectrum SNR, OS = 2.

Figure A.57 Asteroid Azimuthal RMS Phase Error, OS = 2.
Figure A.58 KT Recovered Asteroid, No Tilt, $r_0 = 0.103$, $N_p = 10^5$, $N_t = 400$, OS = 2.

Figure A.59 TC Recovered Asteroid, No Tilt, $r_0 = 0.103$, $N_p = 10^5$, $N_t = 400$, OS = 2.
**Figure A.60** Asteroid Phasor Spectrum SNR, $r_0 = 0.013$, OS = 2.

**Figure A.61** Asteroid Azimuthal RMS Phase Error, $r_0 = 0.103$, OS = 2.
Figure A.62 KT Recovered Asteroid, With Tilt, $r_0 = 0.206$, $N_p = 10^5$, $N_f = 400$, OS = 4.

Figure A.63 TC Recovered Asteroid, With Tilt, $r_0 = 0.206$, $N_p = 10^5$, $N_f = 400$, OS = 4.
Figure A.64 Asteroid Phasor Spectrum SNR, Tilt.

Figure A.65 Asteroid Azimuthal RMS Phase Error, Tilt
Figure A.66 KT Recovered Asteroid, with Tilt, $r_0 = 0.206$, $N_p = 10^4$, $N_f = 400$, OS = 4.

Figure A.67 TC Recovered Asteroid, with Tilt, $r_0 = 0.206$, $N_p = 10^4$, $N_f = 400$, OS = 4.
Figure A.68  Asteroid Phasor Spectrum SNR, Tilt, $N_p = 10^4$.

Figure A.69  Asteroid Azimuthal RMS Phase Error, Tilt, $N_p = 10^4$. 
Figure A.70  KT Recovered Asteroid, With Tilt, $r_0 = 0.206$, $N_p = 10^3$, $N_r = 400$, OS = 4.

Figure A.71  TC Recovered Asteroid, With Tilt, $r_0 = 0.206$, $N_p = 10^3$, $N_r = 400$, OS = 4.
Figure A.72 Asteroid Phasor Spectrum SNR, Tilt, $N_p = 10^3$.  

Figure A.73 Asteroid Azimuthal RMS Phase Error, Tilt, $N_p = 10^3$.  

88
Figure A.74 KT Recovered Asteroid,
With Tilt, $r_0 = 0.206$, $N_p = 10^3$,
$N_r = 1600$, OS = 4.

Figure A.75 TC Recovered Asteroid,
With Tilt, $r_0 = 0.206$, $N_p = 10^3$,
$N_r = 1600$, OS = 4.
Figure A.76 Asteroid Phasor Spectrum SNR, Tilt, $N_f = 1600$.

Figure A.77 Asteroid Azimuthal RMS Phase Error, Tilt, $N_f = 1600$. 

90
Figure A.78  KT Recovered Asteroid, With Tilt, Centroided, $r_0 = 0.206$, $N_p = 10^3$, $N_f = 1600$, OS = 4.

Figure A.79  TC Recovered Asteroid, With Tilt, Centroided, $r_0 = 0.206$, $N_p = 10^3$, $N_f = 1600$, OS = 4.
Figure A.80 Asteroid Phasor Spectrum SNR, Tilt, Centroiding, $N_f = 1600$.

Figure A.81 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding, $N_f = 1600$. 
Figure A.82 KT Recovered Asteroid, With Tilt, Centroided, $r_0 = 0.206$, $N_p = 10^5, N_f = 400, OS = 4$. 

Figure A.83 TC Recovered Asteroid, With Tilt, Centroided, $r_0 = 0.206$, $N_p = 10^5, N_f = 400, OS = 4$. 

93
Figure A.84 Asteroid Phasor Spectrum SNR, Tilt, Centroiding.

Figure A.85 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding.
Figure A.86 KT Recovered Asteroid, With Tilt, Centroided, $r_0 = 0.206$, $N_p = 10^4$, $N_t = 400$, OS = 4.

Figure A.87 TC Recovered Asteroid, With Tilt, Centroided, $r_0 = 0.206$, $N_p = 10^4$, $N_t = 400$, OS = 4.
Figure A.88 Asteroid Phasor Spectrum SNR, Tilt, Centroiding, $N_p = 10^4$.

Figure A.89 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding, $N_p = 10^4$. 
Figure A.90 KT Recovered Asteroid, With Tilt, Centroided, \( r_0 = 0.206 \), \( N_p = 10^3 \), \( N_f = 400 \), OS = 4.

Figure A.91 TC Recovered Asteroid, With Tilt, Centroided, \( r_0 = 0.206 \), \( N_p = 10^3 \), \( N_f = 400 \), OS = 4.
Figure A.92  Asteroid Phasor Spectrum SNR, Tilt, Centroiding, \( N_p = 10^3 \).

Figure A.93  Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding, \( N_p = 10^3 \).
Figure A.94  KT Recovered Binary Star, No Tilt, 
\( r_0 = 0.206, N_p = 10^3, N_t = 400, OS = 4. \)

Figure A.95  TC Recovered Binary Star, No Tilt, 
\( r_0 = 0.206, N_p = 10^3, N_t = 400, OS = 4. \)
Figure A.96 Binary Star Phasor Spectrum SNR, $N_p = 10^3$.

Figure A.97 Binary Star Azimuthal RMS Phase Error, $N_p = 10^3$. 
Figure A.98  KT Recovered Binary Star, No Tilt, 
$r_0 = 0.206, N_p = 10^2, N_t = 400, OS = 4.$

Figure A.99  TC Recovered Binary Star, No Tilt, 
$r_0 = 0.206, N_p = 10^2, N_t = 400, OS = 4.$
Figure A.100 Binary Star Phasor Spectrum SNR, $N_p = 10^2$.

Figure A.101 Binary Star Azimuthal RMS Phase Error, $N_p = 10^2$. 
Figure A.102  KT Recovered Binary Star, No Tilt, $r_0 = 0.103, N_p = 103, N_f = 400, OS = 4$.

Figure A.103  TC Recovered Binary Star, No Tilt, $r_0 = 0.103, N_p = 103, N_f = 400, OS = 4$. 

Figure A.104  Binary Star Phasor Spectrum SNR, \( r_0 = 0.103, N_p = 10^3 \).

Figure A.105  Binary Star Azimuthal RMS Phase Error, \
\( r_0 = 0.103, N_p = 10^3 \).
Figure A.106 KT Recovered Binary Star, No Tilt, $r_0 = 0.103$, $N_p = 10^2$, $N_f = 400$, OS = 4.

Figure A.107 TC Recovered Binary Star, No Tilt, $r_0 = 0.103$, $N_p = 10^2$, $N_f = 400$, OS = 4.
Figure A.108 Binary Star Phasor Spectrum SNR, $r_0 = 0.103$, $N_p = 10^2$.

Figure A.109 Binary Star Azimuthal RMS Phase Error, $r_0 = 0.103$, $N_p = 10^2$. 

Spatial Frequency (1/arcsec)
APPENDIX B. KNOX-THOMPSON MAIN PROGRAM

This program creates one of three images: a star, a binary star and an asteroid. It then degrades the image by simulating atmospheric conditions and photon noise, and then reconstructs the image using the Knox-Thompson algorithm.

Author: LT James M. Lackemacher
Comple Date: 26 October 1990
Reason: Complete requirements for a masters degree in physics.
Goal: Simulate object, degrade object, reconstruct object using Knox-Thompson and triple-correlation methods, filter and compare.

PROGRAM KNOXTHOMPSON

Main program complex variable list

F n DIM ARRAY USED IN THE FOURIER TRANSFORM
GAUSSIAN n x n DIM ARRAY THAT REPRESENTS THE GAUSSIAN PORTION OF THE ASTEROID
I n x n DIM ARRAY THAT REPRESENTS THE DEGRADED IMAGE
IKT n x n x 5 x 9 ARRAY THAT REPRESENTS THE CROSS-SPECTRUM OF THE IMAGE
IS n x n DIM ARRAY THAT REPRESENTS THE DEGRADED POINT SOURCE
ISKT n x n DIM ARRAY THAT REPRESENTS THE MODULUS SQUARED OF THE POINT SOURCE
OKT n x n DIM ARRAY THAT REPRESENTS THE OBJECT
SPECTRUM OF THE RECONSTRUCTED IMAGE
OBJDATA n x n DIM ARRAY THAT REPRESENTS THE OBJECT SPECTRUM OF THE RECONSTRUCTED IMAGE
KTPHASOR n x n DIM ARRAY THAT REPRESENTS THE PHASOR OF THE OBJECT IN THE RECONSTRUCTION PROCESS
PUPIL n x n DIM ARRAY THAT REPRESENTS THE PUPIL PORTION OF THE ASTEROID
TEMPDATA n x n DIM ARRAY THAT IS USED AS A TEMPORARY ARRAY

MAIN PROGRAM REAL VARIABLE LIST

KTsnr n x n DIM ARRAY THAT REPRESENTS THE SNR OF EACH PHASOR
mod n x n DIM ARRAY THAT REPRESENTS THE MODULUS OF THE RECONSTRUCTED IMAGE
rsnr n/2 DIM ARRAY THAT REPRESENTS THE SNR AS A FUNCTION OF RADIUS
xvarrkt n x n x 5 x 9 DIM ARRAY THAT REPRESENTS THE REAL PART OF THE VARIANCE OF THE CROSS-SPECTRUM
xvariikt n x n x 5 x 9 DIM ARRAY THAT REPRESENTS THE IMAGINARY PART OF THE VARIANCE OF THE CROSS-SPECTRUM

MAIN PROGRAM INTEGER VARIABLE LIST

fwd VALUE OF 1 FOR FORWARD FFT
icounter COUNTER THAT COUNTS THE NUMBER OF SNAPSHOTS
inv VALUE OF -1 FOR INVERSE FFT
ln 2^ln FOR USE WITH FFT SUBROUTINE
offset VARIABLE THAT REPRESENTS THE NUMBER OF PIXELS THAT ARE AVERAGED IN THE KNOX-THOMPSON PROCESS
mseed VARIABLE USED TO ENSURE ONLY ONE PASS OF INITIAL PART OF PHSUB SUBROUTINE
DIMENSION OF ONE SIDE OF 2-DIM ARRAY

TOTAL NUMBER OF SHORT EXPOSURE SNAPSHOTS

TOTAL NUMBER OF PHOTONS IN SNAPSHOT

EQUAL TO THE TELESCOPE PUPIL FUNCTION RADIUS

DERIVED FROM THE RELATION:

nyquist = (telescope diameter x number of pixels per r0)/r0

MAIN PROGRAM

PARAMETER(u=64,ln=6,fwd=1,inv=-1)

COMPLEX*16 OBJDATA(n,n), TEMPDATA(n,n), F(n),
+ PUPIL(n,n), STARDATA(n,n), GAUSSIAN(n,n),
+ KTPHASOR(n,n), IKT(n,n,5,9), ISKT(n,n),
+ I(n,n), IS(n,n), OKT(n,n)

REAL*8 mod(n,n), xvarrrkt(n,n,5,9), xvarikt(n,n,5,9),
+ KTSnr(n,n), rsnr(n/2)

INTEGER offset

CHARACTER*16 file1, file2
CHARACTER*1 cent

INITIALIZE MSEED TO ALLOW ONLY ONE PASS THROUGH FIRST PART OF PHSUB

mseed = 1

INITIALIZE PROGRAM READING REQUIRED VARIABLES AND CREATE THE DESIRED OBJECT

CALL Initialize(OBJDATA,GAUSSIAN,PUPIL,F,TEMPDATA,
+ file1,file2,fwd,inv,ln,nframes,nphoton,nyquist,
+ offset,cent,n)

TRANSFORM THE OBJECT TO FREQUENCY SPACE

CALL FFT2D(OBJDATA,TEMPDATA,F,ln,fwd,n)

109
NORMALIZE ALL OF THE ARRAY ELEMENTS TO THE VALUE OF THE NUMBER OF PHOTONS WHERE THE DC TERM EQUALS THE NUMBER OF PHOTONS OF THE IMAGE

CALL Photons(OBJDATA,nphoton,n)

CREATE THE POINT SOURCE

CALL Star(STARDATA,n)

TRANSFORM THE POINT SOURCE TO FREQUENCY SPACE

CALL FFT2D(STARDATA,TEMPDATA,F,ln,fwd,n)

NORMALIZE ALL OF THE ARRAY ELEMENTS TO THE VALUE OF THE NUMBER OF PHOTONS WHERE THE DC TERM EQUALS THE NUMBER OF PHOTONS OF THE IMAGE

CALL Photons(STARDATA,nphoton,n)

COMMENCE THE LOOP THAT COUNTS THE SNAPSHOTS

DO 10 icounter = 1, nframes

DEGRADE THE IMAGE WITH THE TELESCOPE, THE ATMOSPHERE, AND THE PHOTON NOISE

CALL PHSUB(I,F,TEMPDATA,OBJDATA,icounter,nframes, + nyquist,fwd,inv,ln,n,mseed)

CENTROID THE DEGRADED IMAGE ONLY SINCE ONLY PHASE IS EFFECTED BY CENTROIDING IF CENTROIDING IS DESIRED.

IF ((cent.EQ.'Y').OR.(cent.EQ.'y')) THEN
  CALL Centroid(I,TEMPDATA,n)
ENDIF

TRANSFORM THE DEGRADED IMAGE TO FREQUENCY SPACE

CALL FFT2D(I,TEMPDATA,F,ln,fwd,n)
c DEGRADE THE POINT SOURCE WITH THE TELESCOPE, THE
c ATMOSPHERE, AND THE PHOTON NOISE

CALL PHSUB(IS,F,TEMPDATA,STARDATA,icounter,nframes,
+       nyquist,fwd,inv,ln,n,mseed)

c TRANSFORM THE DEGRADED IMAGE TO FREQUENCY SPACE

CALL FFT2D(IS,TEMPDATA,F,ln,fwd,n)

C DETERMINE THE MODULUS OF THE DEGRADED IMAGE AND
C POINT SOURCE

CALL Modulus(I,IS,IKT,ISKT,mod,icounter,
+           nframes,nphoton,n)

C RECONSTRUCT THE OBJECT FROM THE DEGRADED IMAGE AND
C POINT SOURCE

CALL KTrecon(KTPHASOR,IKT,KTsrr,offset,xvarrkt,
+          xvarkt,icounter,nframes,nyquist,n)

WRITE(*,*)icounter,'FRAMES COMPLETED'

C END THE LOOP

10 CONTINUE

C CALCULATE THE AVERAGE SNR AS A FUNCTION OF RADIUS

CALL SNRcalc(KTsrr,rsnr,nyquist,n)

C COMBINE THE MODULUS WITH THE PHASOR

CALL Combine(OKT,mod,KTPHASOR,n)

C WRITE RECONSTRUCTED PHASE AND POWER SPECTRUM TO A
C FILE

CALL Writefile(OKT,rsnr,file1,file2,nyquist,n)

STOP
END
APPENDIX C. TRIPLE-CORRELATION MAIN PROGRAM

This program creates one of three images: a star, a binary star and an asteroid. It then degrades the image by simulating atmospheric conditions and photon noise, and then reconstructs the image using the triple-correlation algorithm.

Author: LT James M. Lackemacher
Completed Date: 26 October 1990
Reason: Complete requirements for a Masters degree in Physics.
Goal: Simulate object, degrade object, reconstruct object using Knox-Thompson and triple-correlation methods, filter and compare.

PROGRAM TRIPLECORR

MAIN PROGRAM COMPLEX VARIABLE LIST

BSPHASOR n x n DIM ARRAY THAT REPRESENTS THE PHASOR OF THE OBJECT IN THE RECONSTRUCTION PROCESS
F n DIM ARRAY USED IN THE FOURIER TRANSFORM
GAUSSIAN n x n DIM ARRAY THAT REPRESENTS THE GAUSSIAN PORTION OF THE ASTEROID
I n x n DIM ARRAY THAT REPRESENTS THE DEGRADED IMAGE
IBS n x n x 5 x 9 ARRAY THAT REPRESENTS THE BISPECTRUM OF THE IMAGE
IDBS 5 x 9 ARRAY THAT REPRESENTS THE DEVIATION FROM THE DC TERM
c IS n x n DIM ARRAY THAT REPRESENTS THE DEGRADED
  POINT SOURCE

c ISBS n x n DIM ARRAY THAT REPRESENTS THE MODULUS
  SQUARED OF THE POINT SOURCE

c OBS n x n DIM ARRAY THAT REPRESENTS THE OBJECT
  SPECTRUM OF THE RECONSTRUCTED IMAGE

c OBJDATA n x n DIM ARRAY THAT REPRESENTS THE
  OBJECT SPECTRUM OF THE RECONSTRUCTED IMAGE

c PUPIL n x n DIM ARRAY THAT REPRESENTS THE PUPIL
  PORTION OF THE ASTEROID

c TEMPDATA n x n DIM ARRAY THAT IS USED AS A TEMPORARY
  ARRAY IN THE FOURIER TRANSFORM

MAIN PROGRAM REAL VARIABLE LIST

c BSsnr n x n DIM ARRAY THAT REPRESENTS THE SNR OF
  EACH PHASOR

c mod n x n DIM ARRAY THAT REPRESENTS THE
  MODULUS OF THE RECONSTRUCTED IMAGE

c rsnr n/2 DIM ARRAY THAT REPRESENTS THE SNR AS A
  FUNCTION OF RADIUS

c xvarrbs n x n x 5 x 9 DIM ARRAY THAT REPRESENTS THE
  REAL PART OF THE VARIANCE OF THE BISPECTRUM

c xvaribs n x n x 5 x 9 DIM ARRAY THAT REPRESENTS THE
  IMAGINARY PART OF THE VARIANCE OF THE
  BISPECTRUM

MAIN PROGRAM INTEGER VARIABLE LIST

c fwd VALUE OF 1 FOR FORWARD FFT

c icounter COUNTER THAT COUNTS THE NUMBER OF
  SNAPSHOTS

c inv VALUE OF -1 FOR INVERSE FFT

c ln 2^ln FOR USE WITH FFT SUBROUTINE

c offset VARIABLE THAT REPRESENTS THE NUMBER OF
  PIXELS THAT ARE AVERAGE IN THE TRIPLE-
  CORRELATION PROCESS
mseed VARIABLE USED TO ENSURE ONLY ONE PASS OF INITIAL PART OF PHSUB SUBROUTINE

n DIMENSION OF ONE SIDE OF 2-DIM ARRAY

ntotal TOTAL NUMBER OF SHORT EXPOSURE SNAPSHOTS

nphtot TOTAL NUMBER OF PHOTONS IN SNAPSHOT

nyquist EQUAL TO THE TELESCOPE PUPIL FUNCTION RADIUS

Derived from the relation:

nyquist = (telescope diameter x number of pixels per r0)/r0

MAIN PROGRAM

PARAMETER(n=64,ln=6,fwd=1,inv=-1)

COMPLEX*16 OBJDATA(n,n), TEMPDATA(n,n), F(n),
+ PUPIL(n,n), I(n,n), STARDATA(n,n), GAUSSIAN(n,n),
+ BSPHASOR(n,n), IBS(n,n,5,9), ISBS(n,n), IS(n,n),
+ OBS(n,n), IDBS(5,9)
REAL*8 mod(n,n), xvarrbs(n,n,5,9), xvaribs(n,n,5,9),
+ BSsnr(n,n), rsnr(n/2)

INTEGER offset

CHARACTER*16 file1, file2
CHARACTER*1 cent

INITIALIZE MSEED TO ALLOW ONLY ONE PASS THROUGH FIRST PART OF PHSUB

mseed = 1

initialize program reading required variables and create the desired object

CALL Initialize(OBJDATA,GAUSSIAN,PUPIL,F,TEMPDATA,
+ file1,file2,fwd,inv,ln,ntotal, nphtot, nyquist, offset, cent, n)
TRANSFORM THE OBJECT TO FREQUENCY SPACE
CALL FFT2D(OBJDATA,TEMPDATA,F,ln,fwd,n)

NORMALIZE ALL OF THE ARRAY ELEMENTS TO THE VALUE OF
THE NUMBER OF PHOTONS WHERE THE DC TERM EQUALS THE
NUMBER OF PHOTONS OF THE IMAGE
CALL Photons(OBJDATA,nphoton,n)

CREATE THE POINT SOURCE
CALL Star(STARDATA,n)

TRANSFORM THE POINT SOURCE TO FREQUENCY SPACE
CALL FFT2D(STARDATA,TEMPDATA,F,ln,fwd,n)

NORMALIZE ALL OF THE ARRAY ELEMENTS TO THE VALUE OF
THE NUMBER OF PHOTONS WHERE THE DC TERM EQUALS THE
NUMBER OF PHOTONS OF THE IMAGE
CALL Photons(STARDATA,nphoton,n)

COMMENCE THE LOOP THAT COUNTS THE SNAPSHOTS
DO 10 icounter = 1, nframes

DEGRADE THE IMAGE WITH THE TELESCOPE, THE ATMOSPHERE,
AND THE PHOTON NOISE
CALL PHSUB(I,F,TEMPDATA,OBJDATA,icounter,nframes,+
+ nyquist,fwd,inv,ln,n,mseed)

CENTROID THE DEGRADED IMAGE ONLY SINCE ONLY PHASE IS
EFFECTED BY CENTROIDING IF CENTROIDING IS DESIRED.

IF ((cent.EQ.'Y').OR.(cent.EQ.'y')) THEN
   CALL Centroid(I,TEMPDATA,n)
ENDIF
c TRANSFORM THE DEGRADED IMAGE TO FREQUENCY SPACE
CALL FFT2D(I,TEMPDATA,F,ln,fwd,n)

C DEGRADE THE POINT SOURCE WITH THE TELESCOPE, THE ATMOSPHERE, AND THE PHOTON NOISE
CALL PHSUB(IS,F,TEMPDATA,STARDATA,icounter,nframes, +     nyquist,fwd,inv,ln,n,mseed)

C TRANSFORM THE DEGRADED IMAGE TO FREQUENCY SPACE
CALL FFT2D(IS,TEMPDATA,F,ln,fwd,n)

C DETERMINE THE MODULUS OF THE DEGRADED IMAGE AND POINT SOURCE
CALL Modulus(I,IS,IBS,ISBS,mod,icounter, +     nframes,nphoton,n)

C RECONSTRUCT THE OBJECT FROM THE DEGRADED IMAGE AND POINT SOURCE
CALL BSrecon(BSPHASOR,I,IBS,IDBS,BSnr,offset, +     xvarbs,xvaribs,icounter,nframes, +     nyquist,nphoton,n)
WRITE(*,*)icounter,'FRAMES COMPLETED'

C END THE LOOP
10 CONTINUE

C CALCULATE THE AVERAGE SNR AS A FUNCTION OF RADIUS
CALL SNRcalc(BSnr,rsnr,nyquist,n)

C COMBINE THE MODULUS WITH THE PHASOR
CALL Combine(OBS,mod,BSPHASOR,n)
WRITE RECONSTRUCTED PHASE AND POWER SPECTRUM TO A FILE

CALL Writefile(OBS,rsnr,file1,file2,nyquist,n)

STOP
END
APPENDIX D. UNIVERSAL IMAGE RECOVERY SUBROUTINES

THE FOLLOWING SUBROUTINES WERE GENERATED BY THE
THESIS AUTHOR AND ARE REQUIRED BY BOTH THE KNOX-
THOMPSON AND THE TRIPLE-CORRELATION PROGRAMS.
ADDITIONALLY, SOME SUBROUTINES ARE REQUIRED FOR
THE FILTRMS PROGRAM IN APPENDIX H.

SUBROUTINE LIST

SUBROUTINE Ast(ASTEROID,GAUSSIAN,PUPIL,F,TEMPDATA,
+       fwdinv,Fn)
COMPLEX*16 GAUSSIAN(n,n), ASTEROID(n,n), PUPIL(n,n),
+       TEMPDATA(n,n), F(n), E1, E2, E3, E4, E5, E6, E7
maxval = 0.0
aval = 4.0
n2p1 = n/2 + 1
rad = 16.0
DO 10 i = 1, n
   DO 10 j = 1, n
      x = float(j - (n2p1))
      y = float((n2p1) - i)
      radius = sqrt(x**2.0 + y**2.0)
      IF (radius.LE.rad) THEN
         PUPIL(i,j) = (1.0,0.0)
      ELSE
         PUPIL(i,j) = (0.0,0.0)
      ENDIF
      IF ((abs(x).LE.aval).AND.(abs(y).LE.aval)) THEN
         GAUSSIAN(i,j) = DCMPLX(exp(-(x**2.0 +
            + y**2.0)/4))
      ELSE
         GAUSSIAN(i,j) = (0.0,0.0)
      ENDIF
   END
10 CONTINUE
CALL FFT2D(GAUSSIAN,TEMPDATA,F,fn,fwd,n)
CALL FFT2D(PUPIL,TEMPDATA,F,fn,fwd,n)
DO 20 i = 1, n
    DO 20 j = 1, n
        ASTEROID(i,j) = GAUSSIAN(i,j) * PUPIL(i,j)
    CONTINUE
CALL FFT2D(ASTEROID,TEMPDATA,F,ln,inv,n)
a = 10
a1 = 2.5
a2 = 2
a3 = 2.5
a4 = 2
a5 = 3.5
a6 = 3
a7 = 2.5
DO 30 i = 1, n
    DO 30 j = 1, n
        x = float(j - (n2p1))
        y = float((n2p1- i)
        x1 = x+6
        y1 = y-8
        xa1 = x1/a1
        ya1 = y1/a1
        IF ((abs(xa1).LE.aval).AND.(abs(ya1).LE.aval))
        THEN
            E1 = DCMPLX(a * exp(-(xa1**2.0 + ya1**2.0)))
        ELSE
            E1 = (0.0,0.0)
        ENDIF
        x2 = x-4
        y2 = y-6
        xa2 = x2/a2
        ya2 = y2/a2
        IF ((abs(xa2).LE.aval).AND.(abs(ya2).LE.aval))
        THEN
            E2 = DCMPLX(a * exp(-(xa2**2.0 + ya2**2.0)))
        ELSE
            E2 = (0.0,0.0)
        ENDIF
        x3 = x-10
        y3 = y-4
        xa3 = x3/a3
        ya3 = y3/a3
IF ((abs(xa3).LE.aval).AND.(abs(ya3).LE.aval))
    THEN
        E3 = DCMPLX(a * exp(-(xa3**2.0 + ya3**2.0)))
    ELSE
        E3 = (0.0,0.0)
    ENDIF

x4 = x+6
y4 = y+6
xa4 = x4/a4
ya4 = y4/a4
IF ((abs(xa4).LE.aval).AND.(abs(ya4).LE.aval))
    THEN
        E4 = DCMPLX(a * exp(-(xa4**2.0 + ya4**2.0)))
    ELSE
        E4 = (0.0,0.0)
    ENDIF

x5 = x+0
y5 = y-2
xa5 = x5/a5
ya5 = y5/a5
IF ((abs(xa5).LE.aval).AND.(abs(ya5).LE.aval))
    THEN
        E5 = DCMPLX(a * exp(-(xa5**2.0 + ya5**2.0)))
    ELSE
        E5 = (0.0,0.0)
    ENDIF

x6 = x+2
y6 = y+8
xa6 = x6/a6
ya6 = y6/a6
IF ((abs(xa6).LE.aval).AND.(abs(ya6).LE.aval))
    THEN
        E6 = DCMPLX(a * exp(-(xa6**2.0 + ya6**2.0)))
    ELSE
        E6 = (0.0,0.0)
    ENDIF

x7 = x-6
y7 = y+6
xa7 = x7/a7
ya7 = y7/a7
IF ((abs(xa7).LE.aval).AND.(abs(ya7).LE.aval))
+ THEN
  E7 = DCMPLX(a * exp(-(xa7**2.0 + ya7**2.0)))
ELSE
  E7 = (0,0,0.0)
ENDIF
ASTEROID(ij) = DCMPLX(DREAL(ASTEROID(ij) -
+ (E1 + E2 + E3 + E4 + E5 + E6 + E7))
IF (DREAL(ASTEROID(ij)).LT.0.0) ASTEROID(ij) =
+ (0,0,0.0)
30 CONTINUE
RETURN
END

SUBROUTINE Bistar(DATA,n)
  c THIS S/R CREATES A SIMULATED BINARY STAR WITH ONE STAR
  c LARGER THAN THE OTHER

COMPLEX*16 DATA(n,n)
n2p1 = n/2 + 1
DO 10 i = 1, n
  DO 10 j = 1, n
    x = float(j - n2p1)
    y = float(n2p1 - i)
    IF ((x.EQ.12.0).AND.(y.EQ.12.0)) THEN
      DATA(ij) = (2.0,0.0)
    ELSEIF ((x.EQ.-12.0).AND.(y.EQ.-12.0)) THEN
      DATA(ij) = (1.0,0.0)
    ELSE
      DATA(ij) = (0.0,0.0)
    ENDIF
  10 CONTINUE
RETURN
END
SUBROUTINE Centroid(DATA,TEMPDATA,n)

    c THIS S/R DETERMINES THE CENTROID OF THE DEGRADED IMAGE
    c AND THEN CENTROIDS THE IMAGE

    COMPLEX*16 DATA(n,n), TEMPDATA(n,n)
    REAL*8 xj, yi, xnum, ynum, xden, yden
    n2p1 = n/2 + 1
    DO 10 i = 1, n
       DO 10 j = 1, n
          xj = dfloat(j - n2p1)
          yi = dfloat(n2p1 - i)
          xnum = xnum + xj * ABS(DATA(i,j))
          ynum = ynum + yi * ABS(DATA(i,j))
          xden = xden + ABS(DATA(i,j))
          yden = yden + ABS(DATA(i,j))
    10 CONTINUE
    jxbar = idnint(xnum/xden)
    iybar = idnint(ynum/yden)
    DO 20 i = 1, n
       DO 20 j = 1, n
          ii = i - iybar
          jj = j + jxbar
          IF ((ii.GT.n).OR.(ii.LT.1).OR.(jj.GT.n).OR.
              (jj.LT.1)) THEN
             TEMPDATA(i,j) = (0.0,0.0)
          ELSE
             TEMPDATA(i,j) = DATA(ii,jj)
          ENDIF
    20 CONTINUE
    DO 30 i = 1, n
       DO 30 j = 1, n
          DATA(i,j) = TEMPDATA(i,j)
    30 CONTINUE
    RETURN
END
SUBROUTINE Combine(O,mod,PHASOR,n)

  c THIS S/R COMBINES THE MODULUS AND PHASOR OF THE
  c RECONSTRUCTED OBJECT

  COMPLEX*16 PHASOR(-n/2:n/2-1,-n/2:n/2-1),
  + O(-n/2:n/2-1,-n/2:n/2-1)
  REAL*8 mod(-n/2:n/2-1,-n/2:n/2-1)
  n2 = n/2
  n2m1 = n2 - 1
  DO 10 i = -n2, n2m1
    DO 10 j = -n2, n2m1
      O(i,j) = mod(i,j) * PHASOR(i,j)
  10 CONTINUE
  RETURN
END

SUBROUTINE Complexconj(DATA,n)

  c THIS S/R IS CALLED BY FFT2D AND DETERMINES THE COMPLEX
  c CONJUGATE OF THE 2-D ARRAY IN ORDER FOR THE ARRAY TO
  c BE INVERSE FFTed.

  COMPLEX*16 DATA(n,n)
  DO 10 i = 1, n
    DO 10 j = 1, n
      DATA(i,j) = DCONJG(DATA(i,j))
  10 CONTINUE
  RETURN
END
SUBROUTINE FFT(F,ln,n)

c THIS S/R IS USED BY THE 2-D FFT TWICE, FIRST FFT'ING THE

ROWS THEN THE COLUMNS OF THE 2-D ARRAY. THIS S/R WAS

ACQUIRED FROM "DIGITAL IMAGE PROCESSING" BY GONZALEZ

AND WINTZ.

COMPLEX*16 F(n),U,W,T
REAL*8 pi, one
one = 1.0D+00
pi = DACOS(-one)
nv2 = n/2
nm1 = n - 1
j = 1
DO 3 i = 1, nm1
   IF(i.GE.j) GOTO 1
   T = F(j)
   F(j) = F(i)
   F(i) = T
1  k = nv2
2  IF(k.GE.j) GOTO 3
   j = j - k
   k = k/2
   GOTO 2
3  j = j + k
DO 5 1 = 1, ln
   le = 2**1
   le1 = le/2
   U = (1.0,0.0)
   W = DCMPLX(DCOS(pi/le1),-DSIN(pi/le1))
   DO 5 j = 1, le1
      DO 4 i = j,n,le
         ip = i + le1
         T = F(ip) * U
         F(ip) = F(i) - T
4      F(i) = F(i) + T
5  U = U * W
RETURN
END
SUBROUTINE FFT2D(DATA,TEMPDATA,F,ln,dir,n)  

  c THIS S/R IS THE MAIN S/R AND PERFORMS A 2-D FFT ON AN  
  c ARRAY OF SIDE LENGTH 'n' WHERE 'ln' IS THE POWER OF 2.  
  c 'dir' IS EITHER +1 OR -1 WHETHER XFORMING OR INVERSE  
  c XFORMING RESPECTIVELY. 'TEMPDATA' IS A WORKING ARRAY  
  c FOR THE QUADRANT SWAPPING S/R. 'F' IS THE 1-D ARRAY USED  
  c BY THE 1-D FFT S/R. THIS S/R CALLS QUADSWAP, FFT, FOR BOTH  
  c FORWARD AND INVERSE FFT AND CALLS NORMFFT AND  
  c COMPLEXCONJ FOR INVERSE FFT's ONLY. THIS FFT NORMALIZES  
  c BY DIVIDING BY n**2 WHEN THE INVERSE FFT IS PERFORMED.  

  COMPLEX*16 DATA(n,n), F(n), TEMPDATA(n,n)  
  INTEGER dir  
  CALL Quadswap(DATA,TEMPDATA,n)  
  IF (dir.EQ.-1) THEN  
    CALL Complexconj(DATA,n)  
  ENDIF  
  DO 10 i = 1, n  
    DO 20 j = 1, n  
      F(j) = DATA(i,j)  
    END 20  
  DO 30 j = 1, n  
    DATA(i,j) = F(j)  
  CONTINUE 30  
 10 CONTINUE  
  DO 40 j = 1, n  
    DO 50 i = 1, n  
      F(i) = DATA(i,j)  
    END 50  
  DO 60 j = 1, n  
    DATA(i,j) = F(i)  
  END 60  
 40 CONTINUE  
  IF (dir.EQ.-1) THEN  
    CALL Complexconj(DATA,n)  
    CALL NormFFT(DATA,n)  
  ENDIF  
  CALL Quadswap(DATA,TEMPDATA,n)  
  RETURN  
END
SUBROUTINE Modulus(I,I1,IDATA,ISDATA,mod,icounter,  
+ nframes,nphoton,n)

c THIS S/R DETERMINES THE MODULUS OF THE DEGRADED IMAGE
+ AND THE POINT SOURCE AND NORMALIZES THEM BY DIVIDING
+ BY THEIR RESPECTIVE DC VALUES.

COMPLEX*16 IDATA(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4),
+ ISDATA(-n/2:n/2-1,-n/2:n/2-1), DC, DCS,
+ I(-n/2:n/2-1,-n/2:n/2-1),
+ IS(-n/2:n/2-1,-n/2:n/2-1)
REAL*8 mod(-n/2:n/2-1,-n/2:n/2-1)

k = 0
n2 = n/2
n2m1 = n2 - 1
DO 10 ii = -n2, n2m1
  DO 10 jj = -n2, n2m1
    IDATA(ii, jj, k, k) = IDATA(ii, jj, k, k) + ((I(ii, jj) *  
+ DCONJG(I(ii, jj))) - I(k, k))
    ISDATA(ii, jj) = ISDATA(ii, jj) + ((IS(ii, jj) *  
+ DCONJG(IS(ii, jj))) - IS(k, k))
  10 CONTINUE
IF (icounter.EQ.nframes) THEN
  DC = IDATA(k, k, k, k)
  DCS = ISDATA(k, k)
  DO 20 ii = -n2, n2m1
    DO 20 jj = -n2, n2m1
      IDATA(ii, jj, k, k) = IDATA(ii, jj, k, k)/DC
      ISDATA(ii, jj) = ISDATA(ii, jj)/DCS
      mod(ii, jj) = nphoton *  
      + dsqrt(ABS(IDATA(ii, jj, k, k)/ISDATA(ii, jj)))
      IF (mod(ii, jj).GT.dfloat(nphoton))  
      + mod(ii, jj) = dfloat(nphoton)
  20 CONTINUE
ENDIF
RETURN
END
SUBROUTINE NormFFT(DATA,n)
c THIS S/R IS CALLED BY FFT2D FOR INVERSE FOURIER XFORMS
AND NORMALIZES THE XFORM BY DIVIDING BY n**2.
COMPLEX*16 DATA(n,n)
nsqrd = n * n
DO 10 i = 1, n
   DO 10 j = 1, n
      DATA(i,j) = DATA(i,j)/nsqrd
10 CONTINUE
RETURN
END

SUBROUTINE Photons(DATA,nphoton,n)
c THIS S/R NORMALIZES THE DATA ARRAY TO MAKE THE DC
VALUE EQUAL TO THE NUMBER OF PHOTONS IN THE OBJECT.
COMPLEX*16 DATA(n,n)
REAL*8 photonum
ic = n/2 + 1
photonum = dfloat(nphoton)/DREAL(DATA(ic,ic))
DO 10 i = 1, n
   DO 10 j = 1, n
      DATA(i,j) = photonum * DATA(i,j)
10 CONTINUE
RETURN
END

SUBROUTINE Quadswap(DATA,TEMPDATA,n)
c THIS S/R IS CALLED BY FFT2D AND SWAPS QUADRANTS OF DATA
ARRAY USING TEMPDATA AS A WORKING ARRAY.
COMPLEX*16 DATA(n,n), TEMPDATA(n,n)
n2 = n/2
DO 10 i = 1, n/2
   DO 10 j = 1, n/2
      TEMPDATA(i,j) = DATA(i+n2,j+n2)
      TEMPDATA(i+n2,j) = DATA(i,j+n2)
      TEMPDATA(i,j+n2) = DATA(i+n2,j)
10 CONTINUE

TEMPDATA(i+n2j+n2) = DATA(ij)

10 CONTINUE
DO 20 i = 1, n
   DO 20 j = 1, n
      DATA(i,j) = TEMPDATA(i,j)
20 CONTINUE
RETURN
END

SUBROUTINE SNRcalc(KTsnr,rsnr,nyquist,n)
  
  c THIS S/R CALCULATES THE AVERAGE SNR AS A FUNCTION OF 
  c RADIUS
  
  REAL*8 KTsnr(n,n), rsnr(n/2)
  INTEGER r
  n2 = n/2
  n2p1 = n2 + 1
  DO 10 r = 1, nyquist
     nsnr = 0
     DO 20 i = 1, n
        DO 20 j = 1, n
           x = float(i - (n2p1))
           y = float(j - (n2p1))
           radius = sqrt(x**2.0 + y**2.0)
           IF ((radius.GT.float(r-1)).AND.
               + (radius.LE.float(r))) THEN
              nsnr = nsnr + 1
              rsnr(r) = rsnr(r) + KTsnr(i,j)
           ENDIF
20 CONTINUE
     rsnr(r) = rsnr(r)/nsnr
10 CONTINUE
RETURN
END
SUBROUTINE Star(DATA,n)

c THIS S/R CREATES A SIMULATED POINT SOURCE OR STAR

COMPLEX*16 DATA(n,n)

n2p1 = n/2 + 1

DO 10 i = 1, n
   DO 10 j = 1, n
      x = float(j - n2p1)
      y = float(n2p1 - i)
      IF ((x.EQ.0.0).AND.(y.EQ.0.0)) THEN
         DATA(i,j) = (1.0,0.0)
      ELSE
         DATA(i,j) = (0.0,0.0)
      ENDIF
   10 CONTINUE

RETURN

END
SUBROUTINE Writefile(DATA1,data2,file1,file2, + nyquist,n)

THIS S/R WRITES THE DATA TO A FILE IN THE FORMAT REQUIRED
BY A PROGRAM WHICH PRODUCES A CONTOUR PLOT OF THE
IMAGE

COMPLEX*16 DATA1(n,n)
REAL*8 data2(n/2-1)
REAL x, lambda, R0, pi
INTEGER r
CHARACTER*16 file1, file2
COMMON /VARS2/DIAM,OBSCUR,LAMBDA,R0,SECDIM
     PIXSCALE,TPFDIM,FILENAME

OPEN (UNIT=30,FILE=file1,STATUS='NEW')
OPEN (UNIT=40,FILE=file2,STATUS='NEW')
pi = acos(-1.0E+00)
DO 10 i = 1, n
   DO 10 j = 1, n
      WRITE(30,* ) DATA1(i,j)
10 CONTINUE
DO 20 r = 1, nyquist
   x = r * (lambda/pixscale) * (180/pi) * 3600.0
   WRITE(40,* ) x, data2(r)
20 CONTINUE
RETURN
END
APPENDIX E. SPECIFIC KNOX-THOMPSON SUBROUTINES

THE FOLLOWING SUBROUTINES WERE GENERATED BY THE
THESIS AUTHOR AND ARE USED BY THE KNOX-THOMPSON
PROGRAM ONLY.

SUBROUTINE LIST

SUBROUTINE Initialize(OBJDATA,GAUSSIAN,PUPIL,F,
+ TEMPDATA,file1,file2,fwd,inv,
+ ln,nframes,nphoton,nyquist,
+ offset,cent,n)

THIS S/R Initializes some of the program variables by
querying the user for input. It also creates the
object desired by the user by calling either asteroid,
bistar, or star s/r.

INTEGER offset
CHARACTER*16 file1, file2
CHARACTER*1 cent, object
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER OFFSET (<= 4 AND <= #'
WRITE(*,*)'PIXELS/R0):'
READ(*,*)offset
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER NUMBER OF SNAPSHOTs:'
READ(*,*)nframes
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER NUMBER OF OBJECT PHOTONS PER'
WRITE(*,*)'SNAPSHOT:'
READ(*,*)nphoton
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER NYQUIST VALUE:'
READ(*,*)nyquist

131
WRITE(*,*),' ENTER OBJECT TYPE: "A" FOR ASTEROID,' WRITE(*,*) 'B" FOR BINARY STAR, OR "S" FOR STAR.' READ(*,40) object
IF ((object.EQ.'a').OR.(object.EQ.'A')) THEN
   CALL AST(OBJDATA,GAUSSIAN,PUPIL,F,TEMPDATA,+ fwd,inv,ln,n)
ELSEIF ((object.EQ.'b').OR.(object.EQ.'B')) THEN
   CALL Bistar(OBJDATA,n)
ELSEIF ((object.EQ.'s').OR.(object.EQ.'S')) THEN
   CALL Star(OBJDATA,n)
ELSE
   WRITE(*,*)' INCORRECT, REENTER'
ENDIF
GOTO 10
ENDIF
WRITE(*,*) ' WOULDL YOU LIKE THE DATA CENTROIDED? (Y/N)' READ(*,40) cent
IF ((cent.NE.'y').AND.(cent.NE.'Y').AND.+ (cent.NE.'n').AND.(cent.NE.'N')) THEN
   WRITE(*,*)' INCORRECT, REENTER'
   GOTO 20
ENDIF
WRITE(*,*) ' ENTER KT OUTPUT FILE NAME (16 CHAR' WRITE(*,*)' MAX):'
READ(*,30) file1
WRITE(*,*) ' ENTER KT SNR FILE NAME (16 CHAR' WRITE(*,*)' MAX):'
READ(*,30) file2
30 FORMAT(A16)
40 FORMAT(A1)
RETURN
END
SUBROUTINE KTr Recon(KTPHASOR, IKT, KTsnr, offset,
  + xvarrktxvarikt, icounter,
  + nframes, nyquist, n)

  THIS S/R IS THE HEART OF THE KT RECONSTRUCTION PROGRAM.

COMPLEX*16 KTPHASOR(-n/2:n/2-1,-n/2:n/2-1), CTEMP,
  + IKT(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4),
  + I(-n/2:n/2-1,-n/2:n/2-1), PTEMP1, PTEMP2
REAL*8 KTsnr(-n/2:n/2-1,-n/2:n/2-1), sigmar, sigmai,
  + xvarrktxvarikt(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4), snr,
  + xvarikt(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4), sigma,
INTEGER r, di, dj, offset
k = 0
KTPHASOR(k,k) = (1.0,0.0)
DO 10 r = 1, nyquist
  DO 10 ii = 0, r
    IF (ii.EQ.0) THEN
      lim = 0
    ELSE
      lim = -r
    ENDIF
    DO10b = lii, r
      rad = sqrt(float(ii)**2.0 + float(jj)**2.0)
      IF ((rad.LE.float(r)).AND.(rad.GT.float(r-1))) THEN
        IF (icounter.EQ.nframes) nsnr = 0
        DO 20 di = 0, offset
          DO 20 dj = -offset, offset
            drad = sqrt(float(di)**2.0 + float(dj)**2.0)
            IF ((drad.LE.float(r)).AND.+
                (drad.LE.float(offset))) THEN
              idi = ii - di
              jdj = jj - dj
              radd = sqrt(float(idi)**2.0 + float(jdj)**2.0)
IF (radd.LE.float(r-1)) THEN
  iHC = -ii
  jHC = -jj
  IF (I(idijdj).EQ.(0.0,0.0))
    I(idijdj) = (1.0,0.0)
  ELSE IF (I(ii,ji).EQ.(0.0,0.0))
    I(ii,ji) = (1.0,0.0)
  ENDIF
  CTEMP = (I(idijdj) * DCONJG(I(ii,ji))) - DCONJG(I(di,di))
  IKT(idijdj,di,di) = IKT(idijdj,di,di) + CTEMP
  xvarrkt(idijdj,di,di) =
    xvarrkt(idijdj,di,di) +
    (DREAL(CTEMP))**2.0
  xvarikt(idijdj,di,di) =
    xvarikt(idijdj,di,di) +
    (DIMAG(CTEMP))**2.0
  IF (icounter.EQ.nframes) THEN
    nsnr = nsnr + 1
    sigmar = dabs((xvarrkt(idijdj,di,di) -
      (DREAL(IKT(idijdj,di,di))))**2.0)/
      dfloat(nframes)/dfloat(nframes - 1))
    sigmai = dabs((xvarikt(idijdj,di,di) -
      (DIMAG(IKT(idijdj,di,di))))**2.0)/
      dfloat(nframes)/dfloat(nframes - 1))
    sigma = dsqrt(sigmar + sigmai)
    snr = (((ABS(IKT(idijdj,di,di)))/
      dfloat(nframes))/sigma) *
      dsqrt(dfloat(nframes))
    KTSnr(ii,ji) = KTSnr(ii,ji) + snr
    PTEMP1 = IKT(idijdj,di,di)/
      ABS(IKT(idijdj,di,di))
    PTEMP2 = KTPHASOR(idijdj)
    KTPHASOR(ii,ji) = KTPHASOR(ii,ji) +
      ((snr**2.0) * (DCONJG(PTEMP1/PTEMP2)))
  ENDIF
ENDIF
ENDIF
ENDIF
20  CONTINUE
IF (icounter.EQ.nframes) THEN
    KTsnr(ii, jj) = KTsnr(ii, jj)/nsnr
    KTsnr(iHC, jHC) = KTsnr(ii, jj)/nsnr
    KTPHASOR(ii, jj) = KTPHASOR(ii, jj)/
                      + ABS(KTPHASOR(ii, jj))
                      KTPHASOR(iHC, jHC) = DCONJG(KTPHASOR(ii, jj))
    ENDIF
    ENDIF
10 CONTINUE
RETURN
END
APPENDIX F. SPECIFIC TRIPLE-CORRELATION SUBROUTINES

THE FOLLOWING SUBROUTINES WERE GENERATED BY THE
THESIS AUTHOR AND ARE USED BY THE TRIPLE-CORRELATION
PROGRAM ONLY.

SUBROUTINE LIST

SUBROUTINE BSrecon(BSPHASOR,I,IBS,IDBS,BSnr,offset, + xvarrbs,xvaribs,icounter,nframes, + nyquist,nphoton,n)

THIS S/R IS THE HEART OF THE BS RECONSTRUCTION PROGRAM.
IT DETERMINES THE VARIANCE-WEIGHTED,
NOISE-BIAS-CORRECTED BISPECTRUM OF THE DEGRADED IMAGE
AND RECOVERS THE PHASEORS FROM THE AVERAGE OF
THIS BISPECTRUM THIS S/R IS CALLED ONCE FOR EACH
OBJECT/POINT SOURCE SNAP SHOT TO DETERMINE A RUNNING
AVERAGE BISPECTRUM AND WITH THE LAST SNAP SHOT, THE
PHASOR IS RECONSTRUCTED.

INTEGER r, di, dj, offset
k = 0
loop = 1
BSPHASOR(k,k) = (1.0,0,0)
DO 10 r = 1, nyquist
DO 10 ii = 0, r

136
IF (ii.EQ.0) THEN
  lim = 0
ELSE
  lim = -r
ENDIF
DO 10 jj = lim, r
  rad = sqrt(float(ii)**2.0 + float(jj)**2.0)
  IF (((rad.LE.float(r)).AND.(rad.GT.float(r-1))) THEN
    IF (icounter.EQ.nframes) nsnr = 0
    DO 20 di = 0, offset
      DO 20 dj = -offset, offset
        IF (loop.EQ.1) THEN
          IDBS(di,dj) = IDBS(di,dj) + I(di,dj)
        ENDIF
        drad = sqrt(float(di)**2.0 + float(dj)**2.0)
        IF ((drad.LE.float(r)).AND.(drad.LE.float(offset))) THEN
          idi = ii - di
          jdj = jj - dj
          radd = sqrt(float(idi)**2.0 + float(jdj)**2.0)
          IF (radd.LE.float(r-1)) THEN
            iHC = -ii
            jHC = -jj
            IF (I(idi,jdj).EQ.(0.0,0.0)) THEN
              I(idi,jdj) = (1.0,0.0)
              I(ii,jj) = (1.0,0.0)
            ELSE
              CTEMP = I(idi,jdj) * DCONJG(I(ii,jj)) * 
              + I(di,dj) - (ABS(I(idi,jdj)))**2.0 - 
              + (ABS(I(ii,jj)))**2.0 - 
              + (ABS(I(idi,jdj)))**2.0 + (2.0 * nphoton)
              IBS(idi,jdj,di,dj) = IBS(idi,jdj,di,dj) + 
              + CTEMP
              xvarrbs(idi,jdj,di,dj) = 
              + xvarrbs(idi,jdj,di,dj) + 
              + (DREAL(CTEMP))**2.0
              xvaribs(idi,jdj,di,dj) = 
              + xvaribs(idi,jdj,di,dj) + 
              + (DIMAG(CTEMP))**2.0
  ENDIF
IF (icounter.EQ.nframes) THEN
  nsnr = nsnr + 1
  sigmar = dabs((xvarrbs(idi,jdj,di,dj) -
  ((DREAL(IBS(idi,jdj,di,dj)))**2.0)/
  dfloat(nframes))/dfloat(nframes - 1))
  sigmai = dabs((xvaribs(idi,jdj,di,dj) -
  ((DIMAG(IBS(idi,jdj,di,dj)))**2.0)/
  dfloat(nframes))/dfloat(nframes - 1))
  sigma = dsqrt(sigmar + sigmai)
  snr = (((ABS(IBS(idi,jdj,di,dj)))/*
  dfloat(nframes))/sigma) *
  dsqrt(dfloat(nframes))
  BSsnr(ii,jj) = BSsnr(ii,jj) + snr
  PTEMP1 = IBS(idi,jdj,di,dj)/
  ABS(IBS(idi,jdj,di,dj))
  PTEMP2 = BSPHASOR(idi,jdj)
  * IDBS(di,dj)/ABS(IDBS(di,dj))
  BSPHASOR(ii,jj) = BSPHASOR(ii,jj) +
  ((snr**2.0) * (DCONJG(PTEMP1/PTEMP2)))
ENDIF
ENDIF
ENDIF
20 CONTINUE
IF (loop.EQ.1) loop = 2
IF (icounter.EQ.nframes) THEN
  BSsnr(ii,jj) = BSsnr(ii,jj)/nsnr
  BSsnr(iHC,jHC) = BSsnr(ii,jj)/nsnr
  BSPHASOR(ii,jj) = BSPHASOR(ii,jj)/
  ABS(BSPHASOR(ii,jj))
  BSPHASOR(iHC,jHC) = DCONJG(BSPHASOR(ii,jj))
ENDIF
ENDIF
10 CONTINUE
RETURN
END
SUBROUTINE Initialize(OBJDATA,GAUSSIAN,PUPIL,F,
+         TEMPDATA,file1,file2,fwd,inv,ln,
+         nframes,nphoton,nyquist,offset,
+         cent,n)

c THIS S/R INITIALIZES SOME OF THE PROGRAM VARIABLES BY
  QUEARYING THE USER FOR INPUT. IT ALSO CREATES THE
  OBJECT DESIRED BY THE USER BY CALLING EITHER ASTEROID,
  BISTAR, OR STAR S/R.

INTEGER offset
CHARACTER*16 file1, file2
CHARACTER*1 cent, object
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER OFFSET (<= 4):'
READ(*,*)offset
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER NUMBER OF SNAPSHOTS:'
READ(*,*)nframes
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER NUMBER OF OBJECT PHOTONS PER'
WRITE(*,*)'SNAPSHOT:'
READ(*,*)nphoton
WRITE(*,*)
WRITE(*,*)'ENTER INTEGER NYQUIST VALUE:'
READ(*,*)nyquist
10 WRITE(*,*)
WRITE(*,*)'ENTER OBJECT TYPE: "A" FOR ASTEROID,'
WRITE(*,*)'"B" FOR BINARY STAR, OR "S" FOR STAR.'
READ(*,40) object
IF ((object.EQ.'a').OR.(object.EQ.'A')) THEN
  CALL AST(OBJDATA,GAUSSIAN,PUPIL,F,TEMPDATA,
+         fwd,inv,ln,n)
ELSEIF ((object.EQ.'b').OR.(object.EQ.'B')) THEN
  CALL Bistar(OBJDATA,n)
ELSEIF ((object.EQ.'s').OR.(object.EQ.'S')) THEN
  CALL Star(OBJDATA,n)
ELSE
  WRITE(*,*)'INCORRECT, REENTER'
  GOTO 10
ENDIF

139
20  WRITE(*,*)''
    WRITE(*,*)'WOULD YOU LIKE THE DATA CENTROIDED? (Y/N)'
    READ(*,40) cent
    IF ((cent.NE.'y').AND.(cent.NE.'Y').AND. 
      +  (cent.NE.'n').AND.(cent.NE.'N')) THEN
      WRITE(*,*)'INCORRECT, REENTER'
      GOTO 20
    ENDIF
    WRITE(*,*)''
    WRITE(*,*)'ENTER TC OUTPUT FILE NAME (16 CHAR MAX):'
    READ(*,30) file1
    WRITE(*,*)''
    WRITE(*,*)'ENTER TC SNR FILE NAME (16 CHAR MAX):'
    READ(*,30) file2
30  FORMAT(A16)
40  FORMAT(A1)
RETURN
END
APPENDIX G. ATMOSPHERIC DEGRADATION SUBROUTINES

THE FOLLOWING SUBROUTINES WERE PROVIDED BY CAPTAIN CHUCK MATSON AND MS. IDA DRUNZER OF WL/ARCI AND MODIFIED FOR USE WITH THE KNOX-THOMPSON AND TRIPLE-CORRELATION IMAGE RECONSTRUCTION PROGRAMS. THESE SUBROUTINES ARE REQUIRED FOR BOTH PROGRAMS.

SUBROUTINE LIST

SUBROUTINE PHSUB(SEIMG,F,TEMPDATA,OBJARR,
+ icounter,nframes,nyquist,
+ fwd,inv,ln,n,mseed)

FUNCTION: Creates an image of an object located in space by filtering a single snapshot of an object with a phase screen.

ORDER OF EVENTS TO OBTAIN IMAGE:

- Create an object
- Create Gaussian Array
- Expose array to what the atmosphere will do to the object (Correlation Filter Function)
- Expose array to the stipulations of the telescope
- Autocorrelate the object (incoherent transfer function)
- Multiply the object by the phase screen to obtain the Image

DICTIONARY:

DIAM - Telescope pupil diameter
ICOUNTER - Number of user chosen iterations of phase screens to process
ISEED - Seed to be used for random number generator
LAMBDA - Center wavelength
MU - Mean
c PIXSCALE - Corresponds to each pixel in cycles/m

c R0 - Variable characterizing the strength of atmospheric turbulence with respect to the optical system

c TPFDIM - Dimension for pupil of telescope

c PCOUNT - The actual number of photons per frame

c NFRAMES - The number of frames to process

c NPHOTON - The number of photons the user wants per frame

c NKL - The number of Karhunen-Loeve functions projected off and added back in the phase map.

PARAMETER (IDIM = 64, LDIM = IDIM*IDIM)
PARAMETER (IDIM2 = 64, LDIM2 = IDIM2*IDIM2)
PARAMETER (IDIM1 = .7071*IDIM2)
PARAMETER (WDM = 5*IDIM2, WDM2 = 5*IDIM2/2)
PARAMETER (N1 = 11, N2 = (N1/2) + 1,
+ N3 = (N1 - 1)*(N1 + 2)/2)
PARAMETER (N4 = (N1 - 1)*(N1 + 3)/4)
PARAMETER (NKL = 20)

CHARACTER*16 FILENAME
CHARACTER*1 TILT

COMPLEX*16 PHAMAP(LDIM2), PS1(LDIM), IMAGE(LDIM),
+ OBJARR(LDIM), PS2(LDIM2), SEIMG(LDIM)

REAL POIARRAY(LDIM), RVAR, MULT1, MULT2, LAMBDA,
+ PIXSCALE, ZERO, SIZE, DIAM, TPFDIM, RO, INNERSCALE,
+ CORR(LDIM2), SECDIM, OUTERSCALE, OBSCUR, ARRAY(LDIM)

INTEGER PCOUNT

DIMENSION ZKLMAP(IDIM2, IDIM2, NKL), LINPTR(N3), BVAL(N3)

COMMON /VARS2/DIAM, OBSCUR, LAMBDA, RO, SECDIM,
+ PIXSCALE, TPFDIM, FILENAME

Initialize variables

CMINUS1 = (-1.E0, 0.E0)
ZERO = 0.E0
MU = 0.E0
SIGMA = 1.E0
PI = DACOS(-1.0D+00)
JDIM = IDIM
JDIM2 = IDIM2
IC = IDIM/2+1
INNERSCALE = .0001
OUTERSCALE = 10000.E0
MULT1 = 5.92/INNERSCALE
MULT2 = (2.0*PI)/OUTERSCALE

c Only do certain subroutines the first time calling this subroutine (when wanting
c several snapshots).

CONTINUE IF((ICOUNTER.EQ.1).AND.(MSEED.EQ.1)) THEN
FILENAME = 'phvar2.d'
OPEN(10,FILE=FILENAME,STATUS='OLD',ERR=500)
READ(10,*)DIAM,OBSCUR,LAMBDA,R0
CLOSE(UNIT=10)

c Call the subroutine which prompts the user for all names and variables that the
c subroutine will need in order to run.

CALL PARAM2(IDIM, NYQUIST, TILT, ISEED)

c Generate a filter function to be used as a multiplier on a complex gaussian
array to be stored in array CORR. However, only call the filter function on
multiple runs if there are changing values for the width of the object and R0.

CALIFILT2(RARRAY,CORRLDIM2,IDIM2,MULT1,MULT2,PIXSCALE, +
ZERO,RO)

c Calculate the simulated size of the telescope (tpfdim). Compare it with the
dimension that is know (size). If size is less than tpfdim, there will be an error
due to the array being too small to store the data from the fourier transform,
and thus, data will be truncated.

SIZE = (IDIM/2)-1
IF(SIZE.LT.TPFDIM) THEN
   WRITE(*,*)'ERROR. IMAGE ARRAY SIZE MUST BE 512x512.'
   RETURN
   IF(SIZE.LE.(.5*TPFDIM)) THEN
      WRITE(*,*)'ERROR. IMAGE ARRAY SIZE MUST BE 512x512.'
      RETURN
   END
   CONTINUE
END
ELSE
    WRITE(*,*)'ERROR. IMAGE ARRAY SIZE MUST BE 256x256.'
ENDIF
WRITE(*,*) ''
GOTO 400
ENDIF

Call BGSCREEN subroutine first time through phase screen. The BGSCREEN subroutine manipulates the phase map to result in a phase map with proper statistics.

CALL BGSCR2(BVAL,ZKLMAP,LINPTR,IDIM2)

End the if statement if first time through the subroutine.

mseed = mseed + 1

ENDIF

Get Gaussian numbers and then multiply these two numbers by the correlation filter function to obtain the phase map (PHAMAP).

DO 20 J = 1,LDIM2
    CALL GAUSS(MU,SIGMA,R1,R2,PI,ISEED)
    PHAMAP(J) = DCMPLX(R1*CORR(J),R2*CORR(J))
20 CONTINUE

Get phase screen which is in the frequency domain. Next perform the inverse FFT on array.

CALL FFT2D(PHAMAP,TEMPDATA,F,ln,invn)

DO 25 J = 1, LDIM
    PHAMAP(J) = PHAMAP(J)*(DFLOAT(LDIM))
25 CONTINUE
CALL PROJ(PHAMAP,BVAL,ZKLMAP,LINPTR,iseed,
    + NKL,R0,N1,N2,N3,N4,IDIM2,P1,PIXSCALE,TILT)
c Calculate $ae^{i \text{realization of phase screen}}$. Take cosine and sine of every
c real element in the array and put into the phase screen array.

DO 60 J = 1,LDIM2
   PS2(J) = DCMPLX(DCOS(DREAL(PHAMAP(J))),
             + DSIN(DREAL(PHAMAP(J))))
60 CONTINUE

Performs the Telescope Pupil Function (TPF) on the phase map to obtain the
coherent transfer function. First, call this routine to cut out chunk of phase
c screen based on Telescope Pupil Function of $D/LAMBDAt$ be stored in array
c PS1.

CALL TPF2A(PS1,PS2,IDIM,IDIM2,TPFDIM,SECDIM)

c Calculate the incoherent transfer function by multiplying phase screen by the
c auto correlation filter function. Begin by taking the Fourier Transform of the
c phase screen.

CALL FFT2D(PS1,TEMPDATA,F,ln,fwd,n)

c Take the magnitude squared of the array and put the data into the real part of
c the phase screen.

DO 120 J = 1,LDIM
   PS1(J) = DCMPLX(DREAL(DCONJG(PS1(J))*PS1(J)),0.e0)
120 CONTINUE

CALL FFT2D(PS1,TEMPDATA,F,ln,inv,n)

c Divide the phase screen by the value at dc in order to normalize the array by
c that value (dcval).

   TEMP = DREAL(PS1(IC+(IC-1)*IDIM))
DO 125 J = 1,LDIM
   PS1(J) = PS1(J)/TEMP
125 CONTINUE
Incoherent transfer function is completed. Now get the image in the fourier domain by multiplying the object and the phase screen to get the image.

```fortran
DO 160 J = 1, LDIM
  IMAGE(J) = (PS1(J) * OBJARR(J))
160 CONTINUE

CALL FFT2D(IMAGE, TEMPDATA, F, ln, inv, n)
```

Call the function, poisson, which will return an integer value that is a random deviate drawn from a Poisson distribution of mean equal to image value, using another real function, uniform, as a source of uniform random deviates.

```fortran
PCOUNT = ZERO

DO 230 J = 1, LDIM
  RVAR = DREAL(IMAGE(J))
  IF(RVAR,LE.0.0) THEN
    POIARRAY(J) = ZERO
  ELSE
    ITEMP = POISSON(RVAR, ISEED)
    POIARRAY(J) = DFLOAT(ITEMP)
    PCOUNT = PCOUNT + ITEMP
  ENDIF
  SEIMG(J) = DCMPLX(POIARRAY(J), ZERO)
230 CONTINUE

300 RETURN
400 WRITE(*,*)
   WRITE(*,*) 'STopping program due to error'
   STOP
```

Error Statements

```fortran
500 CONTINUE
   WRITE(*,*)
   WRITE(*,*) 'Error, this file does not exist. Reenter.'
   WRITE(*,*)
600 GOTO 11
800 CONTINUE
END
```
SUBROUTINE BGSCR2(bval, zklmap, linptr, nsid)

This program does the things that result in a phase screen with proper statistics

parameter(lu = 12, lu4 = 18)
parameter(nr = 100, nkl = 20)
PARAMETER (N1 = 11, N2 = (N1/2) + 1,
+    N3 = (N1 - 1)*(N1 + 2)/2)
PARAMETER (N4 = (N1 - 1)*(N1 + 3)/4)
real bval(n3), bvec(n2, n3), rmap(nr, n4), nsid1
dimension zklmap(nsid, nsid, nkl)
dimension ival(n3, 3), ir(n3), linptr(n3)
character*16 fname, noname
nsid1 = .7071*(nsid - 1)
fname = 'eigen.d'
noname = 'k1rad.d'

Dictionary

bval = contains eigenvalues of covariance matrix
bvec = contains eigenvectors of covariance matrix
ir = pointers
ival = pointers
linptr = pointers
n1 = number of azimuthal orders to be used in the Zernike functions
nkl = number of Karhunen-Loeve functions projected off and added
    back
nr = number of points in rmap (100 is more than enough)
nsid = dimension of side of screen
rmap = stored radial cut of Karhunen-Loeve functions, computed in
    michelin
zklmap = stored Karhunen-Loeve functions

open(lu, file = fname, form = 'formatted', status = 'old')
rewind(lu)
read(lu, *) linptr
read(lu, *) bval
read(lu, *) ival
do 95 i = 1, n2
    do 90 j = 1, n3
        read(lu, *) bvec(i, j)
    90 continue
95 continue
SUBROUTINE FADD(bmap,zklmap,iq,nsid,nkl,sum)

   bmap is the phase screen

complex*16 bmap(nsid,nsid),sum
dimension zklmap(nsid,nsid,nkl)
data pi/3.14159265358979/

   do 10 i = 1,nsid
   do 20 j = 1,nsid
      bmap(i,j) = bmap(i,j) + sum*zklmap(i,j,iq)
   20 continue
   10 continue
   return
end

SUBROUTINE FILT2(RARRAY,CORR,LDIM2,IDIM2,MULT1,
+ MULT2,PIXSCALE,ZERO,R0)

   Subroutine Function: This function generates an array representing the effects
   of what the atmosphere will do to an object when it passes through it.

REAL RARRAY(LDIM2),CORR(LDIM2),MULT1,MULT2,RCONST,
+ PIXSCALE,ZERO,R0,FCONST,R,ARRAYW
INTEGER JOFFSET
Calculate the radial average of an input image.

ARRAYS:  
CORR(LDIM2) - idim2 x idim2 array
RARRAY(IDIM2) - this is a small work array
that needs to be at least
IDIM2 in size
other variables:  
IDIM2 - x dimension1
RCONST,RCONST2 - real constants

c  load CORR with X (J) indicies squared

FCONST = .1517  
DO 10 J=1,IDIM2  
  JOFFSET=(J-1)*IDIM2  
  R=FLOAT( J-(IDIM2/2+1) )  
  RCONST=R*R  
  DO 15 I=1,IDIM2  
     CORR(I+JOFFSET) = RCONST
15 CONTINUE
10 CONTINUE

c  Add to L1, Y (I) indicies squared (one column of Y indicies stored in L2)

DO 20 I=1,IDIM2  
  R=FLOAT( I-(IDIM2/2+1) )  
  RARRAY(I) = R*R
20 CONTINUE
DO 30 J=1,IDIM2  
  JOFFSET=(J-1)*IDIM2  
  DO 40 I = 1,IDIM2  
     CORR(JOFFSET + I) = RARRAY(I) + CORR(JOFFSET + I)
40 CONTINUE
30 CONTINUE

c  Move the scaling array to another array in order to do calculations for the filter

c  function

DO 50 I = 1,LDIM2  
  RARRAY(I) = CORR(I)
50 CONTINUE
Multiply the scaling array by the correlation filter function

\[
\text{RCONST} = \left(-11.0/12.0\right) \\
\text{DO } 60 \text{ I } = \text{1,LDIM2} \\
\quad \text{RARRAY(I) } = \left(\text{RARRAY(I)} + (\text{MULT2} \times \text{MULT2})\right)^*\text{RCONST} \\
60 \text{ CONTINUE} \\
\text{ARRAYW } = \text{PIXSCALE} \times \text{IDIM2} \\
\text{RCONST } = \left(\text{R0/ARRAYW}\right)^*\left(-5.0/6.0\right) \\
\text{DO } 70 \text{ I } = \text{1,LDIM2} \\
\quad \text{RARRAY(I) } = \left(\text{RARRAY(I)} \times \text{RCONST}\right) \times \text{FCONST} \\
70 \text{ CONTINUE} \\
\]

Set the middle point in array to zero to normalize the array by that value

\[
\text{RARRAY(IDIM2*(IDIM2/2)+(IDIM2/2+1)) } = \text{ZERO} \\
\]

Calculate second part of correlation filter function and multiply it by the first part to complete the filter array.

\[
\text{RCONST } = -2.0 *(\text{MULT1} \times \text{MULT1}) \\
\text{DO } 90 \text{ I } = \text{1,LDIM2} \\
\quad \text{CORR(I) } = \left(\text{EXP(CORR(I)/RCONST)}\right) \times \text{RARRAY(I)} \\
90 \text{ CONTINUE} \\
\]

SUBROUTINE GAUSS(MU,SIGMA,RNUM1,RNUM2,PI,iseed)

PURPOSE:

GET GAUSSIAN DISTRIBUTED RANDOM NUMBER WITH MEAN MU AND STANDARD DEVIATION SIGMA

REAL MU,SIGMA,Y1,Y2,RNUM1,RNUM2,PI,TWOPI

TWOPI = 2.0*PI

Y1= uniform(iseed)
Y2= uniform(iseed)
c Calculate first random number

RNUM1 = MU + SIGMA*SQR(-2.0*ALOG(Y1))*COS(TWOPI*Y2)

c Calculate the second random number

RNUM2 = MU + SIGMA*SQR(-2.0*ALOG(Y1))*SIN(TWOPI*Y2)

RETURN
END

SUBROUTINE INPROD(bmap,zklmap,iq,nsid,nkl,zap)

complex*16 bmap(nsid,nsid)
complex*16 zap
dimension zklmap(nsid,nsid,nkl)

zap = (0.0,0.0)
area = 0.0
do 10 i = 1,nsid
   do 20 j = 1,nsid
      zap = zap + bmap(i,j)*zklmap(i,j,iq)
      area = area + ( zklmap(i,j,iq) )**2
   20 continue
10 continue
zap = zap/area
return
end
SUBROUTINE KELP(zklmap,rmap,ir,ival,linptr, 
+ n3,n4,nsid,nr,nkl)

c zklmap is the map of the k-l functions, sigh

dimension rmap(nr,n4),ir(n3)
dimension zklmap(nsid,nsid,nkl)
dimension ival(n3,3),linptr(n3)
data pi/3.14159265358979/
dcent = nsid/2 + 1
dx = 1.0/(nsid/2 - 1)
dr = 1.0/(nr - 1)
do 15 iq = 1,nkl
    m = ival( linptr(iq),1 )
    iod = ival( linptr(iq),2 )
    ipq = ir(iq)
do 10 i = 1,nsid
    x = (i - icent)*dx
do 20 j = 1,nsid
    zklmap(i,j,iq) = 0.0
    y = (j - icent)*dx
    r = sqrt(x**2 + y**2)
    th = 0.0
    if( r.le.1.0) then
        if( r.gt.0.0) then
            th = atan2(y,x)
            if(th.le.0.0) th = 2.0*pi + th
            else
                th = 0.0
        endif
    if(r.lt.0.99999) then
        k1 = int(r/dr) + 1
        k2 = k1 + 1
        r1 = (k1 - 1)*dr
        r2 = (k2 - 1)*dr
        coef1 = (r2 - r)/dr
        coef2 = (r - r1)/dr
        top = coef1*rmap(k1,ipq) + coef2*rmap(k2,ipq)
    else
        top = rmap(nr,ipq)
    endif
    if(iod.eq.1) zz = cos(m*th)
    if(iod.eq.2) zz = sin(m*th)

152
SUBROUTINE PARAM2(IDIM, NYQUIST, TILT, ISEED)

The function of this subroutine is to ask the user various questions about the subroutines he or she wishes to use and what values he or she wants to assign to the variables in the program.

REAL DIAM, LAMBDAR0, PIXELNUMOBSCUR, PIXSCALE,
+ SECDIM, TPFDIM
INTEGER IDIM
CHARACTER*16 FILENAME
CHARACTER*1 FILEANSWER, FLAG, TILT
COMMON/VARS2/DIAM, OBSCUR, LAMBDA, R0, SECDIM,
+ PIXSCALE, TPFDIM, FILENAME

WRITE(*,*) 'THE DEFAULT VALUES FOR THIS PROGRAM ARE AS FOLLOWS: '
WRITE(*,*) 'TELESCOPE DIAMETER: ', DIAM
WRITE(*,*) 'CENTER WAVELENGTH: ', LAMBDAR0
WRITE(*,*) 'TELESCOPE SECONDARY DIAMETER: ', OBSCUR

20 CONTINUE
WRITE(*,*) 'USE THE DEFAULT VALUES? (Y/N)'
READ(*,14) FLAG
IF(((FLAG.NE.'Y').AND.(FLAG.NE.'y')).AND.
+ ((FLAG.NE.'N').AND.(FLAG.NE.'n'))) THEN
  WRITE(*,*) 'YOUR ANSWER HAS TO BE EITHER Y OR N.'
  FLAG = ''
  GOTO 20
ENDIF
CONTINUE

IF((FLAG.EQ.'N').OR.(FLAG.EQ.'n'))THEN
    WRITE(*,'')
    WRITE(*,'A: ALL VARIABLES')
    WRITE(*,'D: TELESCOPE DIAMETER')
    WRITE(*,'W: CENTER WAVELENGTH (LAMBDA)')
    WRITE(*,'R: R0')
    WRITE(*,'S: SECONDARY DIAMETER')
    WRITE(*,'')
    WRITE(*,'WHICH VALUE WOULD YOU LIKE TO CHANGE?')
    READ(*,14) ANSWER
    IF(((ANSWER.NE.'A').AND.(ANSWER.NE.'a')).AND.(
        ((ANSWER.NE.'D').AND.(ANSWER.NE.'d')).AND.(
        ((ANSWER.NE.'W').AND.(ANSWER.NE.'w')).AND.(
        ((ANSWER.NE.'R').AND.(ANSWER.NE.'r')).AND.(
        (ANSWER.NE.'S').AND.(ANSWER.NE.'s'))))THEN
        WRITE(*,')
        WRITE(*,'ANSWER IS NOT A CHOICE. REENTER')
        GOTO 30
    ENDIF
    WRITE(*,'')
    THE CURRENT VALUES ARE:
    WRITE(*,'')
    WRITE(,'TELESCOPE DIAMETER: ',DIAM
    WRITE(,'CENTER WAVELENGTH: ',LAMBDA
    WRITE(,'R0: ',R0
    WRITE(,'SECONDARY DIAMETER ',OBSCUR
    WRITE(,'')
    IF((ANSWER.EQ.'A').OR.(ANSWER.EQ.'a'))THEN
        WRITE(*,'INPUNT THE FOLLOWING')
        WRITE(*,'ENTER NEW TELESCOPE DIAMETER:')
        READ(*,100) DIAM
        WRITE(*,'ENTER NEW CENTER WAVELENGTH:')
        READ(*,100) LAMBDA
        WRITE(*,'ENTER NEW R0:')
        READ(*,100) R0
        WRITE(*,'ENTER NEW TELESCOPE SECONDARY DIAM:');
READ(*,100) OBSCUR
ELSEIF((ANSWER.EQ.'D').OR.(ANSWER.EQ.'d'))THEN
  WRITE(*,*)'ENTER NEW TELESCOPE DIAMETER:'
  READ(*,100)DIAM
ELSEIF((ANSWER.EQ.'W').OR.(ANSWER.EQ.'w'))THEN
  WRITE(*,*)'ENTER NEW CENTER WAVELENGTH:'
  READ(*,100) LAMBDA
ELSEIF((ANSWER.EQ.'R').OR.(ANSWER.EQ.'r'))THEN
  WRITE(*,*)'ENTER THE NEW R0:'
  READ(*,100)R0
ELSEIF((ANSWER.EQ.'S').OR.(ANSWER.EQ.'s'))THEN
  WRITE(*,*)'ENTER THE NEW SECONDARY DIAMETER:'
  READ(*,100) OBSCUR
ENDIF
CONTINUE
WRITE(*,*)'WOULD YOU LIKE TO CHANGE ANY OTHER VALUE?
+ (Y/N)'
READ(*,14)ANSWER
IF(((ANSWER.NE.'Y').AND.(ANSWER.NE.'y')).AND.
+ ((ANSWER.NE.'N').AND.(ANSWER.NE.'n')))THEN
  WRITE(*,*)'YOUR ANSWER HAS TO BE EITHER Y OR N.
+ REENTER'
  GOTO 50
ENDIF
IF(((ANSWER.EQ.'Y').OR.(ANSWER.EQ.'y')))THEN
  GOTO 30
ENDIF
OPEN(UNIT=10,FILE=FILENAME,FORM='FORMATTED',
+ STATUS='UNKNOWN')
REWIND(10)
WRITE(10,*)DLAM,OBSCUR,LAMBDA,R0
CLOSE(UNIT=10)
ENDIF

WRITE(*,*)'ENTER THE NUMBER OF PIXELS PER R0:'
READ(*,*)PIXELNUM
WRITE(*,*) 'ENTER SEED VALUE:'
READ(*,*) ISEED
PIXSCALE = R0/PIXELNUM
TPFDIM = nint(DIAM*PIXELNUM/R0)
SECDIM = nint(OBSCUR*PIXELNUM/R0)

60 CONTINUE
65 CONTINUE
WRITE(*,*) 'DO YOU WISH TO INCLUDE X/Y TILT? (Y/N)'
READ(*,14) TILT
IF(((TILT.NE.'Y').AND.(TILT.NE.'y')).AND. 
+ ((TILT.NE.'N').AND.(TILT.NE.'n'))) THEN
   WRITE(*,*) 'YOUR ANSWER MUST BE EITHER Y OR N.'
   + REENTER'
   GOTO 65
ENDIF

c Check the value for nyquist to assure it is equal to tpfdim

IF(NYQUIST.NE.TPFDIM GOTO 999
RETURN

   c Format Statements

14 FORMAT(A1)
100 FORMAT(E11.4)
110 FORMAT(I6)

   c Error statements

999 WRITE(*,*) '
   WRITE(*,*) 'ERROR. VALUE FOR NYQUIST MUST EQUAL 
   + TO: ', TPFDIM
   + ', NOT ', nyquist, '.'
   WRITE(*,*) '
   WRITE(*,*) 'PROGRAM STOPPING. CHANGE NYQUIST.'
   STOP
END
SUBROUTINE PROJ(bmap,bval,zi,kmap,linptr,iseed,  
+ nkl,r0,n1,n2,n3,n4,idim2,pi,pixscale,tilt)

The function of this subroutine is to project off the low-order Karhunen-Loeve  
functions and then adds them back with the proper strength.

n1 is the number of azimuthal orders, bmap is the phase screen

complex*16 bmap(idim2,idim2),zap,sum  
dimension bval(n3),linptr(n3)  
dimension zlmap(idim2,idim2,nkl)  
real x,y,yi,pi,twopi,pixscale,dr0,r0  
integer izern,idim2  
character tilt

izern = 5  
twopi = 2.e0*pi  
dr0 = (pixscale*idim2)/r0  
DO 10 i = 1,izern

A random size is generated for each karhunen function

x = sqrt(bval(linptr(i) ) )  
y = 0.e0  
yi = 0.e0  
mu = 0.e0  
sigma = X  
call gauss(musigmayyi,pi,iseed)  
sum = cmplx(Y,YI)  
sum = sum*(dr0**.83333333)  
call inprod(bmap,zi,kmap,i,idim2,nkl,zap)

The function is projected out

if(tilt.eq.'N'.or.tilt.eq.'n') then  
  if(i.le.2)sum = 0.0  
endif  
sum = sum - zap  
call fadd(bmap,zi,kmap,i,idim2,nkl,sum)

10 CONTINUE
RETURN
END
SUBROUTINE TPF2A(PS1,PS2,IDIM,IDIM2,TPFDIM,SECDIM)

Function: The function of this subroutine is to cut out a portion of array with
the dimension of TPFDIM and put the data into an array of zeros.

COMPLEX*16 PS1(IDIM,IDIM),PS2(IDIM2,IDIM2)
REAL TPFDIM,SECDIM
INTEGER IC,IC2

IC = IDIM/2 + 1
IC2 = IDIM2/2 + 1

Zero out psi to ensure that no reminents of previous phase screen is left over.

DO 10 J = 1,IDIM
   DO 10 K = 1,IDIM
      PS1(J,K) = (0.0,0.0)
   10 CONTINUE

Divide the telescope pupil dimension by 2 in order to obtain the radius instead
of the diameter of the pupil.

TPFDIM2 = TPFDIM/2
SECDIM2 = SECDIM/2

IF(SECDIM.EQ.0.0) PS1(IC,IC) = PS2(IC2,IC2)

Now insert the phase screen into the larger array. The smaller array is the data
that this telescope is able to see due to the inhibition of its size.

DO 20 I = 0, TPFDIM2
   DO 20 J = 0, TPFDIM2
      IF((I**2 + J**2).LE.TPFDIM2**2) THEN
         IF((I**2 + J**2).GT.SECDIM2**2) THEN
            PS1(IC+I,IC+J) = PS2(IC2+I,IC2+J)
            PS1(IC+I,IC-J) = PS2(IC2+I,IC2-J)
            PS1(IC-I,IC+J) = PS2(IC2-I,IC2+J)
            PS1(IC-I,IC-J) = PS2(IC2-I,IC2-J)
         ENDIF
      ENDIF
   20 CONTINUE
RETURN
END
FUNCTION GAMMLN(xx)
real cof(6),stp, half, one, fpf, x, tmp, ser
data cof, stp/76.18009173d0,-86.50532033d0,
+ 24.01409822d0,-1.231739516d0, 120858003d-2,
+ -536382d-5,2.50662827456d0/
data half, one, fpf/0.5d0, 1.0d0, 5.5d0/

x = xx - one
tmp = x + fpf
tmp = (x + half)*log(tmp) - tmp
ser = one

do 10 j = 1,6
   x = x + one
   ser = ser + cof(j)/x
10 continue

gammln = tmp + log(stp*ser)

return
end

REAL FUNCTION POISSON(pmean, iseed)
c mean of poisson distribution
real pmean
c Seed for random number generator
integer iseed
c Summary of purpose
c Generate a random number with a poisson distribution of mean pmean (poisson deviate)
c Author
c numerical recipes, p207, routine poidev.for
c Modifications

13 nov, 1987: pat fitch
1988: erik m johansson - fixed integer conversion problem with maxint

Routines called

uniform
gammln

(c) Copyright 1987 the Regents of the University of California. All rights reserved.

This software is a result of work performed at Lawrence Livermore National Laboratory. The United States Government retains certain rights therein.

real pi
parameter (pi=3.141592654)

maxint is the largest real number which can be converted to an integer without resulting in an arithmetic error (32 bits)

real maxint
parameter (maxint = .214748352e10)

real pexpmean, oldmean, t, em, sq, alxm, y, gammln,
+ uniform
integer times
data oldmean /=1./
if(pmean.lt.12.0)then

Use direct computation method

if(pmean.ne.oldmean) then

New mean, calculate the exponential

oldmean = pmean
pexpmean= exp(-pmean)
endif
em = -1.0
t = 1.0
2 em = em + 1.0

160
t = t * uniform(iseed)
if(t.gt.pexpmean) go to 2
else
  if(pmean.ne.oldmean) then
    oldmean = pmean
    sq = sqrt(2.0*pmean)
    abxm = alog(pmean)
  endif
  times = 0
endif

y = tan( pi* uniform(iseed))
times = times + 1
if(times .ge. 1000) then
  write(*,*)'ERROR: STUCK IN LOOP IN POISSON'
  stop
endif
em = sq*y+pmean
if((em .lt. 0.) .or. (em .gt. maxint)) go to 1
em = int(em)
t = 0.9*(1.+y**2)*exp(em*abxm-gamln(em+1.)-
  +  pexpmean)
  if(uniform(iseed).gt.t) go to 1
endif

poisson = em

return
end
REAL FUNCTION UNIFORM(seed)

integer*4 seed

c  Summary of purpose

c  implements a multiplicative linear congruential generator generates random

c  numbers uniformly distributed on the open interval 0.0 to 1.0 (0 and 1 are NOT

c  included)

c  Author

c  from "Random Number Generators: Good Ones Are Hard to Find," Stephen


c  31, No 10, pp 1192 - 1201 routine is listed on p 1195


c  modifications

4/11/89 erik m johansson - modified code to use less computational steps.

c  Equations used are from the article "Efficient and Portable Combined Random

c  Number Generators," Pierre L'Ecuyer, Communications of the ACM, June

c  1988, Vol 31, No 6, at the top of p745.

(c) Copyright 1989 the Regents of the University of California. All rights

c  reserved.

c  This software is a result of work performed at Lawrence Livermore National

c  Laboratory. The United States Government retains certain rights therein.

integer*4 a, m, q, r

parameter (a = 16807)
parameter (m = 2147483647)
parameter (q = 127773)
parameter (r = 2836)

real h
parameter (h = 1. / 2147483647.)
integer*4 k
The function

\[ k = \text{seed} / q \]
\[ \text{seed} = a \times (\text{seed} - k \times q) - k \times r \]
\[ \text{if (seed} \lt 0) \text{seed} = \text{seed} + m \]
\[ \text{uniform} = \text{seed} \times h \]
\[ \text{return} \]
\[ \text{end} \]
APPENDIX H. PHASE ERROR AND LOW PASS FILTER PROGRAM

This program was developed by the thesis author and develops a low pass filter in the frequency domain to filter the reconstructed object data file for both the Knox-Thompson and Triple-Correlation methods. Additionally, this program determines the azimuthal RMS phase error for the input Fourier spectra.

The following subroutines are required from Universal Subroutines in Appendix D:

- Complexconj
- FFT
- FFT2D
- NormFFT
- Quadswap

Author: LT James M. Lackemacher
Compl. Date: 26 October 1990
Reason: Complete requirements for a Masters Degree in Physics
Goal: Simulate object, degrade object, reconstruct object using Knox-Thompson and Triple-Correlation methods filter the data and compare the two methods.

Program FREQFILTER

Main program complex variable list

BSDATA n x n DIM array representing the input triple-correlation reconstructed object in
FREQUENCY SPACE

BSLP n x n DIM ARRAY REPRESENTING THE LOW PASS FILTERED TRIPLE-CORRELATION RECONSTRUCTED OBJECT IN FREQUENCY SPACE

F n DIM ARRAY USED IN THE FOURIER TRANSFORM

KTDATA n x n DIM ARRAY REPRESENTING THE INPUT KNOX-THOMPSON RECONSTRUCTED OBJECT IN FREQUENCY SPACE

KTLP n x n DIM ARRAY REPRESENTING THE LOW PASS FILTERED KNOX-THOMPSON RECONSTRUCTED OBJECT IN FREQUENCY SPACE

TEMPDATA n x n DIM ARRAY THAT IS USED AS A TEMPORARY ARRAY IN THE FOURIER TRANSFORM

TRUDATA n x n DIM ARRAY REPRESENTING THE TRUTH DATA

MAIN PROGRAM REAL VARIABLE LIST

bserror n x n DIM ARRAY THAT REPRESENTS THE SQUARE PHASE ERROR FOR THE TRIPLE-CORRELATION RECONSTRUCTED IMAGE

bslpmod n x n DIM ARRAY THAT REPRESENTS THE MODULUS OF THE LOW PASS FILTERED TRIPLE-CORRELATION RECONSTRUCTED IMAGE

bsr n/2 DIM ARRAY THAT REPRESENTS THE RADIAL FREQ VALUE IN INVERSE ARCSECONDS FOR THE TRIPLE-CORRELATION IMAGE

bssnr n/2 DIM ARRAY THAT REPRESENTS THE INPUT SNR VALUES OF TRIPLE-CORRELATION IMAGE

bstrunc TRIPLE-CORRELATION RADIAL TRUNCATION VALUE

kterror n x n DIM ARRAY THAT REPRESENTS THE SQUARE PHASE ERROR FOR THE KNOX-THOMPSON RECONSTRUCTED IMAGE

ktlpmod n x n DIM ARRAY THAT REPRESENTS THE MODULUS OF THE LOW PASS FILTERED KNOX-THOMPSON RECONSTRUCTED IMAGE

ktr n/2 DIM ARRAY THAT REPRESENTS THE RADIAL FREQ VALUE IN INVERSE ARCSECONDS FOR THE KNOX-THOMPSON IMAGE

ktsnr n/2 DIM ARRAY THAT REPRESENTS THE INPUT SNR VALUES OF KNOX-THOMPSON IMAGE

165
c kttrunc KNOX-THOMPSON RADIAL TRUNCATION VALUE

lambda WAVELENGTH OF QUASI-MONOCHROMATIC LIGHT

rBSerror n/2 DIM ARRAY THAT REPRESENTS THE AZIMUTHAL RMS PHASE ERROR OF THE TRIPLE-CORRELATION RECONSTRUCTED IMAGE

rKTerror n/2 DIM ARRAY THAT REPRESENTS THE AZIMUTHAL RMS PHASE ERROR OF THE KNOX-THOMPSON RECONSTRUCTED IMAGE

R0 COHERENCE LENGTH

MAIN PROGRAM INTEGER VARIABLE LIST

fwd VALUE OF 1 FOR FORWARD FFT

iktcount NUMBER OF KT PIXELS WITH SNR GREATER THAN 1.0

inv VALUE OF -1 FOR INVERSE FFT

itccount NUMBER OF TC PIXELS WITH SNR GREATER THAN 1.0

ln $2^\ln$ FOR USE WITH FFT SUBROUTINE

n DIMENSION OF ONE SIDE OF 2-DIM ARRAY

nyquist EQUAL TO THE TELESCOPE PUPIL FUNCTION DERIVED FROM THE FOLLOWING FORMULA:

nyquist = (telescope diameter x number of pixels/r0)/r0

numpix THE NUMBER OF PIXELS PER R0

MAIN PROGRAM INTEGER VARIABLE LIST

end CHAR INPUT TO DETERMINE WHETHER TO STOP THE PROGRAM

mean CHAR INPUT TO DETERMINE WHETHER TO FIND THE AMSPE

rite CHAR INPUT TO DETERMINE WHETHER TO WRITE DATA TO FILE
MAIN PROGRAM

PARAMETER(n=64, ln=6, fwd=1, inv=-1)

COMPLEX*16 KTDATA(n,n), BSDATA(n,n), TEMPDATA(n,n),
+ F(n), KTLP(n,n), BSLP(n,n), TRUDATA(n,n)

REAL*8 ktlpmod(n,n), bslpmod(n,n), kterror(n,n),
+ berror(n,n), rKTerror(n/2), rBSerror(n/2),
+ ktsnr(n/2), bssnr(n/2)

REAL ktr(n/2), bsr(n/2), kttrunc, bstrunc, lambda

CHARACTER*1 mean, end, rite

c INPUT THE DATA

CALL Readfile(KTDATA, BSDATA, n)
WRITE(*,*) 'ENTER NYQUIST VALUE:'
READ(*,*) nyquist
WRITE(*,*)
WRITE(*,*) 'ENTER THE NUMBER OF PIXELS PER R0:'
READ(*,*) numpix
WRITE(*,*)
WRITE(*,*) 'ENTER R0 VALUE IN METERS:'
READ(*,*) R0
WRITE(*,*)
WRITE(*,*) 'ENTER WAVELENGTH VALUE IN METERS:'
READ(*,*) lambda
WRITE(*,*)
CALL Readsnr(ktsnr, bssnr, ktr, bsr, nyquist, n)

C LOW PASS FILTER BASED ON RADIAL TRUNCATION VALUE
C DETERMINED FROM SNR DATA

CALL Truncval(ktsnr, ktr, kttrunc, nyquist, iktcount, n)
WRITE(*,*) 'KT RADIAL TRUNCATION VALUE IS:'
WRITE(*,*) kttrunc, '1/ARCSEC.'
WRITE(*,*) 'VALUES GREATER THAN THIS RADIUS'
WRITE(*,*) 'WILL BE TRUNCATED.'
WRITE(*,*)
CALL Truncval(bssnr, bsr, bstrunc, nyquist, itccount, n)
WRITE(*,*) 'TC RADIAL TRUNCATION VALUE IS:'
WRITE(*,*) bstrunc, '1/ARCSEC.'
WRITE(*,*) 'VALUES GREATER THAN THIS RADIUS'
WRITE(*,*) 'WILL BE TRUNCATED.'
WRITE(*,*) '

C TRUNCATE THE DATA

CALL Truncate(KTDATA, KTLP, ktrunc, lambda, R0, numpix, n)
CALL Truncate(BSDATA, BSLP, bstrunc, lambda, R0, numpix, n)

C INVERSE FFT THE DATA AFTER FILTERING

CALL FFT2D(KTLP, TEMPDATA, F, ln, inv, n)
CALL FFT2D(BSLP, TEMPDATA, F, ln, inv, n)

C DETERMINE THE MODULUS OF THE DATA IN IMAGE SPACE

CALL Modulus(ktlpmod, KTLP, n)
CALL Modulus(bslpmod, BSLP, n)

C NORMALIZE THE MODULI FOR PLOTTING

CALL Normalize(ktlpmod, n)
CALL Normalize(bslpmod, n)

C WRITE THE MODULUS TO A FILE FOR PLOTTING IF DESIRED

20 WRITE(*,*) 'WRITE MODULUS TO A FILE? (Y/N)'
READ(*,*) rite
WRITE(*,*) '
IF ((rite.NE.'Y').AND.(rite.NE.'y').AND. + (rite.NE.'N').AND.(rite.NE.'n')) THEN
   WRITE(*,*) 'ERROR, REENTER.'
   WRITE(*,*) '
   GOTO 20
ENDIF
IF ((rite.EQ.'Y').OR.(rite.EQ.'y')) THEN
   CALL Writefile(ktlpmod, bslpmod, numpix, lambda, R0, n)
ENDIF
DETERMINE THE AZIMUTHAL RMS PHASE ERROR

30 WRITE(*,*)'FIND THE AZIMUTHAL RMS PHASE ERROR? (Y/N)'
READ(*,*)mean
WRITE(*,*)'
IF ((mean.NE.'Y').AND.(mean.NE.'y').AND.
+ (mean.NE.'N').AND.(mean.NE.'n')) THEN
WRITE(*,*)'ERROR, REENTER.'
WRITE(*,*)'
GOTO 30
ENDIF
IF ((mean.EQ.'Y').OR.(mean.EQ.'y')) THEN

READ IN DATA
CALL Readtrufile(TRUDATA,n)

DETERMINE SQUARE PHASE ERROR
CALL AMSPE(KDATA,BSDATA,TRUDATA,
+ kterror,bseerror,nyquist,n)

DETERMINE AZIMUTHAL AVERAGE OF THE RMS PHASE ERROR
CALL AMSPEcalc(kterror,bseerror,rKError,rBError,
+ nyquist,n)

WRITE AMSPE TO A FILE FOR EACH CORRELATION TECHNIQUE
CALL Writersfile(rKError,rBError,nyquist,lambda,
+ R0,numpix,ikcount,itccount,n)

ENDIF
STOP
END
SUBROUTINE AMSPE(KTDATA, BSDATA, TRUDATA, kterror, berror, nyquist, n)

    THIS S/R CALCULATES THE SQUARE PHASE ERROR OF THE RECONSTRUCTED PHASES

    COMPLEX*16 KTDATA(n,n), BSDATA(n,n), TRUDATA(n,n)
    REAL*8 kterror(n,n), berror(n,n), truphase, ktphase,
    + bsphase, pi, pi2
    pi = dacos(-1.0D+00)
    pi2 = 2 * pi
    n2p1 = n/2 + 1
    DO 10 i = 1, n
        DO 10 j = 1, n
            x = float(j - (n2p1))
            y = float((n2p1) - i)
            radius = sqrt(x**2.0 + y**2.0)
            IF (radius.LE.nyquist) THEN
                truphase = datan2(DIMAG(TRUDATA(i,j)),
                + DREAL(TRUDATA(i,j)))
                ktphase = datan2(DIMAG(KTDATA(i,j)),
                + DREAL(KTDATA(i,j)))
                bsphase = datan2(DIMAG(BSDATA(i,j)),
                + DREAL(BSDATA(i,j)))
            20 IF (truphase.GT.pi2) THEN
                truphase = truphase - pi2
                GOTO 20
            ENDIF
            30 IF (truphase.LT.-pi2) THEN
                truphase = truphase + pi2
                GOTO 30
            ENDIF
            40 IF (ktphase.GT.pi2) THEN
                ktphase = ktphase - pi2
                GOTO 40
            ENDIF
            50 IF (ktphase.LT.-pi2) THEN
                ktphase = ktphase + pi2
                GOTO 50
            ENDIF
        10 CONTINUE
    END
IF (bsphase.GT.pi2) THEN
  bsphase = bsphase - pi2
GOTO 60
ENDIF

IF (bsphase.LT.-pi2) THEN
  bsphase = bsphase + pi2
GOTO 70
ENDIF

kterror(i,j) = truphase - ktphase

IF (kterror(i,j).GT.pi2) THEN
  kterror(i,j) = kterror(i,j) - pi2
GOTO 80
ENDIF

IF (kterror(i,j).LT.-pi2) THEN
  kterror(i,j) = kterror(i,j) + pi2
GOTO 90
ENDIF

IF (kterror(i,j).GT.pi)
  kterror(i,j) = kterror(i,j) - pi2
ENDIF

bserror(i,j) = truphase - bsphase

IF (bserror(i,j).GT.pi2) THEN
  bserror(i,j) = bserror(i,j) - pi2
GOTO 100
ENDIF

IF (bserror(i,j).LT.-pi2) THEN
  bserror(i,j) = bserror(i,j) + pi2
GOTO 110
ENDIF

IF (bserror(i,j).GT.pi)
  bserror(i,j) = bserror(i,j) - pi2
ENDIF

bserror(i,j) = (bserror(i,j))**2.0

CONTINUE
RETURN
END
SUBROUTINE AMSPEcalc(kterror,berror,rKError, + rBSerror,nyquist,n)

c THIS S/R CALCULATES THE RMS PHASE ERROR AS A FUNCTION OF
  c RADIUS

REAL*8 kterror(n,n), berror(n,n), rKError(n/2), + rBSerror(n/2)
INTEGER r
n2p1 = n/2 + 1
DO 10 r = 0, nyquist
  nerr = 0
  DO 20 i = 1, n
    DO 20 j = 1, n
      x = float(j - (n2p1))
      y = float((n2p1) - i)
      radius = sqrt(x**2.0 + y**2.0)
      IF ((radius.GE.float(r)).AND. + (radius.LT.float(r+1))) THEN
        nerr = nerr + 1
        rKError(r) = rKError(r) + kterror(i,j)
        rBSerror(r) = rBSerror(r) + berror(i,j)
      ENDF
    20 CONTINUE
  rKError(r) = dsqrt(rKError(r)/nerr)
  rBSerror(r) = dsqrt(rBSerror(r)/nerr)
10 CONTINUE
RETURN
END

SUBROUTINE Modulus(mod,DATA,n)

  c THIS S/R DETERMINES THE MODULUS OF A COMPLEX NUMBER

COMPLEX*16 DATA(n,n)
REAL*8 mod(n,n)
DO 10 i = 1, n
  DO 10 j = 1, n
    mod(i,j) = ABS(DATA(i,j))
  10 CONTINUE
RETURN
END
SUBROUTINE Normalize(data,n)

  THIS S/R NORMALIZES THE OUTPUT DATA TO 1.0.

REAL*8 data(n,n), maxval
maxval = 0.0D+00
DO 10 i = 1, n
  DO 10 j = 1, n
    IF (data(i,j).GT.maxval) maxval = data(i,j)
  10 CONTINUE
DO 20 i = 1, n
  DO 20 j = 1, n
    data(i,j) = data(i,j)/maxval
  20 CONTINUE
RETURN
END

SUBROUTINE Readfile(KTDATA,BSDATA,n)

  THIS S/R READS THE RECONSTRUCTED DATA FROM A FILE

COMPLEX*16 KTDATA(n,n), BSDATA(n,n)
CHARACTER*16 ktfile, bsfile
WRITE(*,*)'ENTER INPUT KT RECON FILE NAME (16 CHAR')
WRITE(*,*)'MAX):'
READ (*,30) ktfile
WRITE(*,*)'ENTER INPUT TC RECON FILE NAME (16 CHAR'
WRITE(*,*)'MAX):'
READ (*,30) bsfile
WRITE(*,*)'
OPEN(UNIT=40,FILE=ktfile,STATUS='OLD')
OPEN(UNIT=50,FILE=bsfile,STATUS='OLD')
DO 10 i = 1, n
  DO 10 j = 1, n
    READ(40,*) KTDATA(i,j)
  10 CONTINUE
DO 20 i = 1, n
   DO 20 j = 1, n
      READ(50,*) BSDATA(i,j)
   20 CONTINUE
30 FORMAT(A16)
RETURN
END

SUBROUTINE Readsnr(ktsnr,bssnr,ktr,bsr,nyquist,n)

   c THIS S/R READS THE SNR DATA FROM A FILE

   REAL*8 ktsnr(n/2), bssnr(n/2)
   REAL ktr(n/2), bsr(n/2)
   CHARACTER*16 ktsnrfile, bssnrfile

   WRITE(*,*)'ENTER KT SNR FILE NAME (16 CHAR MAX):'
   READ(*,30) ktsnrfile
   WRITE(*,*)'
   WRITE(*,*)'ENTER TC SNR FILE NAME (16 CHAR MAX):'
   READ(*,30) bssnrfile
   WRITE(*,*)'
   OPEN(UNIT=40,FILE=ktsnrfile,STATUS='OLD')
   OPEN(UNIT=50,FILE=bssnrfile,STATUS='OLD')
   DO 10 i = 1, nyquist
      READ(40,*) ktr(i), ktsnr(i)
   10 CONTINUE
   DO 20 i = 1, nyquist
      READ(50,*) bsr(i), bssnr(i)
   20 CONTINUE
30 FORMAT(A16)
RETURN
END

SUBROUTINE Readtrufile(TRUDATA,n)

   c THIS S/R READS THE TRUTH DATA FROM A FILE

   COMPLEX*16 TRUDATA(n,n)
   CHARACTER*16 file
   WRITE(*,*)'ENTER INPUT TRUTH DATA FILE NAME (16 CHAR'
   WRITE(*,*)'MAX):'

   174
READ (*,20) file
WRITE(*,*)
OPEN(UNIT=30,FILE=file,STATUS='OLD')
DO 10 i = 1, n
  DO 10 j = 1, n
    READ(30,*), TRUDATA(i,j)
 10 CONTINUE
20 FORMAT(A16)
RETURN
END

SUBROUTINE Truncate(DATA,LPDATA,trunc,lambda, + R0,numpix,n)
  
  c THIS S/R FILTERS THE OBJECT SPECTRUM ARRAY BY
  c TRUNCATING THE ARRAY AT A RADIUS SET BY THE SNR VALUE
  c GREATER THAN 1

  COMPLEX*16 DATA(n,n), LPDATA(n,n)
  REAL lambda
  n2 = n/2
  n2p1 = n2 + 1
  pi = acos(-1.0E+00)
  DO 10 i = 1, n
    DO 10 j = 1, n
      x = float(j-n2p1)
      y = float(i-n2p1)
      rad = sqrt(x**2 + y**2) * numpix * (lambda/R0) * + (180.0/pi) * 3600.0
      IF (rad.LE.trunc) THEN
        LPDATA(i,j) = DATA(i,j)
      ELSE
        LPDATA(i,j) = (0.0,0.0)
      ENDIF
    10 CONTINUE
  RETURN
END
SUBROUTINE Truncval(snr,r, trunc, nyquist, icount, n)

 c THIS S/R DETERMINES THE TRUNCATION VALUE BASED ON THE
 c SNR. THE VALUE IS BASED ON THE POINT AT WHICH THE SNR IS
 c UNITY OR GREATER.

 REAL*8 snr(n/2)
 REAL r(n/2)
 j = 1
 DO 10 i = 1, nyquist
 IF (snr(i).GE.1.0D+00).AND.(j.EQ.i)) THEN
 trunc = r(i)
 icount = j
 j = j + 1
 ENDIF
 10 CONTINUE
 RETURN
END

SUBROUTINE Writefile(ktlpdata,bslpdata,numpix,
 + lambda,RO,n)

 c THIS S/R WRITES THE DATA TO A FILE

 REAL*8 ktlpdata(n,n), bslpdata(nn)
 REAL conv, lambda, RO, x, y
 CHARACTER*16 ktlpfle, bslpfile
 pi = acos(-1.0E+00)
 WRITE(*,*)'ENTER OUTPUT KT FILE NAME (16 CHAR MAX):'
 READ (*,50) ktlpfile
 WRITE(*,*)'
 WRITE(*,*)'ENTER OUTPUT TC FILE NAME (16 CHAR MAX):'
 READ (*,50) bslpfile
 WRITE(*,*)'
 OPEN(UNIT=30,FILE=ktlpfile,STATUS='NEW')
 OPEN(UNIT=40,FILE=bslpfile,STATUS='NEW')
 DO 10 i = 1, n
    DO 10 j = 1, n
       conv = float(numpix) * (lambda/R0) *
       (180.0/pi) * 3600.0
       x = float(j - (n/2+1)) * conv/float(n)
       y = float(i - (n/2+1)) * conv/float(n)
SUBROUTINE Writerrfile(rKTerror,rBSerror,nyquist, lambda,RO,numpix,iktcount, itccount,n)
!
! THIS S/R WRITES THE ERROR DATA TO A FILE
!
REAL*8 rKTerror(n/2), rBSerror(n/2)
REAL x, y, lambda
CHARACTER*16 ktfile, bsfile
INTEGER r
pi = acos(-1.0+00)
WRITE(*,*)'ENTER OUTPUT KT ERROR FILE NAME (16 CHAR')
WRITE(*,*)'MAX):'
READ (*,30) ktfile
WRITE(*,*)' ENTER OUTPUT TC ERROR FILE NAME (16 CHAR'
WRITE(*,*)'MAX):'
READ (*,30) bsfile
WRITE(*,*)' OPEN(UNIT=1,FILE=ktfile,STATUS='NEW')
OPEN(UNIT=2,FILE=bsfile,STATUS='NEW')
DO 10 r = 1, itccount
 x = r * numpix * (lambda/RO) * (180.0/pi) * 3600.0
 WRITE(1,*) x, rKTerror(r)
10 CONTINUE
DO 20 r = 1, itccount
    x = r * numpix * (lambda/R0) * (180.0/pi) * 3600.0
    WRITE(2,*) x, rBSerror(r)
20 CONTINUE
30 FORMAT(A16)
RETURN
END
INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Cameron Station
   Alexandria, Virginia 22304-6145

2. Library Code 52
   Naval Postgraduate School
   Monterey, California 93943-5002

3. Professor Donald L. Walters, Code PH/We
   Department of Physics
   Naval Postgraduate School
   Monterey, California 93943-5000

4. Professor David S. Davis, Code PH/Dv
   Department of Physics
   Naval Postgraduate School
   Monterey, California 93943-5000

5. Professor Karlheinz E. Woehler, Code PH/Wh
   Department of Physics
   Naval Postgraduate School
   Monterey, California 93943-5000

6. Capt Charles L. Matson
   Weapons Lab/ARCI
   Kirtland Air Force Base
   Albuquerque, New Mexico 87117

7. Capt Michael C. Roggemann
   Weapons Lab/ARCI
   Kirtland Air Force Base
   Albuquerque, New Mexico 87117
8. LT James M. Lackemacher
USS Gallery (FFG-26)
FPO, AA 34093-1482