

AD-A243 127



2

# NAVAL POSTGRADUATE SCHOOL

## Monterey, California

DTIC  
ELECTE  
DEC 09 1991  
S D D



## 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:  
Thesis Co-Advisor:  
Thesis Co-Advisor:

Donald L. Walters  
Charles L. Matson  
Michael C. Roggemann

Approved for public release; distribution is unlimited

91-17301



01

100

# REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION <b>UNCLASSIFIED</b>		1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY <b>MULTIPLE SOURCES</b>		3 DISTRIBUTION/AVAILABILITY OF REPORT <b>APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS INLIMITED</b>	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE		5 MONITORING ORGANIZATION REPORT NUMBER(S)	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)		7a. NAME OF MONITORING ORGANIZATION <b>NAVAL POSTGRADUATE SCHOOL</b>	
6a. NAME OF PERFORMING ORGANIZATION <b>NAVAL POSTGRADUATE SCHOOL</b>	6b OFFICE SYMBOL (if applicable) <b>33</b>	7b ADDRESS (City, State, and ZIP Code) <b>Monterey, California 93943-5000</b>	
6c ADDRESS (City, State, and ZIP Code) <b>Monterey, California 93943-5000</b>		9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8a. NAME OF FUNDING /SPONSORING ORGANIZATION	8b OFFICE SYMBOL (if applicable)	10 SOURCE OF FUNDING NUMBERS	
8c ADDRESS (City, State, and ZIP Code)		PROGRAM ELEMENT NO	PROJECT NO
		TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification)(UNCLASSIFIED) <b>RECONSTRUCTION OF COMPUTER SIMULATED, ATMOSPHERICALLY TURBULENCE-DEGRADED, ASTRONOMICAL OBJECTS BY APPLICATION OF THE KNOX-THOMPSON AND TRIPLE-CORRELATION PHASE RECOVERY TECHNIQUES</b>			
12 PERSONAL AUTHOR(S) <b>Lackemacher, James M.</b>			
13a TYPE OF REPORT <b>Master's Thesis</b>	13b TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Year, Month, Day) <b>December, 1990</b>	15 PAGE COUNT <b>189</b>
16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U. S. Government			
17 COSATI CODES		18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP	
		Image Reconstruction	
		Knox-Thompson	
		Triple-Correlation	
19 ABSTRACT (Continue on reverse if necessary and identify by block number)			
<p>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.</p> <p>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.</p>			
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS		21 ABSTRACT SECURITY CLASSIFICATION <b>UNCLASSIFIED</b>	
22a NAME OF RESPONSIBLE INDIVIDUAL <b>D. L. Walters</b>		22b TELEPHONE (Include Area Code) <b>(408) 646-2116</b>	22c OFFICE SYMBOL <b>33</b>

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

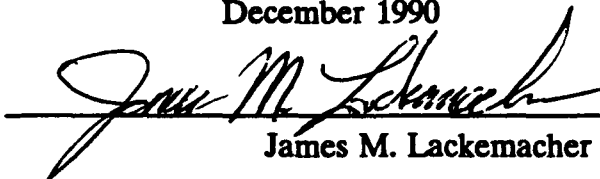
MASTER OF SCIENCE IN PHYSICS

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. Roggeman, 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.



Accession For	
NTIS GRAM	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Date	
Availability Codes	
Dist	Availability Codes
A-1	

## 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
	3. Karhunen-Loeve-Fast-Fourier-Transform Method .....	12
	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

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

$$I(\psi) = I(0) \cdot \left( \frac{2J_1(\psi)}{\psi} \right)^2, \quad (2.1)$$

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

$$\Delta\theta = \frac{1.22 \cdot \lambda}{D}, \quad (2.2)$$

(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 Winitzer 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

$$i(\vec{x}) = o(\vec{x}) * s(\vec{x}) , \quad (3.1)$$

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,  $s(\vec{x})$ , 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,

$$H(\vec{u}) = P(\lambda F \vec{u}) e^{i\Phi(\lambda F \vec{u})} , \quad (3.3)$$

represents the instantaneous coherent transfer function, where  $P(\lambda F \vec{u})$  is the transfer function of the telescope,  $e^{i\Phi(\lambda F \vec{u})}$  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,  $H(\vec{u})$ ,

$$S(\vec{u}) = H(\vec{u}) \star H(\vec{u}) . \quad (3.4)$$

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

$$F_n(\vec{u}) = 0.1517 r_0^{-5/6} |\vec{u}|^{-11/6} , \quad (3.5)$$

where  $|\vec{u}|$  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  $f_k(t)$ ,

$$r(t) = \sum_j r_k f_k(t) . \quad (3.6)$$

The expansion coefficients  $r_k$ , 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 these properties. [Ref. 6]

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

$$C e_i = \lambda_i e_i , \quad (3.7)$$

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

$$K_i(\rho) = \sum_p e_{ip} Z_p(\rho) , \quad (3.8)$$

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

$$\phi(\vec{r}) = \sum_i \gamma_i K_i\left(\frac{\vec{r}}{D/2}\right) , \quad (3.9)$$

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  $(D/r_0)^2$ . 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.

### 3. Karhunen-Loeve-Fast-Fourier-Transform Method

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

$$\langle |I(\vec{u})|^2 \rangle = |O(\vec{u})|^2 \cdot \langle |S(\vec{u})|^2 \rangle . \quad (3.10)$$

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

$$|O(\vec{u})|^2 = \frac{\langle |I(\vec{u})|^2 \rangle}{\langle |S(\vec{u})|^2 \rangle} . \quad (3.11)$$

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,  $\Delta \vec{u}$ . The cross-spectrum,  $CS(\vec{u})$ , is

$$CS(\vec{u}) = X(\vec{u}) \cdot X^*(\vec{u} + \Delta \vec{u}) , \quad (3.12)$$

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

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,  $BS(\vec{u})$ , is

$$BS(\vec{u}) = X(\vec{u}) \cdot X^*(\vec{u} + \Delta \vec{u}) \cdot X(\Delta \vec{u}) , \quad (3.13)$$

where  $X(\Delta \vec{u})$  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

$$\begin{aligned} \langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta \vec{u}) \rangle = \\ [O(\vec{u}) \cdot O^*(\vec{u} + \Delta \vec{u})] \cdot \langle S(\vec{u}) \cdot S^*(\vec{u} + \Delta \vec{u}) \rangle . \end{aligned} \quad (3.14)$$

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



$$\begin{aligned}
\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \cdot I(\Delta\vec{u}) \rangle = \\
[O(\vec{u}) \cdot O^*(\vec{u} + \Delta\vec{u}) \cdot O(\Delta\vec{u})] \cdot \\
\langle S(\vec{u}) \cdot S^*(\vec{u} + \Delta\vec{u}) \cdot S(\Delta\vec{u}) \rangle ,
\end{aligned} \tag{3.15}$$

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

$$O(\vec{u}) \cdot O^*(\vec{u} + \Delta\vec{u}) = \langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \rangle , \tag{3.16}$$

and

$$\begin{aligned}
O(\vec{u}) \cdot O^*(\vec{u} + \Delta\vec{u}) \cdot O(\Delta\vec{u}) = \\
\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \cdot I(\Delta\vec{u}) \rangle .
\end{aligned} \tag{3.17}$$

#### ***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

$$N = |N| e^{i\phi} , \quad (3.18)$$

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

$$e^{i\phi} = N_{ph} = \frac{N}{|N|} , \quad (3.19)$$

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

$$O_{ph}(\vec{u}) \cdot O_{ph}^*(\vec{u} + \Delta\vec{u}) = I_{ph}^{KT}(\vec{u}, \Delta\vec{u}) , \quad (3.20)$$

and

$$O_{ph}(\vec{u}) \cdot O_{ph}^*(\vec{u} + \Delta\vec{u}) \cdot O_{ph}(\Delta\vec{u}) = I_{ph}^{TC}(\vec{u}, \Delta\vec{u}) , \quad (3.21)$$

where

$$I_{ph}^{KT}(\vec{u}, \Delta\vec{u}) = \frac{\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \rangle}{|\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \rangle|} , \quad (3.22)$$

and

$$I_{ph}^{TC}(\vec{u}, \Delta\vec{u}) = \frac{\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \cdot I(\Delta\vec{u}) \rangle}{|\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) \cdot I(\Delta\vec{u}) \rangle|} . \quad (3.23)$$

The terms  $I_{ph}^{KT}(\vec{u}, \Delta \vec{u})$  and  $I_{ph}^{TC}(\vec{u}, \Delta \vec{u})$  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

$$O_{ph}(\vec{u} + \Delta \vec{u}) = \left( \frac{I_{ph}^{KT}(\vec{u}, \Delta \vec{u})}{O_{ph}(\vec{u})} \right)^* , \quad (3.24)$$

and

$$O_{ph}(\vec{u} + \Delta \vec{u}) = \left( \frac{I_{ph}^{TC}(\vec{u}, \Delta \vec{u})}{O_{ph}(\vec{u}) \cdot O_{ph}(\Delta \vec{u})} \right)^* . \quad (3.25)$$

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

$$CS(\vec{u}) = X(\vec{u}) \cdot X^*(\vec{u} + \Delta\vec{u}) - X^*(\Delta\vec{u}) , \quad (3.26)$$

and

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

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

$$O_{ph}(\vec{u} + \Delta\vec{u}) = \left( \frac{I_{ph}^{KTBIAS}(\vec{u}, \Delta\vec{u})}{O_{ph}(\vec{u})} \right)^* , \quad (3.28)$$

and

$$O_{ph}(\vec{u} + \Delta\vec{u}) = \left( \frac{I_{ph}^{TCBIAS}(\vec{u}, \Delta\vec{u})}{O_{ph}(\vec{u}) \cdot O_{ph}(\Delta\vec{u})} \right)^* , \quad (3.29)$$

where

$$I_{ph}^{KTBIAS}(\vec{u}, \Delta\vec{u}) = \frac{\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) - I^*(\Delta\vec{u}) \rangle}{|\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) - I^*(\Delta\vec{u}) \rangle|} , \quad (3.30)$$

and

$$I_{ph}^{TCBIAS}(\vec{u}, \Delta\vec{u}) = \frac{\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) - |I(\vec{u})|^2 - |I(\vec{u} + \Delta\vec{u})|^2 - |I(\Delta\vec{u})|^2 + 2N_p \rangle}{|\langle I(\vec{u}) \cdot I^*(\vec{u} + \Delta\vec{u}) - |I(\vec{u})|^2 - |I(\vec{u} + \Delta\vec{u})|^2 - |I(\Delta\vec{u})|^2 + 2N_p \rangle|} . \quad (3.31)$$

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:

$$\sigma^2(\text{Re}[SS(\vec{u})]) = \left| \frac{\sum_{i=1}^n \left( \text{Re}[SS_i(\vec{u})^2] - \frac{\{\text{Re}[\langle SS(\vec{u}) \rangle]\}^2}{n} \right)}{n-1} \right| \quad (3.32)$$

$$\sigma^2(\text{Im}[SS(\vec{u})]) = \left| \frac{\sum_{i=1}^n \left( \text{Im}[SS_i(\vec{u})^2] - \frac{\{\text{Im}[\langle SS(\vec{u}) \rangle]\}^2}{n} \right)}{n-1} \right| \quad (3.33)$$

$$\sigma = \sqrt{\sigma^2(\text{Re}[SS(\vec{u})]) + \sigma^2(\text{Im}[SS(\vec{u})])} \quad (3.34)$$

$$SNR = \frac{|\langle SS(\vec{u}) \rangle|}{\sigma} \cdot \sqrt{n} , \quad (3.35)$$

where  $SS(\vec{u})$  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,  $|d\vec{u}|$ , is restricted to be less than the seeing limit,  $r_0/\lambda$ , 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,

$$\bar{x} = \frac{\sum_j x_j \cdot I_j(x)}{\sum_j I_j(x)} , \quad (3.36)$$

and

$$\bar{y} = \frac{\sum_j y_j \cdot I_j(y)}{\sum_j I_j(y)} , \quad (3.37)$$

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.

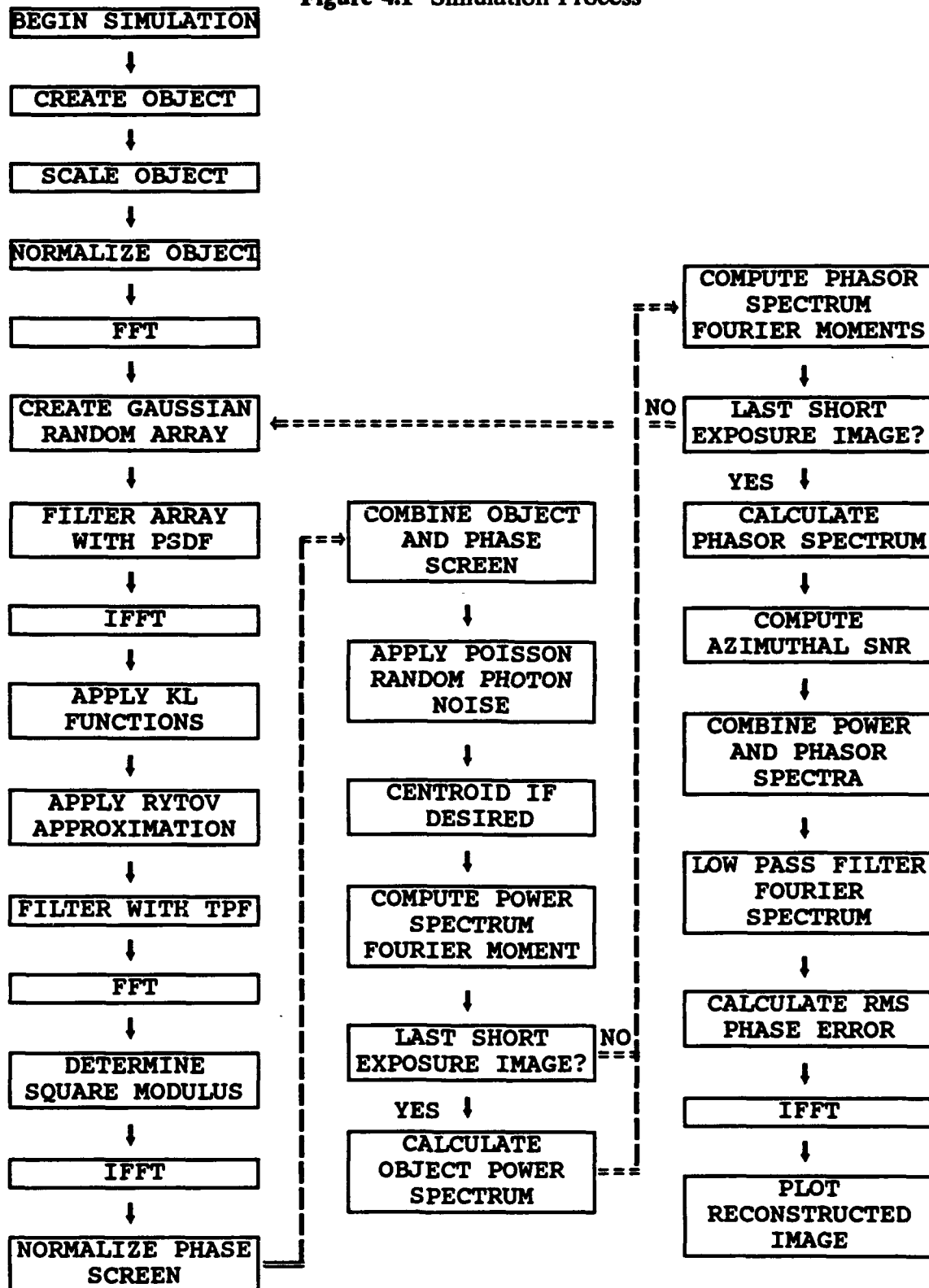
## **1. Object Production**

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



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,  $|\vec{u}|$ , 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

$$FOV = |\vec{u}| \cdot \left( \frac{\lambda}{r_0} \right) , \quad (4.2)$$

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  $\phi_j^{FFT}$ , 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,

$$A \cdot e^{i\phi_j} , \quad (4.3)$$

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

$$O_{i,j} = O_{j,i}^* \quad , \quad (4.4)$$

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  $10^5$ ,  $10^4$ , 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  $N_t$  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  $N_f$  values since the method is shift invariant. However, at  $10^5$  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  $r_0/\lambda$ . 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.

## LIST OF REFERENCES

1. Websters Ninth New Collegiate Dictionary, p. 1004, Merriam-Webster Inc., 1986.
2. Hecht, E., Optics, 2nd ed., pp. 422-423, Addison-Wesley Publishing Co., Inc., 1987.
3. Labeyrie, A., "Attainment of Diffraction Limited Resolution in Large Telescopes by Fourier Analyzing Speckle Patterns in Star Images," Astron. Astrophys., v. 6, pp. 85-87, 1970.
4. Knox, K. T., and Thompson, B. J., "Recovery of Images from Atmospherically Degraded Short-Exposure Photographs," Astrophys. J., v. 193, pp. L45-L48, 1974.
5. Lohmann, A. W., Weigelt, G. P., and Wirtitzer, B., "Speckle Masking in Astronomy: Triple-Correlation Theory and Applications" Appl. Optics, v. 22, pp. 4028-4037, 1983.
6. The Optical Sciences Company Report TR-514, "Wave Front Sensing: A New Approach to Wave Front Reconstruction," by G. A. Tyler and D. L. Fried, pp. 87-90, May 1983.
7. Weapons Lab Technical Report 89-46, "An Atmospheric Computer Simulation for Speckle Imaging Algorithm Analysis," by C. L. Matson and I. E. Drunzer, pp. 14-29, 1989.
8. Whalen, A. D., Detection of Signals in Noise, pp. 285-290, Academic Press, Inc., 1971.
9. Knox, K. T., "Image Retrieval from Astronomical Speckle Patterns," J. Opt. Soc. Am., v. 66, no. 11, pp. 1236-1239, 1976.
10. Knox, K. T., Diffraction-Limited Imaging with Astronomical Telescopes, Ph.D Dissertation, Institute of Optics, University of Rochester, New York, 1975.
11. Ayers, M. J., Northcott, M. J., and Dainty, J. C., "Knox-Thompson and Triple-Correlation Imaging through Atmospheric Turbulence," J. Opt. Soc. Am., v. 5, no. 7, pp. 963-985, 1988.

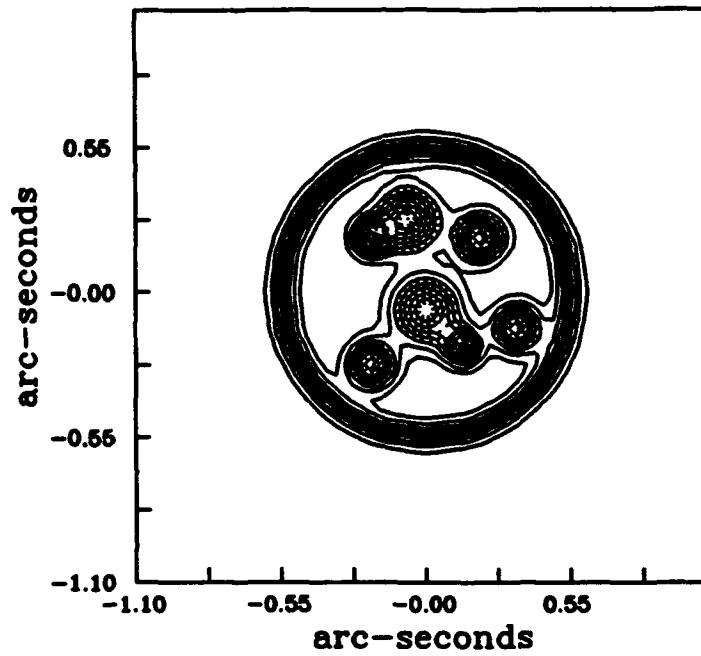
12. Weapons Lab Technical Report 1351-54, "Weighted Least Squares Phase Reconstruction from the Bispectrum," by C. L. Matson, pp. 1-27, 1990.
13. Beletic, J. W., "Comparison of Knox-Thompson and Bispectrum Algorithms for Reconstructing Phase of Complex Extended Objects," paper presented at the ESO conference on High Resolution Imaging by Interferometry, Garching, FRG, March, 1988, pp. 357-362.
14. Walters, D. L., Favier, D. L., and Hines, J. R., "Vertical Path Atmospheric MTF Measurements," J. Opt. Soc. Am., v. 69, pp. 828-837, 1979.
15. Gonzalez, R. C., and Wintz, P., Digital Image Processing, p. 108, Addison-Wesley Publishing Co., Inc., 1987.
16. The Optical Sciences Company Report TR-663, Phase Screen Production, by G. Cochran, pp 6-8, September 1985.

## APPENDIX A. PLOTS AND GRAPHS

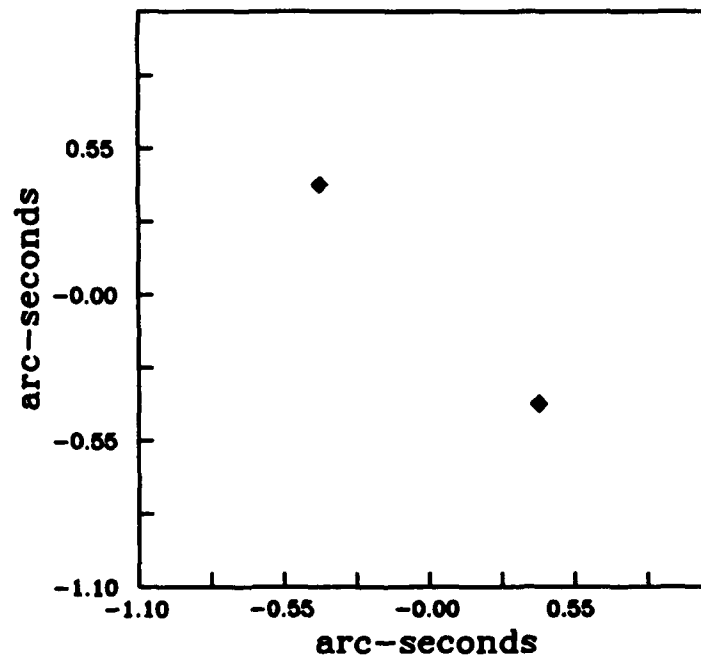
The following table, plots, and graphs were referenced in the text.

**Figure A.I Imaging Parameters.**

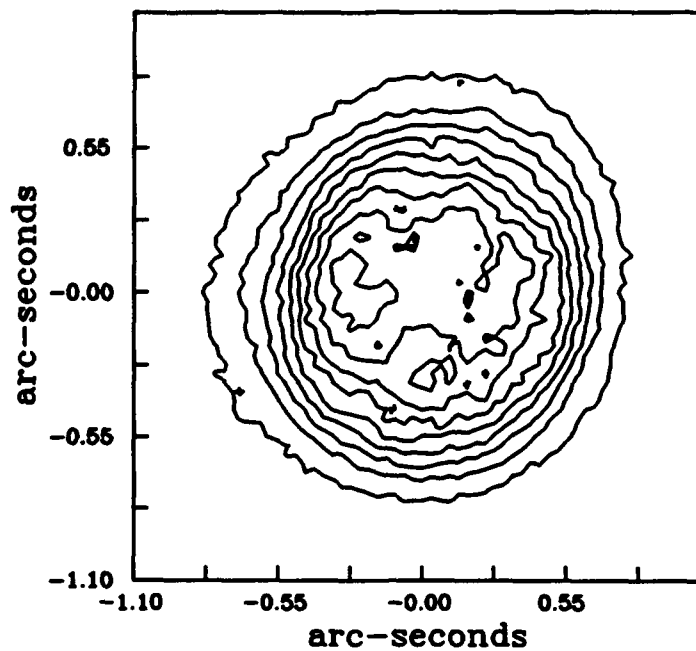
Imaging Parameter	Parameter Abbrev.	Parameter Baseline	Parameter Value
Short Exposure Image Quantity	$N_f$	400	Variable
Coherence Length	$r_0$	0.206 (m)	Variable
Short Exposure Image Photon Quantity	$N_p$	$10^5$	Variable
Offset Value	OS	4 (pixels)	Variable
Telescope Primary Diameter	D	1.6 (m)	Constant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



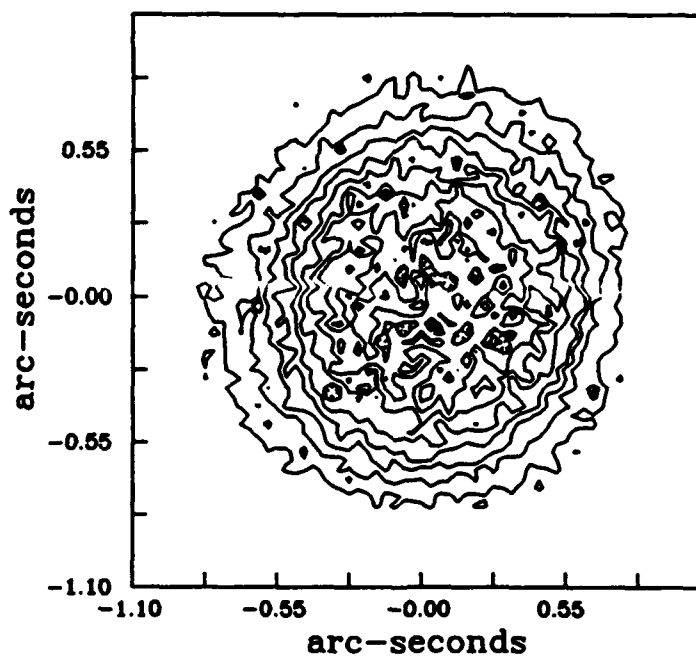
**Figure A.1 True Asteroid Representation With No Aperture Affects.**



**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,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ .

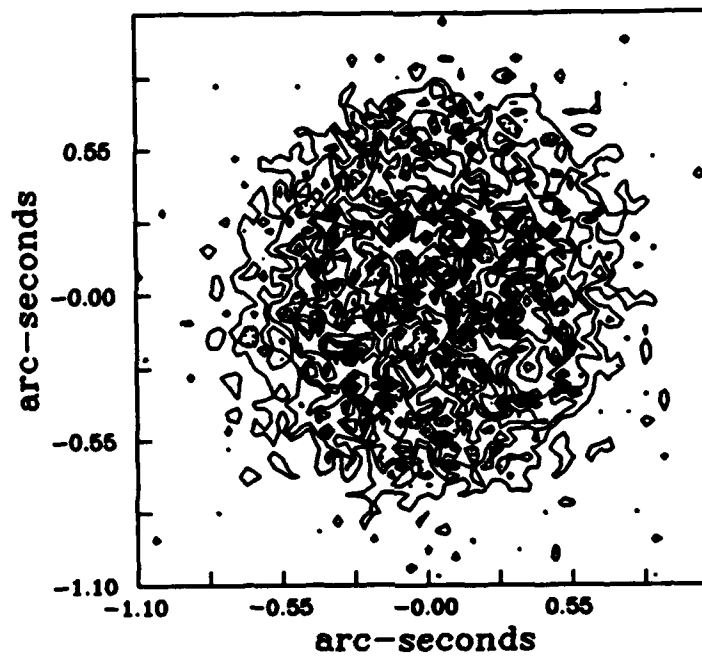


Figure A.5 Asteroid Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^4$ .

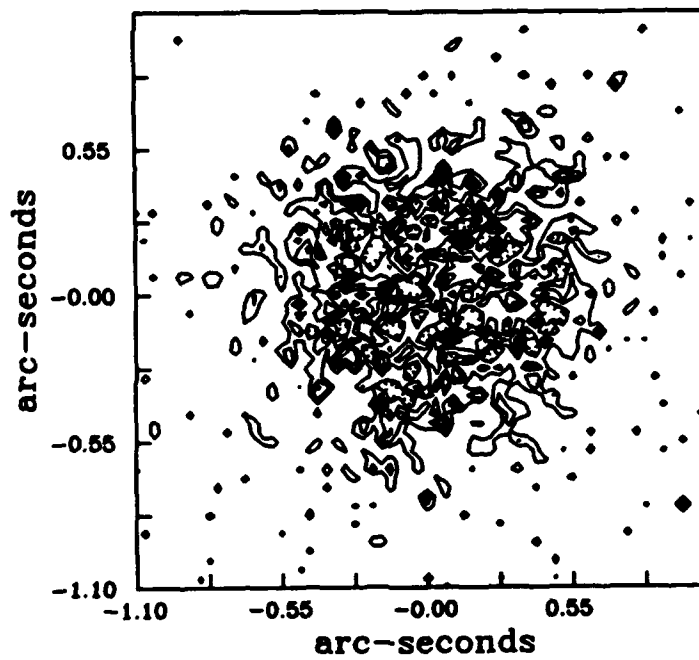
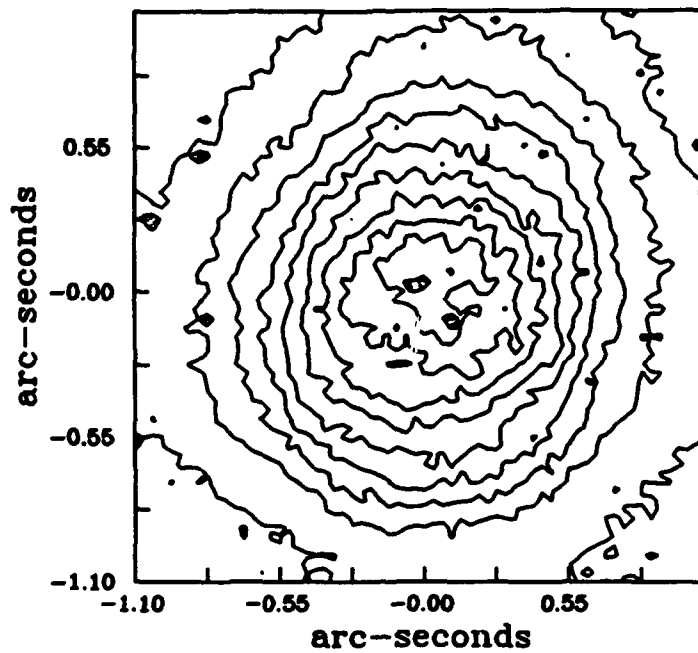
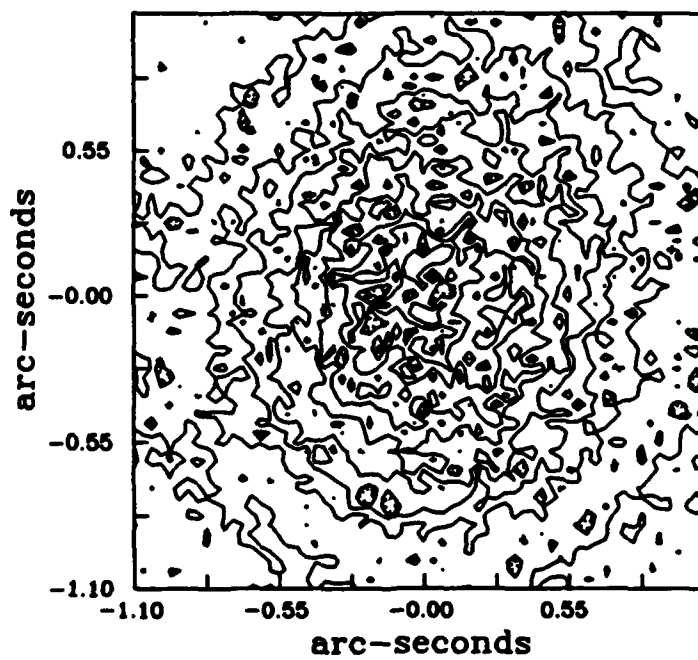


Figure A.6 Asteroid Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^3$ .

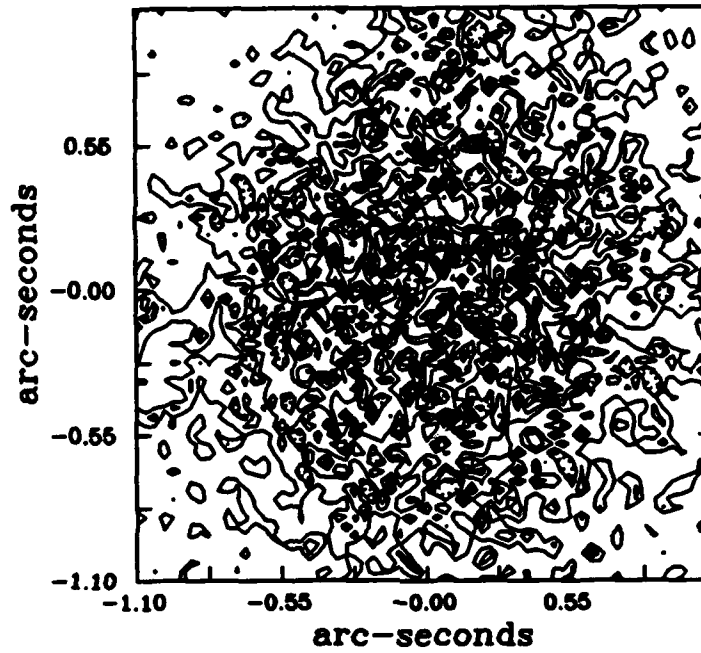


**Figure A.7** Asteroid Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^6$ .

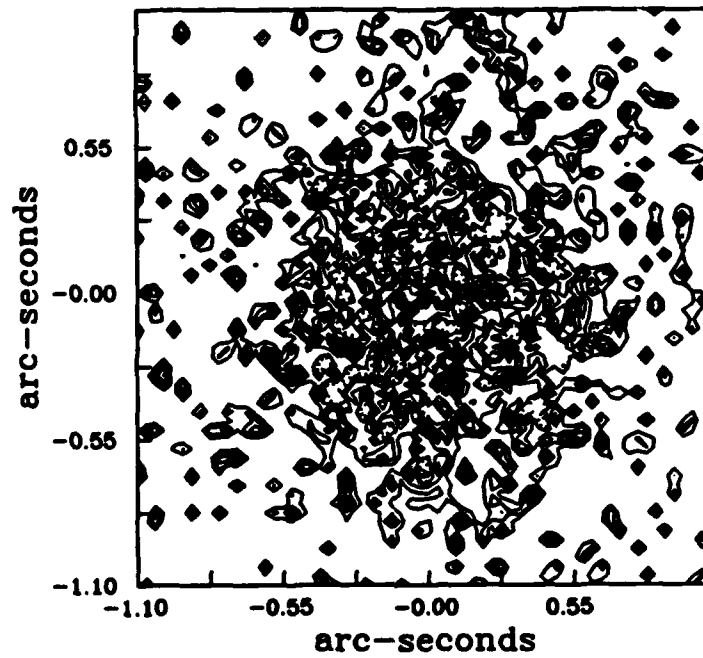


**Figure A.8** Asteroid Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^5$ .

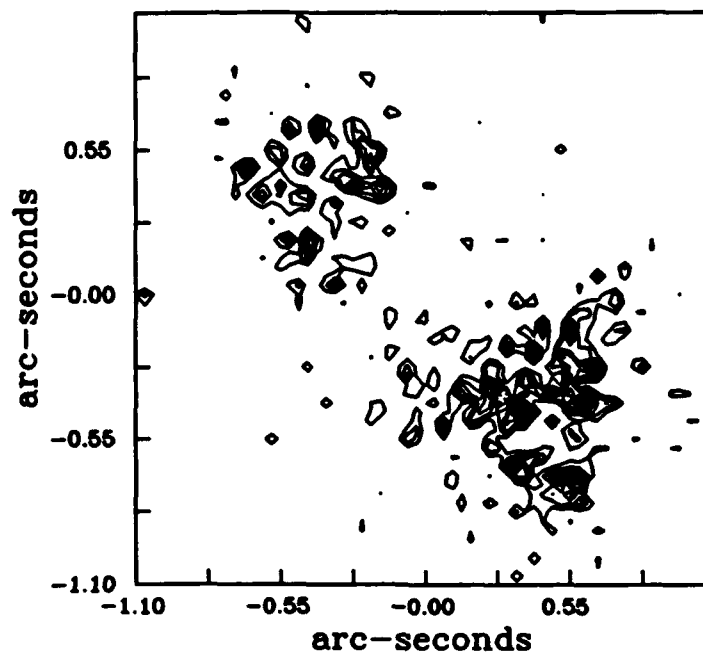




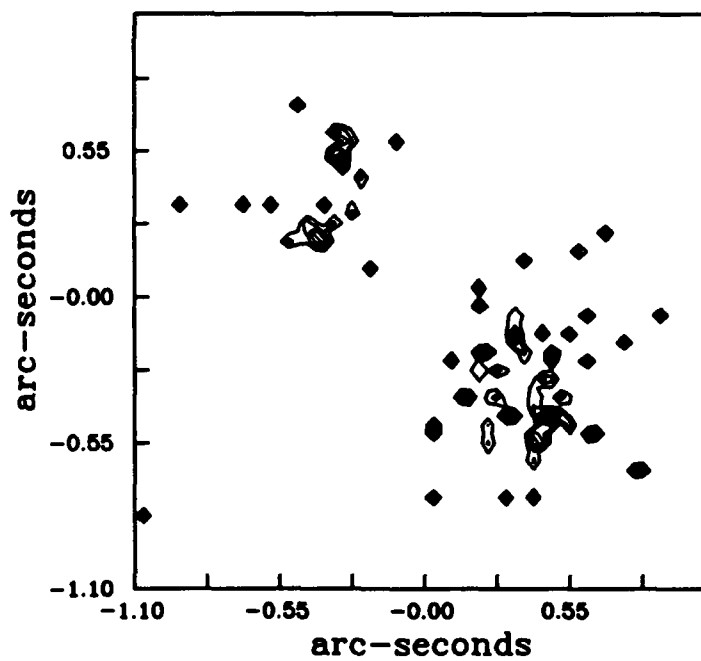
**Figure A.9** Asteroid Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^4$ .



**Figure A.10** Asteroid Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^3$ .



**Figure A.11** Binary Star Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^3$ .



**Figure A.12** Binary Star Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^2$ .

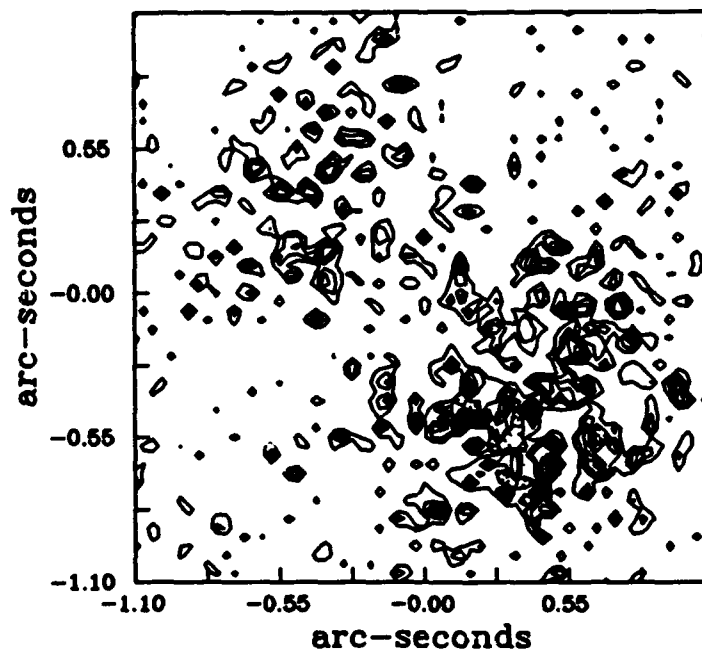


Figure A.13 Binary Star Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^3$ .

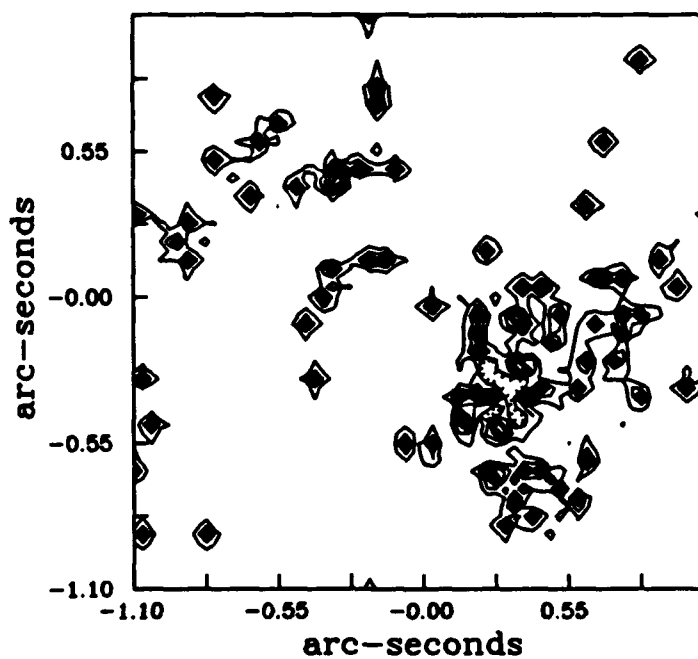
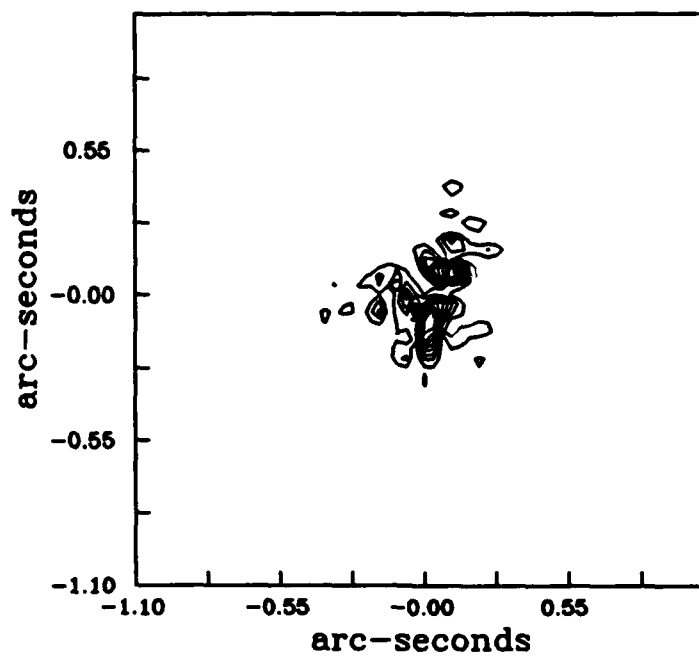
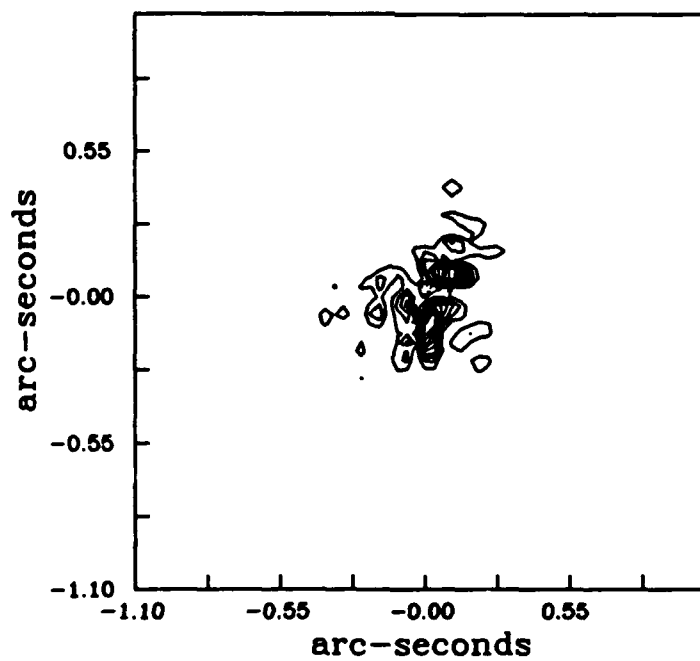


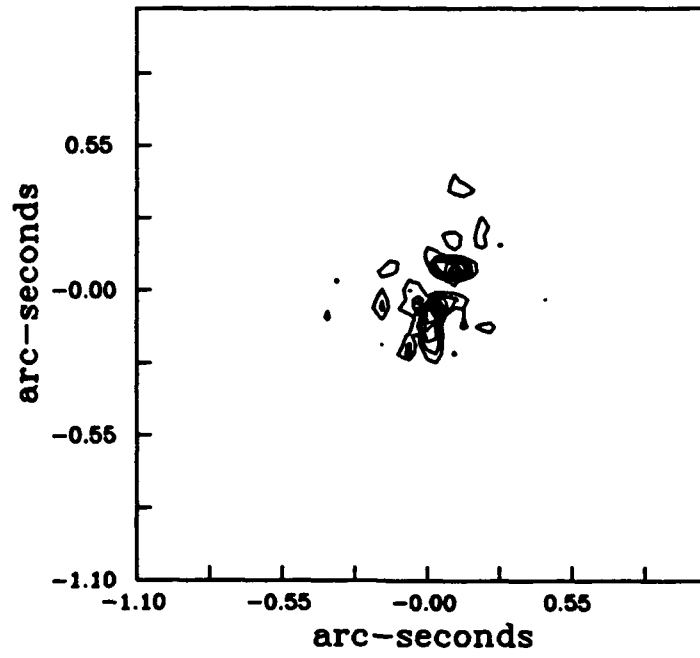
Figure A.14 Binary Star Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^2$ .



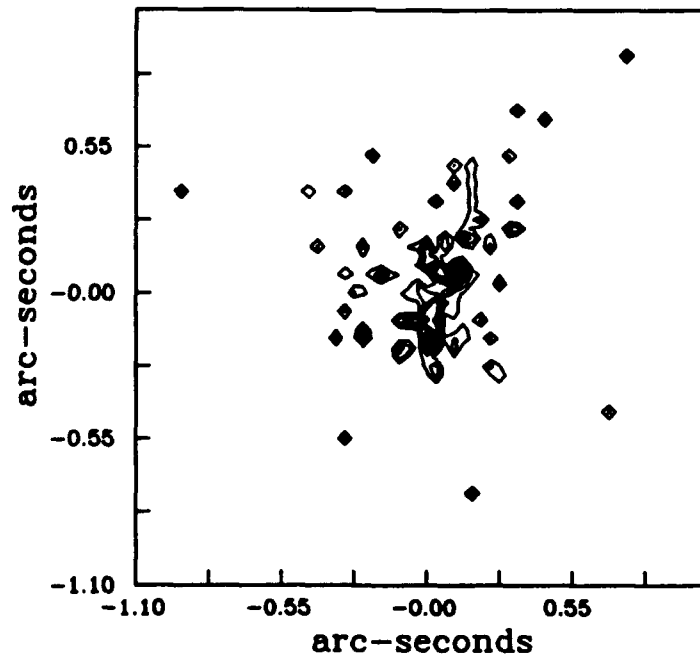
**Figure A.15** Star Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ .



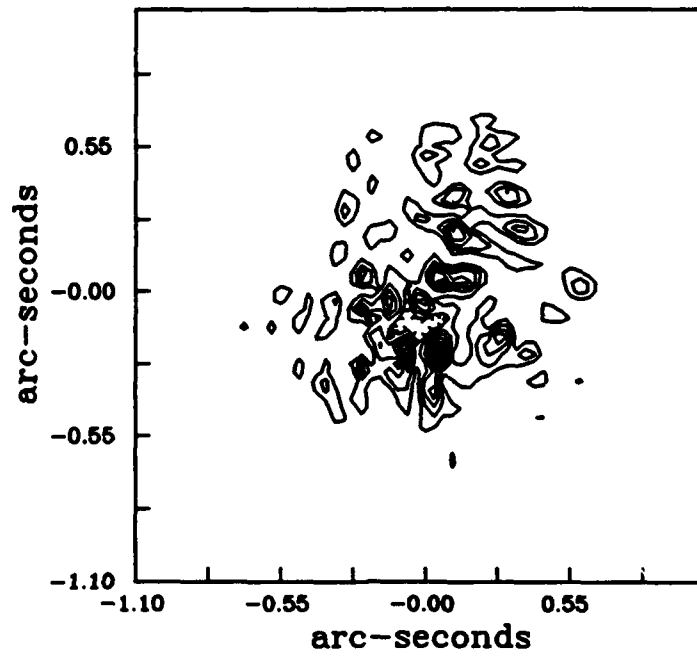
**Figure A.16** Star Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^4$ .



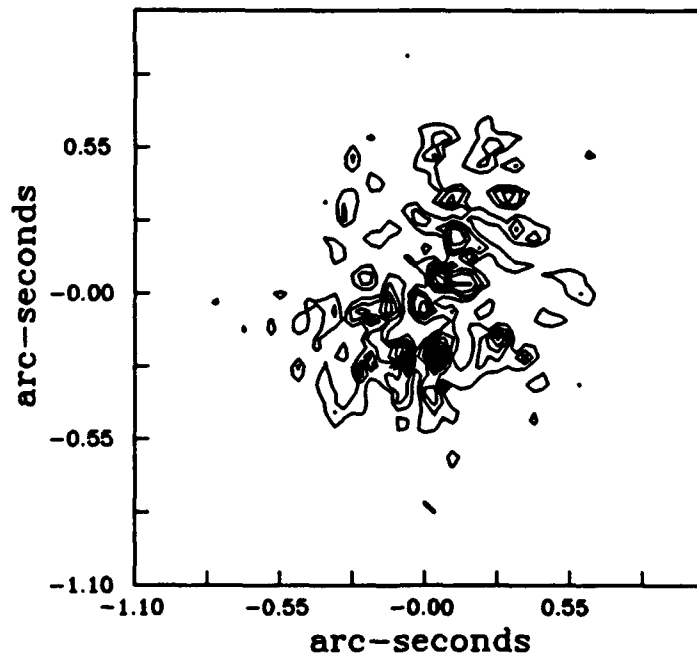
**Figure A.17** Star Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^3$ .



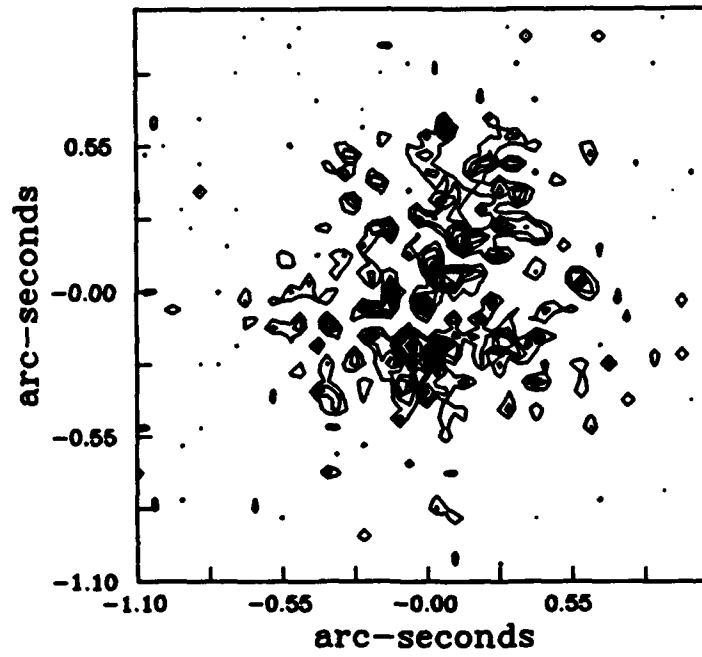
**Figure A.18** Star Short Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^2$ .



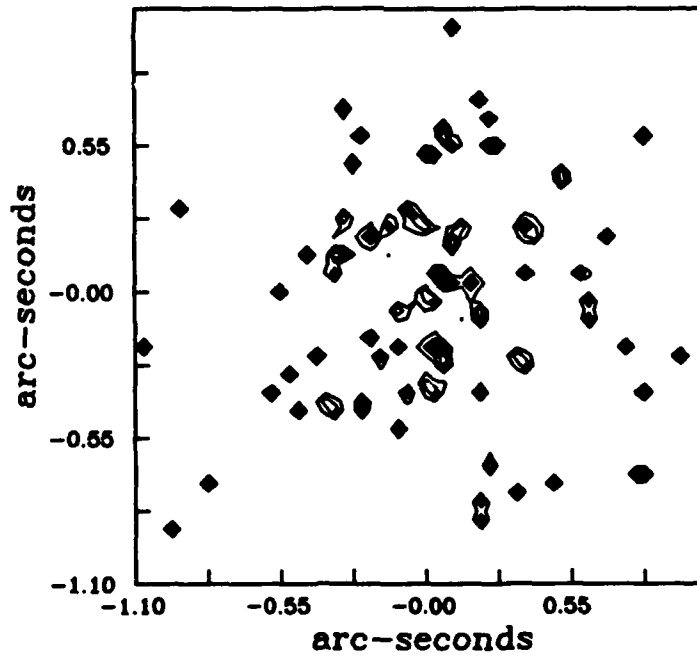
**Figure A.19** Star Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^5$ .



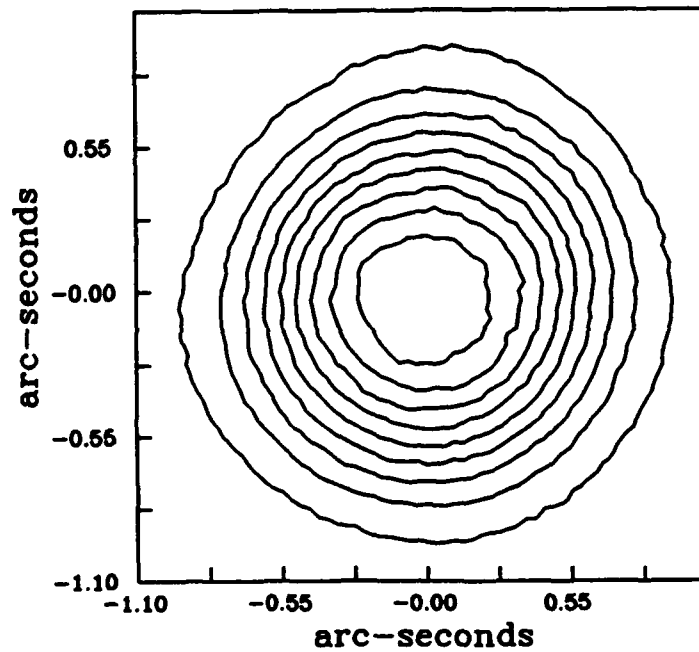
**Figure A.20** Star Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^4$ .



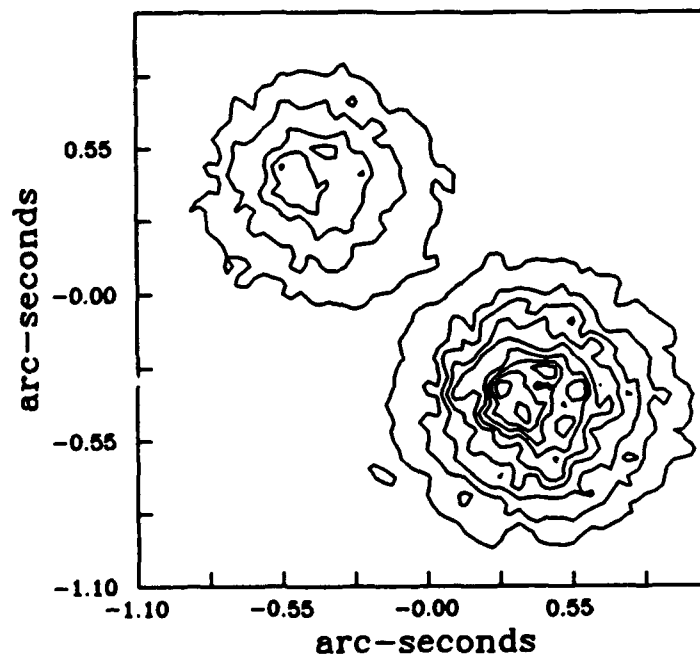
**Figure A.21** Star Short Exposure Image,  
 $r_0 = 0.103$ ,  $N_p = 10^3$ .



**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,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ .



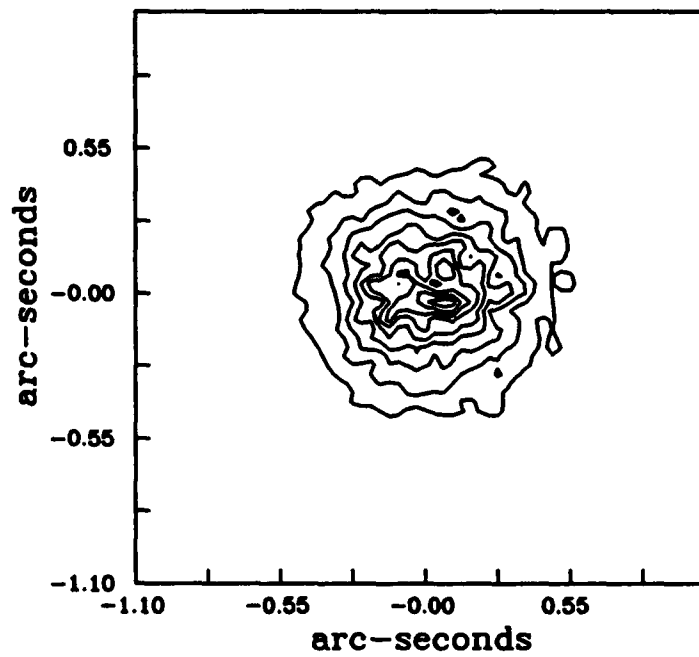
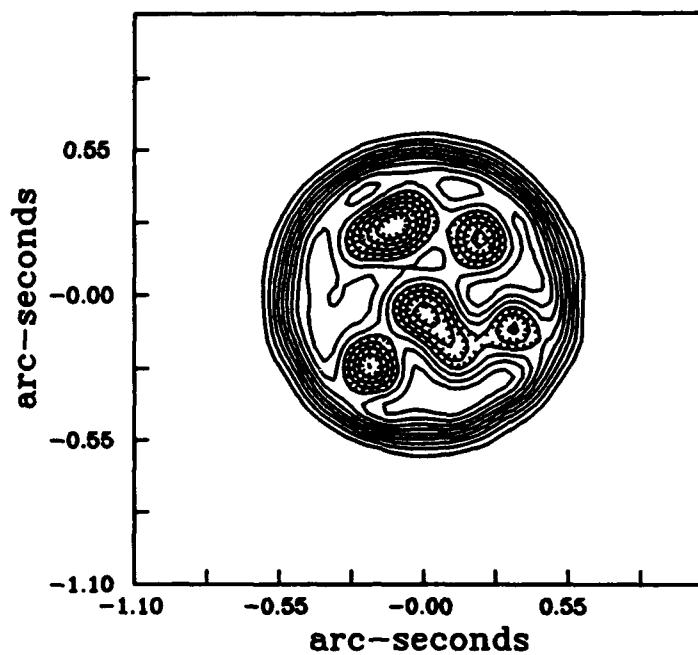
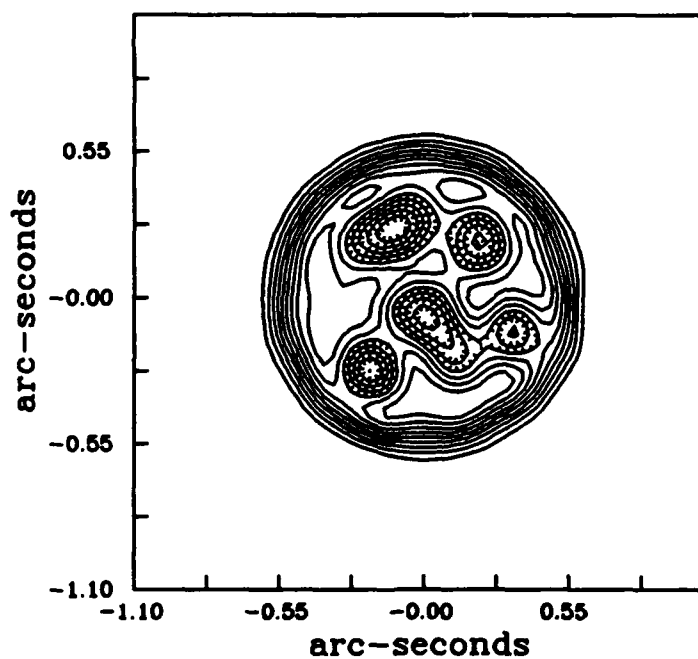


Figure A.25 Star Long Exposure Image,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ .



**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,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ ,  $N_t = 400$ ,  $OS = 4$ .

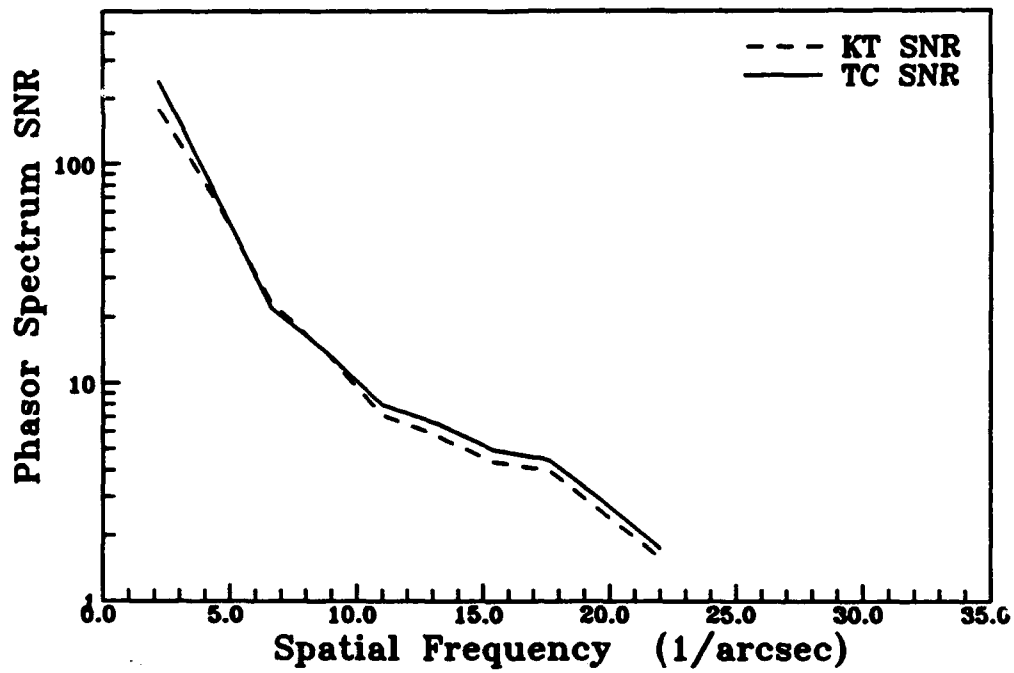


Figure A.28 Asteroid Baseline Phasor Spectrum SNR.

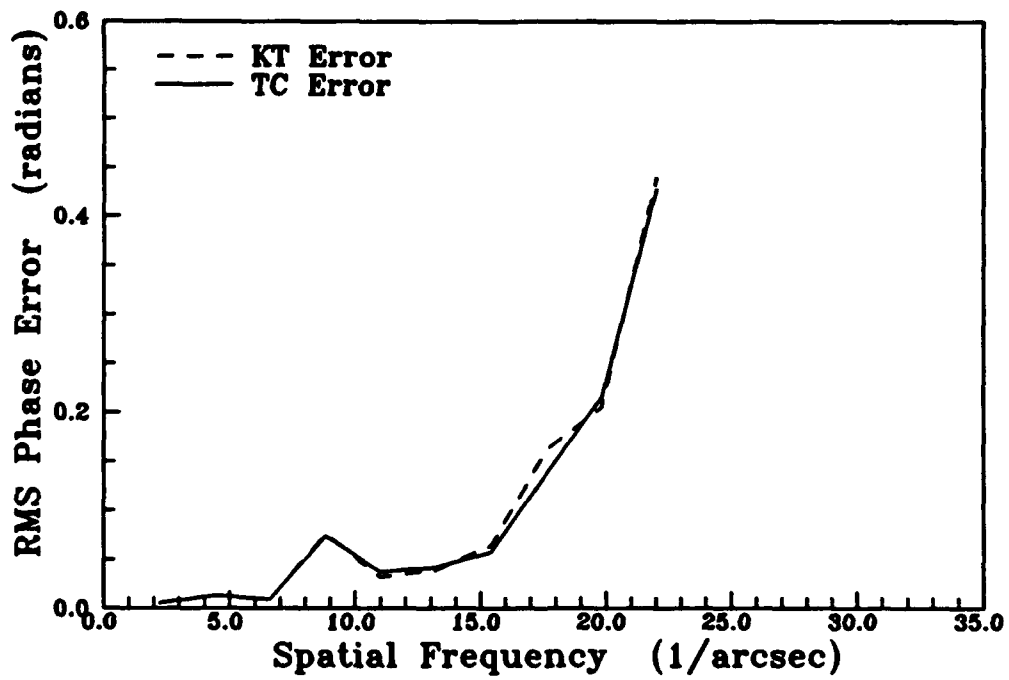
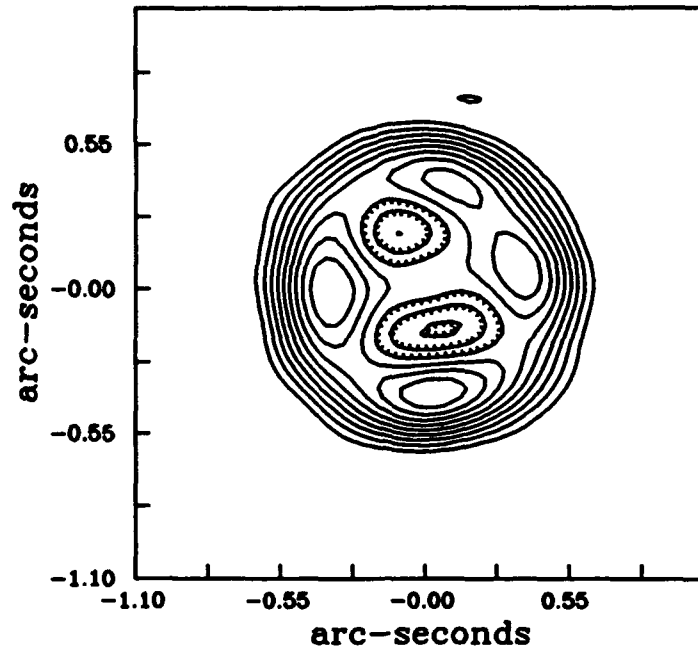
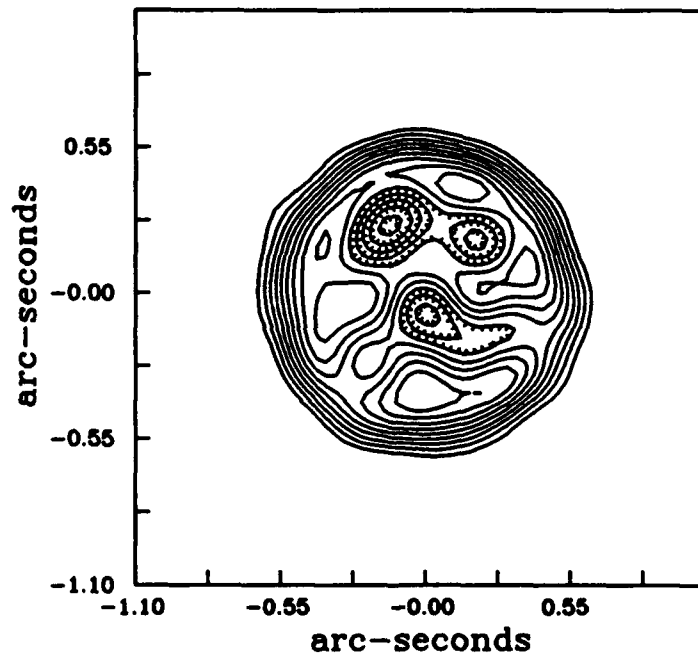


Figure A.29 Asteroid Baseline Azimuthal RMS Phase Error.



**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.

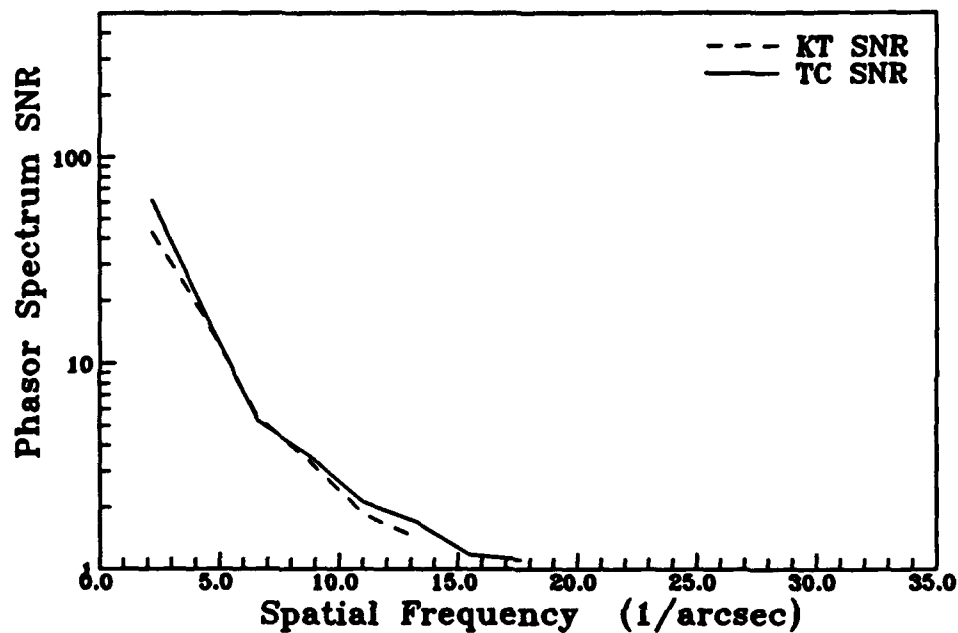


Figure A.32 Asteroid Phasor Spectrum SNR,  $N_f = 25$ .

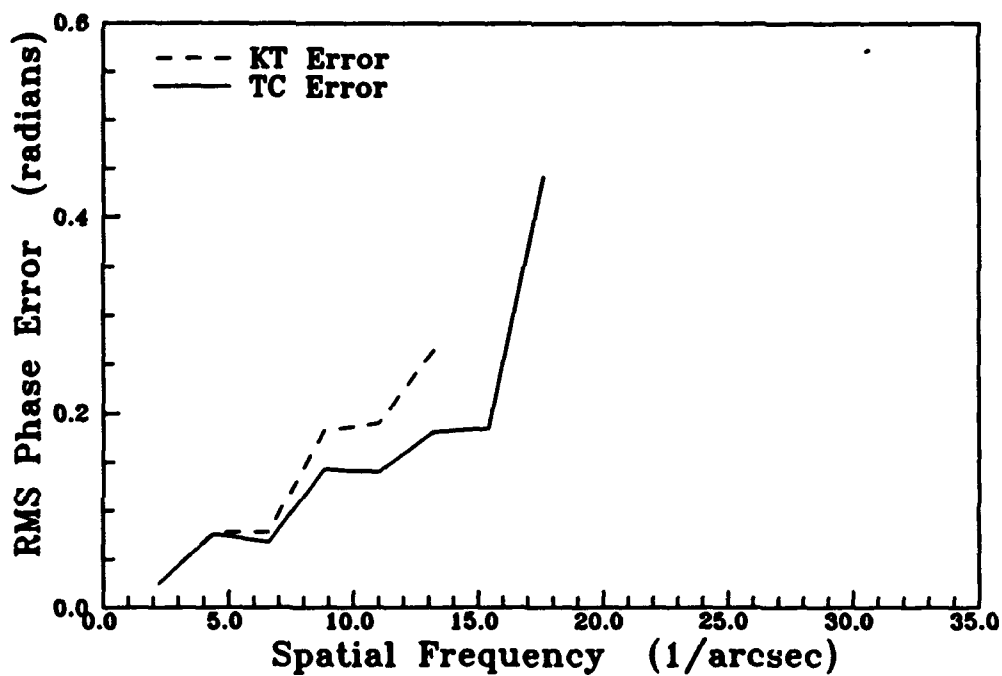


Figure A.33 Asteroid Azimuthal RMS Phase Error,  $N_f = 25$ .

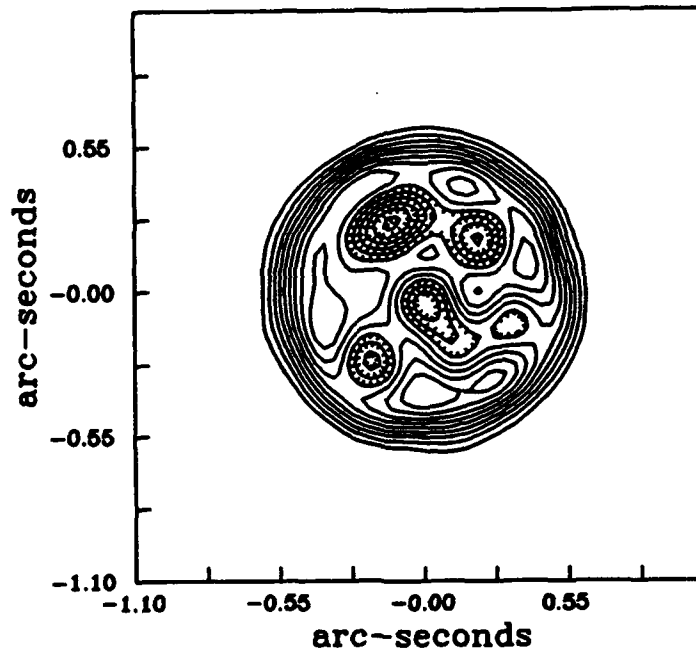


Figure A.34 KT Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ ,  $N_t = 100$ , OS = 4.

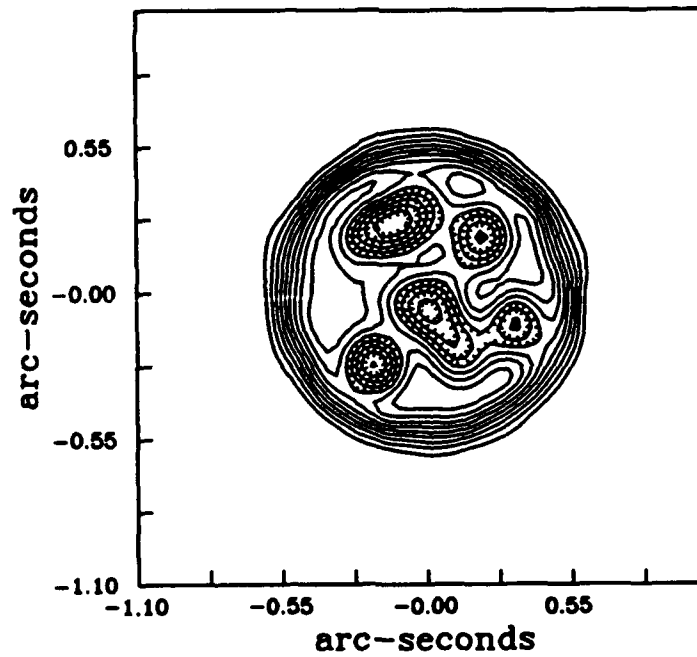


Figure A.35 TC Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ ,  $N_t = 100$ , OS = 4.

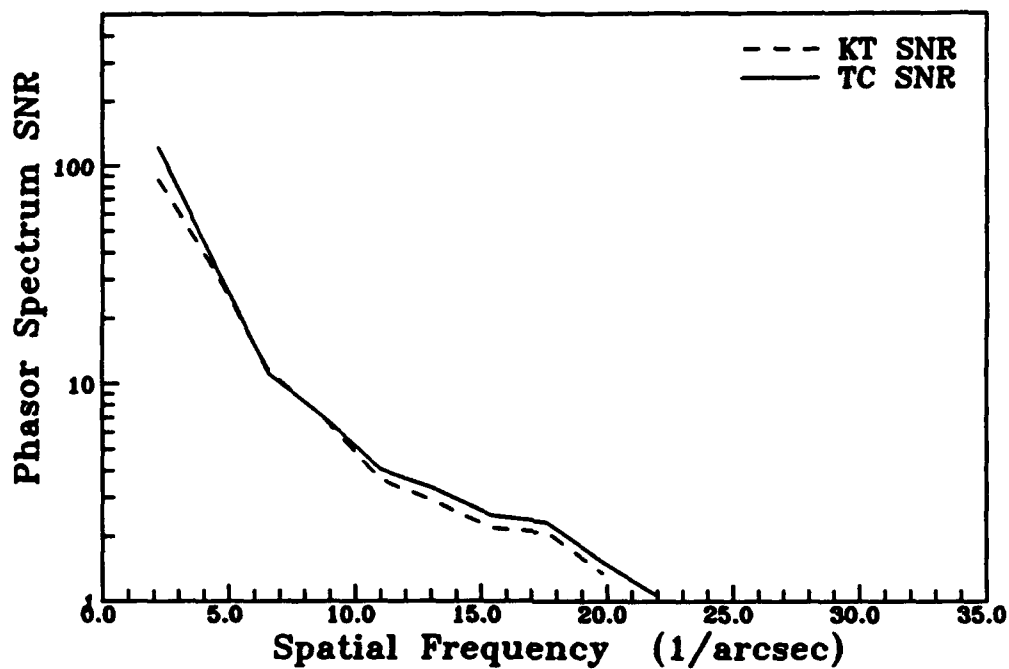


Figure A.36 Asteroid Phasor Spectrum SNR,  $N_f = 100$ .

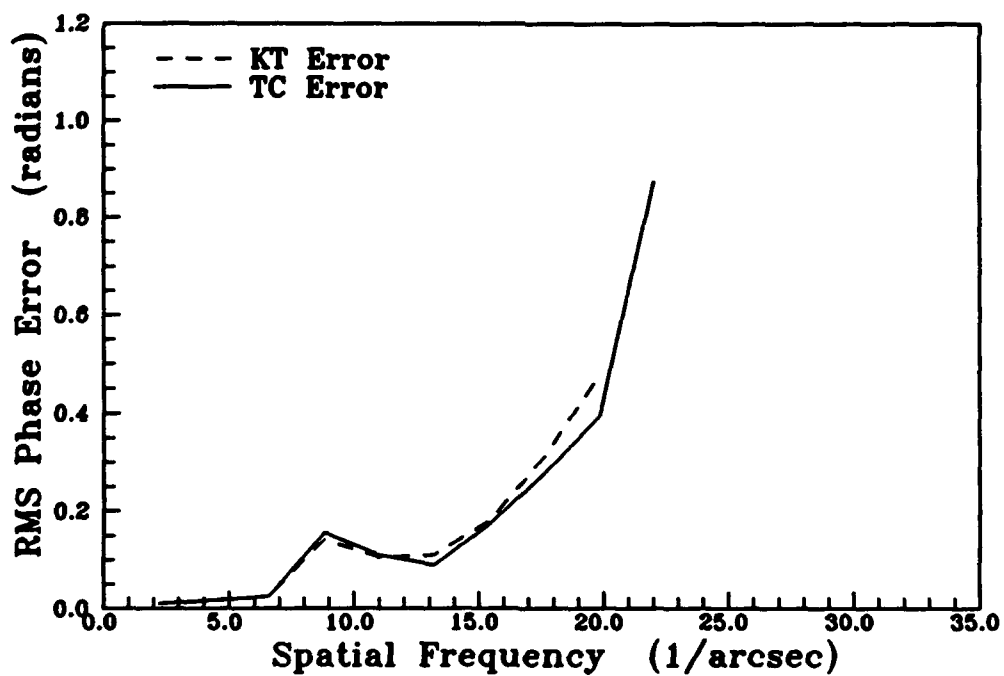
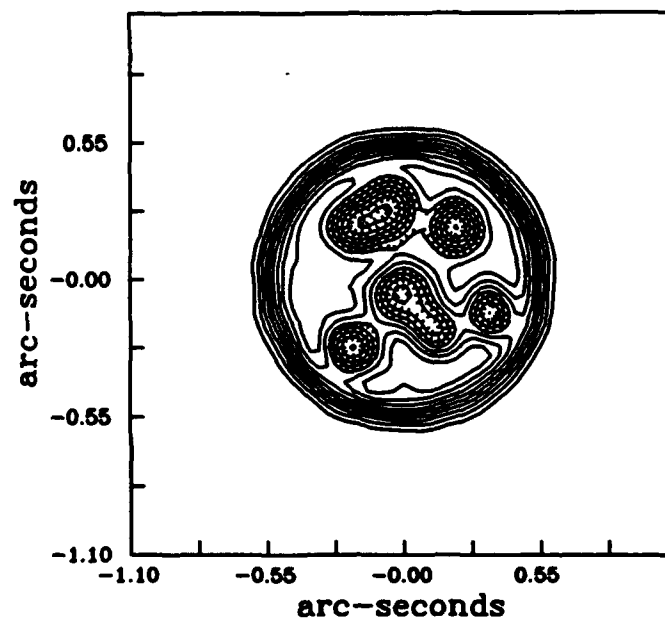
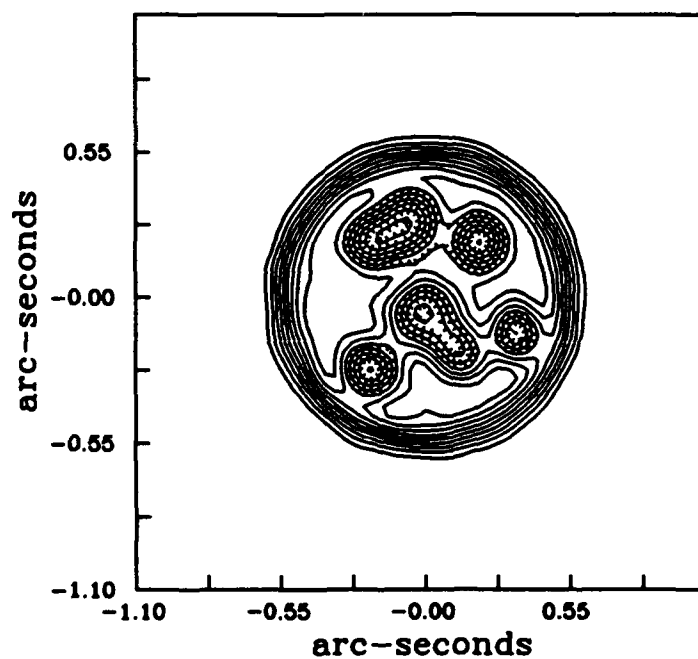


Figure A.37 Asteroid Azimuthal RMS Phase Error,  $N_f = 100$ .



**Figure A.38** KT Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ ,  $N_t = 1600$ , OS = 4.



**Figure A.39** TC Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ ,  $N_t = 1600$ , OS = 4.



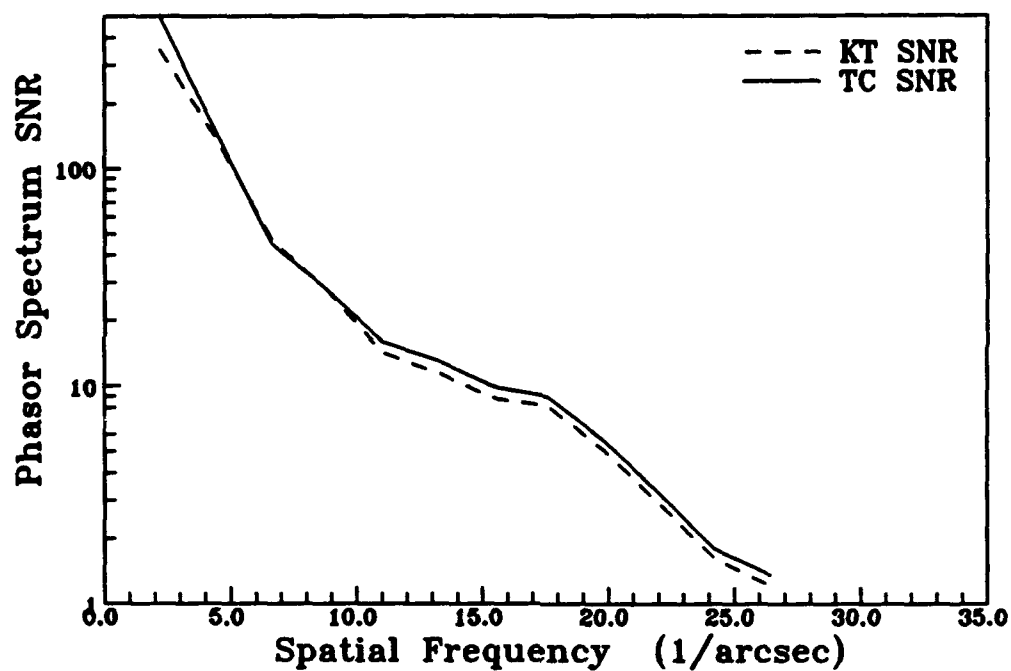


Figure A.40 Asteroid Phasor Spectrum SNR,  $N_t = 1600$ .

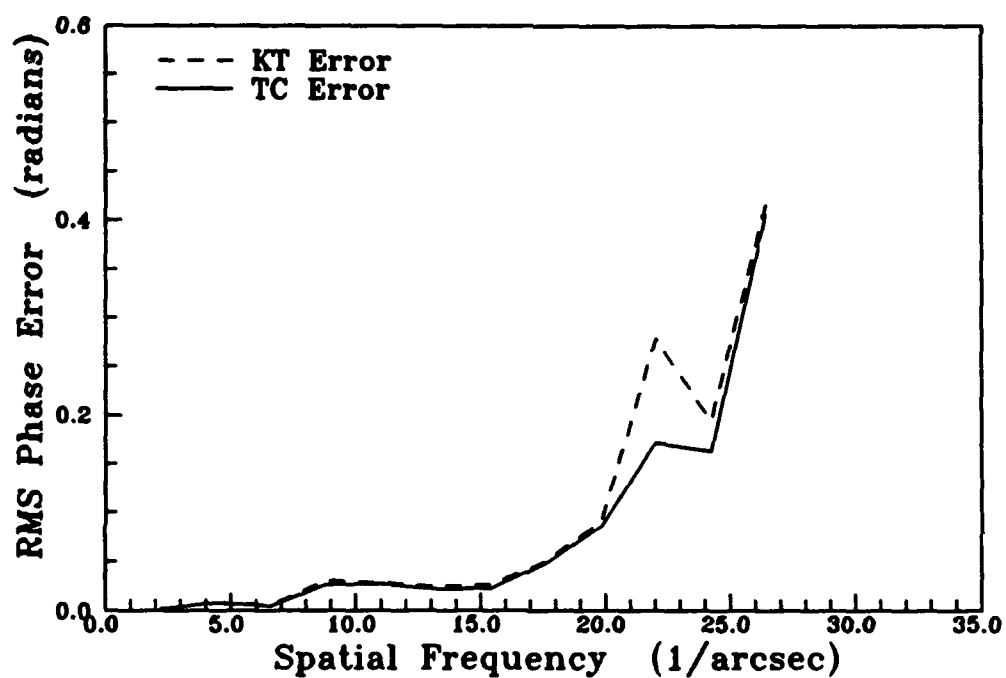
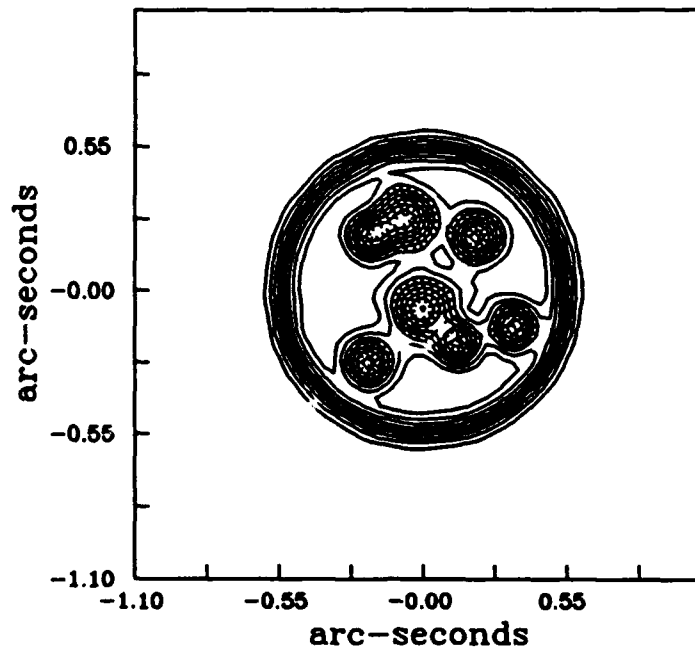
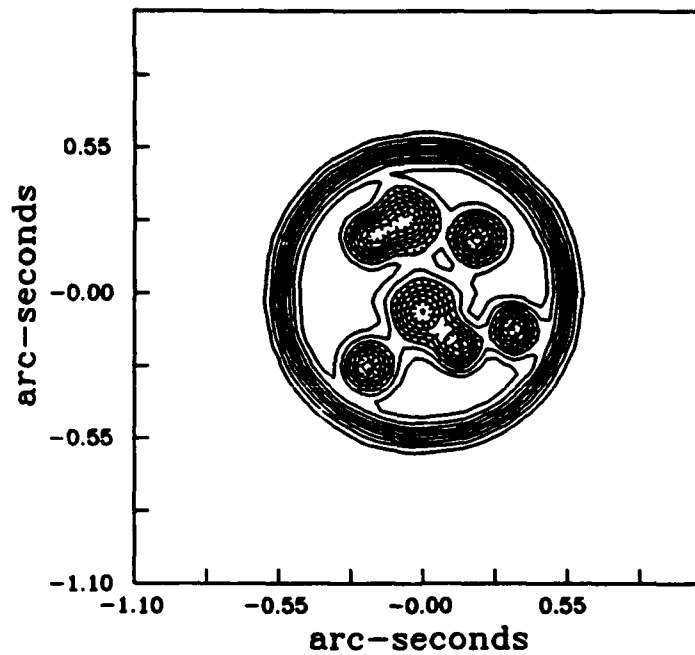


Figure A.41 Asteroid Azimuthal RMS Phase Error,  $N_t = 1600$ .



**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.

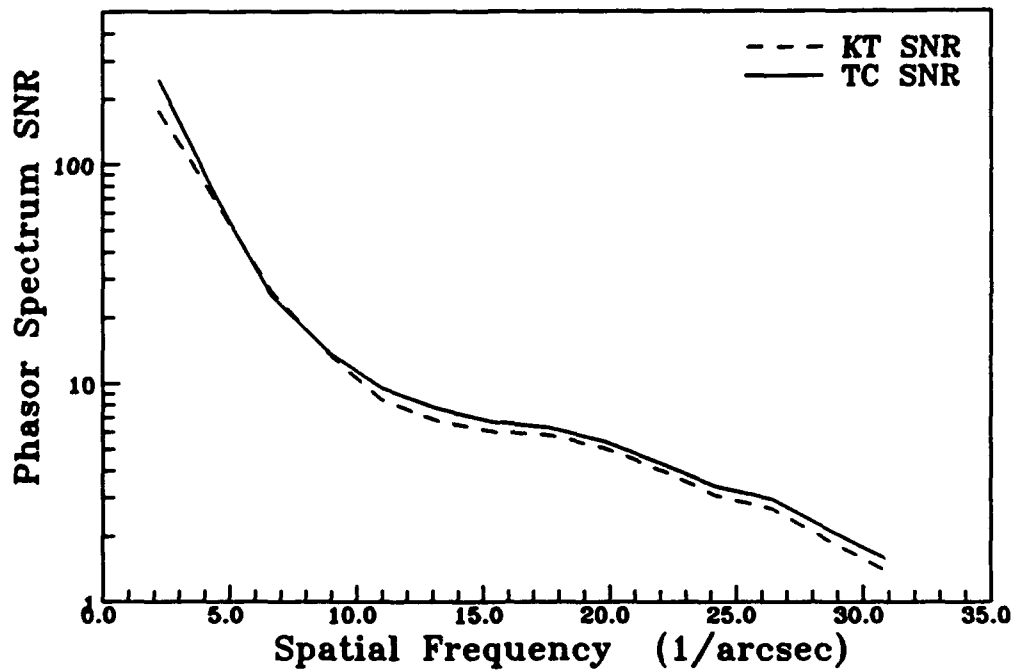


Figure A.44 Asteroid Phasor Spectrum SNR,  $N_p = 10^6$ .

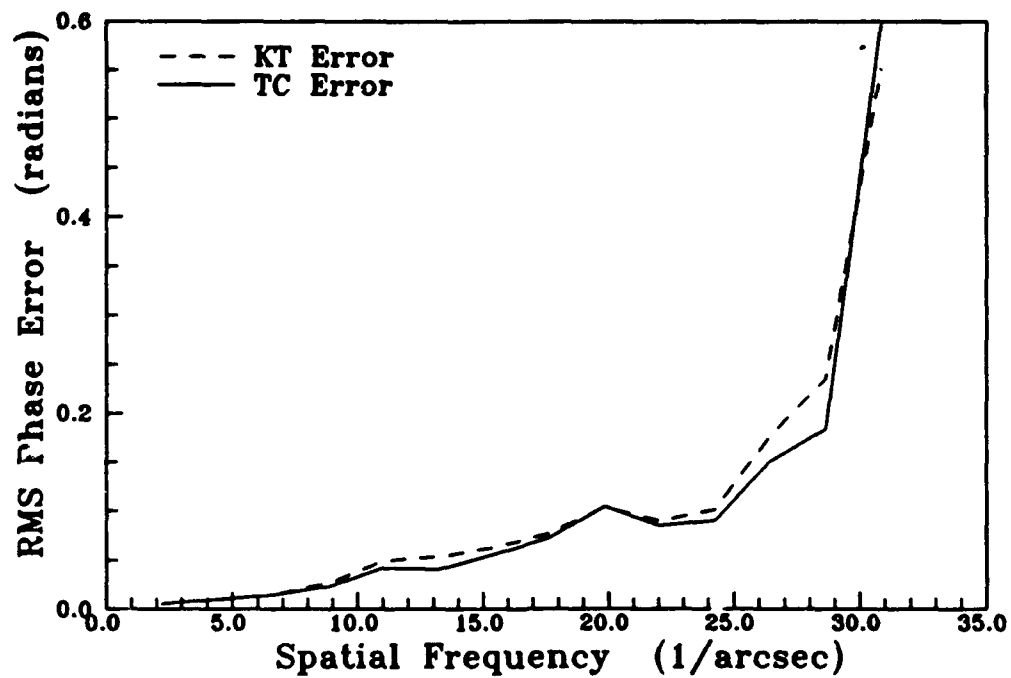
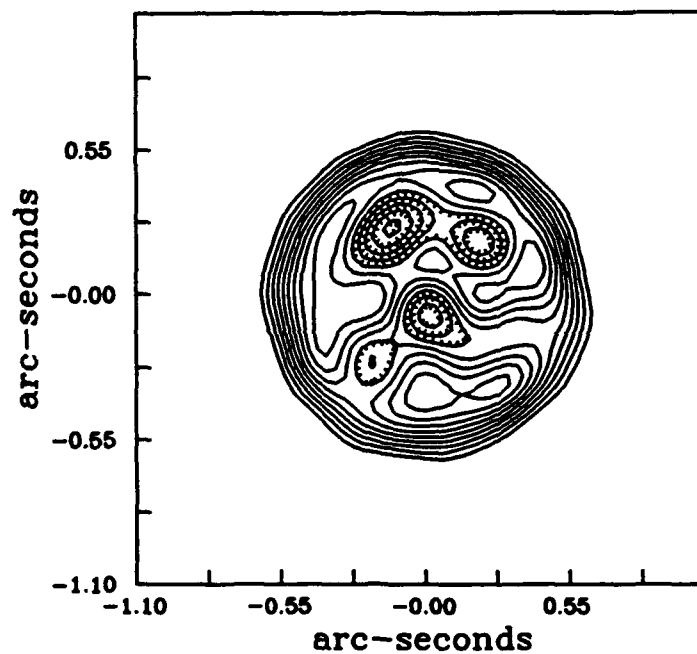
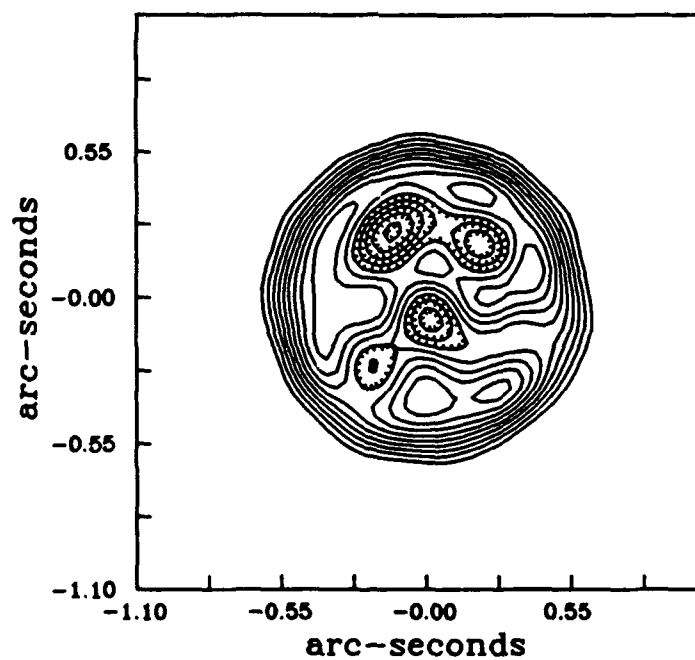


Figure A.45 Asteroid Azimuthal RMS Phase Error,  $N_p = 10^6$ .



**Figure A.46** KT Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^4$ ,  $N_t = 400$ , OS = 4.



**Figure A.47** TC Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^4$ ,  $N_t = 400$ , OS = 4.

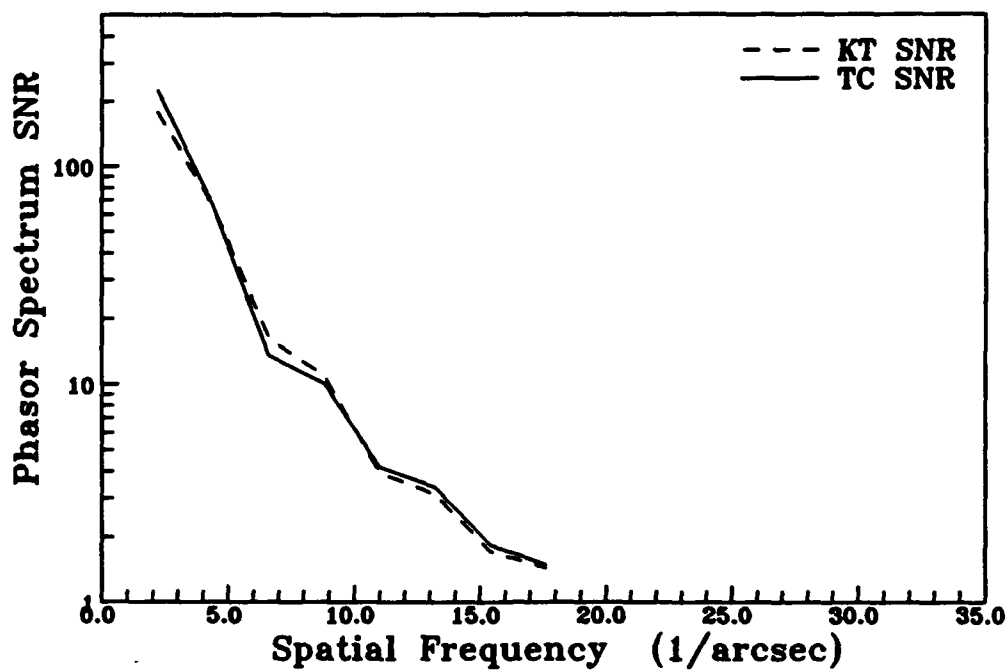


Figure A.48 Asteroid Phasor Spectrum SNR,  $N_p = 10^4$ .

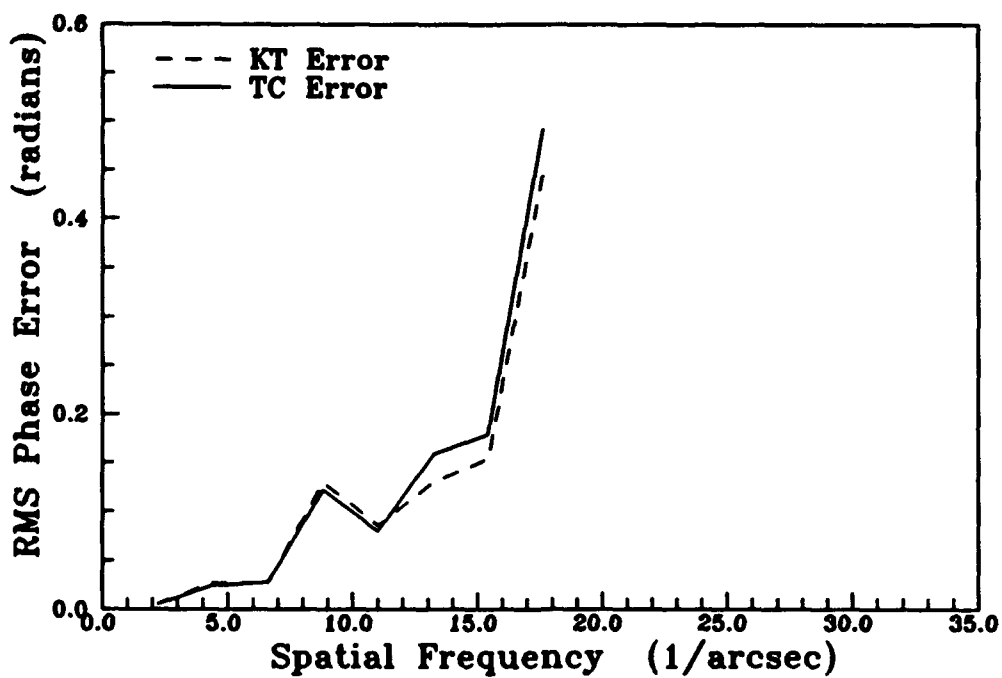
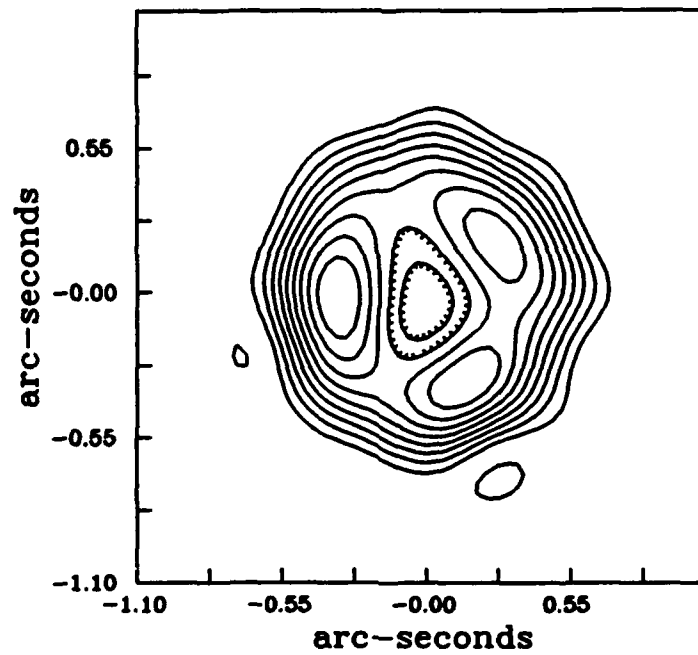
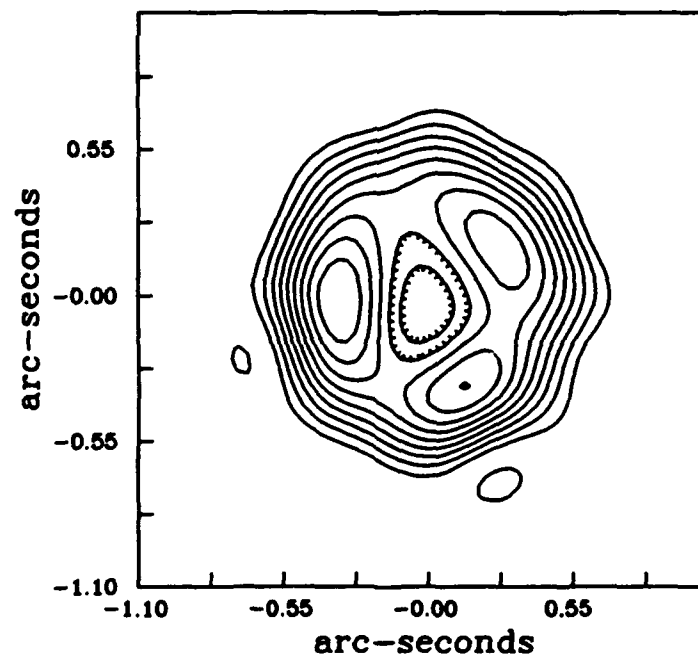


Figure A.49 Asteroid Azimuthal RMS Phase Error,  $N_p = 10^4$ .



**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.

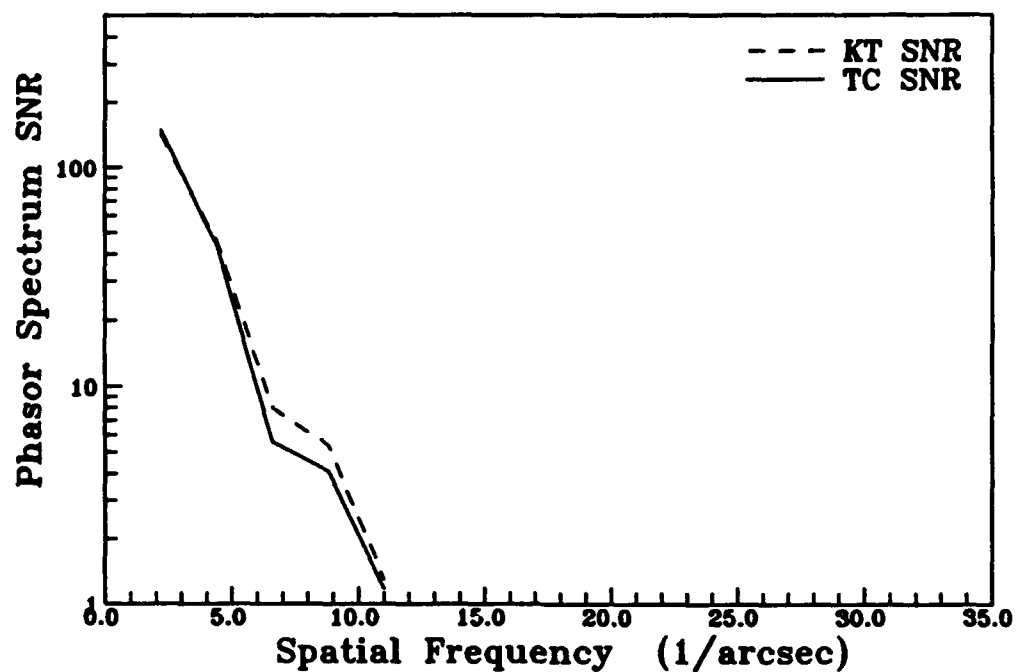


Figure A.52 Asteroid Phasor Spectrum SNR,  $N_p = 10^3$ .

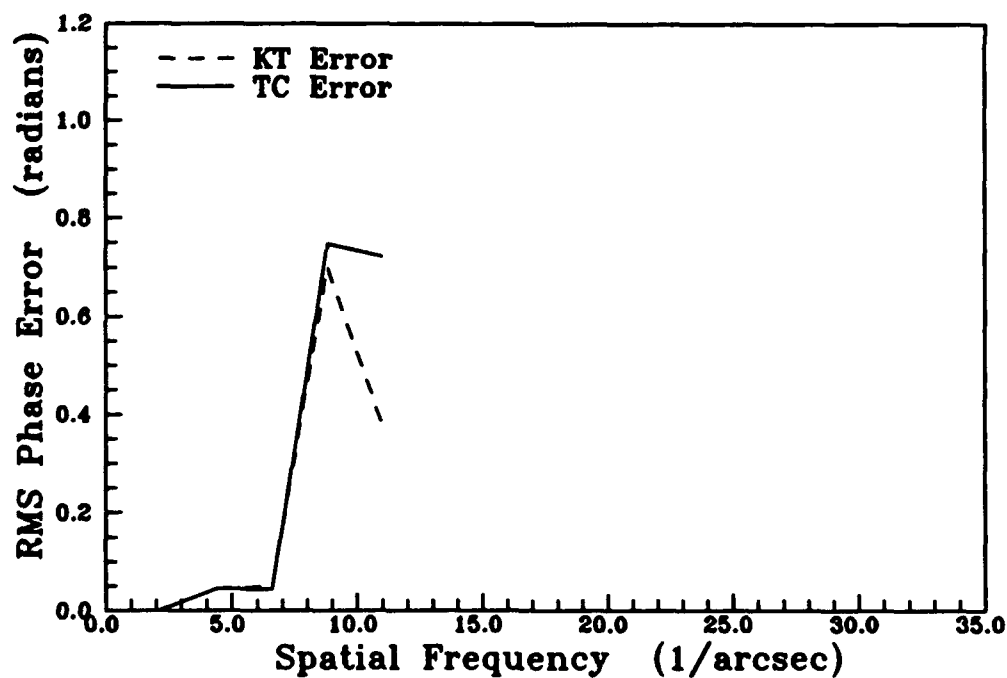
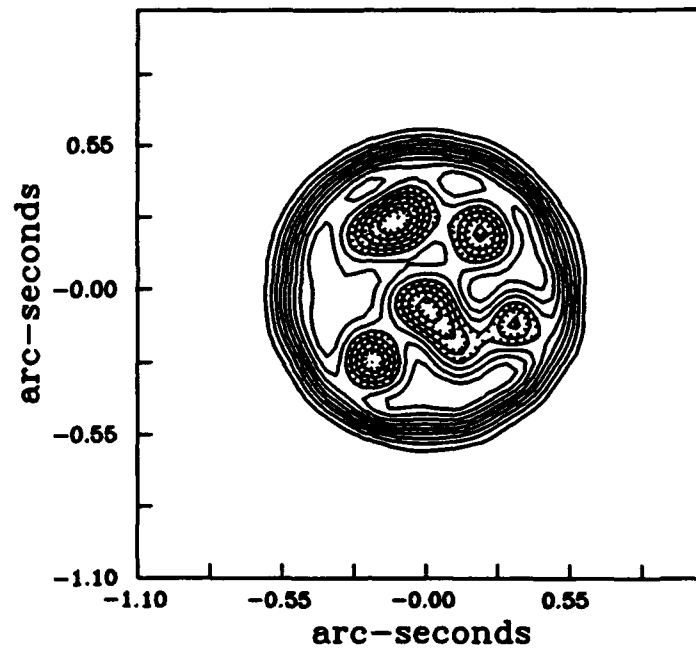
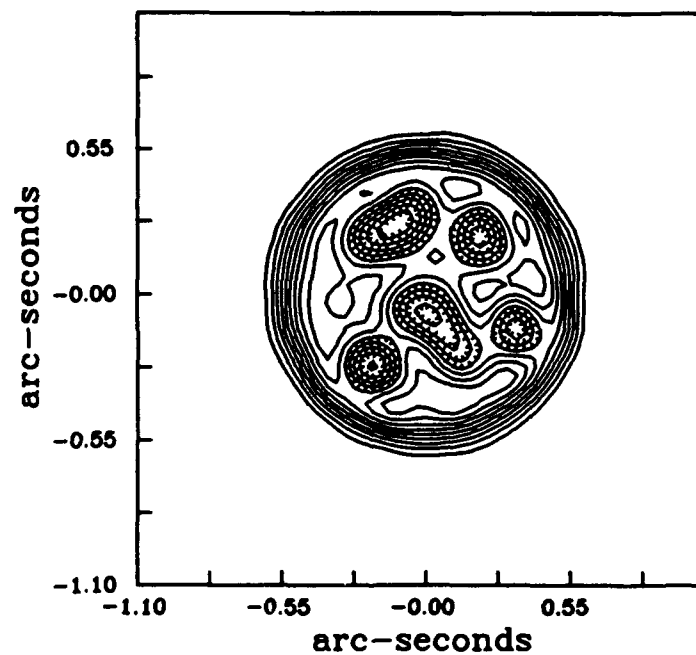


Figure A.53 Asteroid Azimuthal RMS Phase Error,  $N_p = 10^3$ .



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



**Figure A.55** TC Recovered Asteroid, No Tilt,  
 $r_0 = 0.206$ ,  $N_p = 10^5$ ,  $N_t = 400$ , OS = 2.



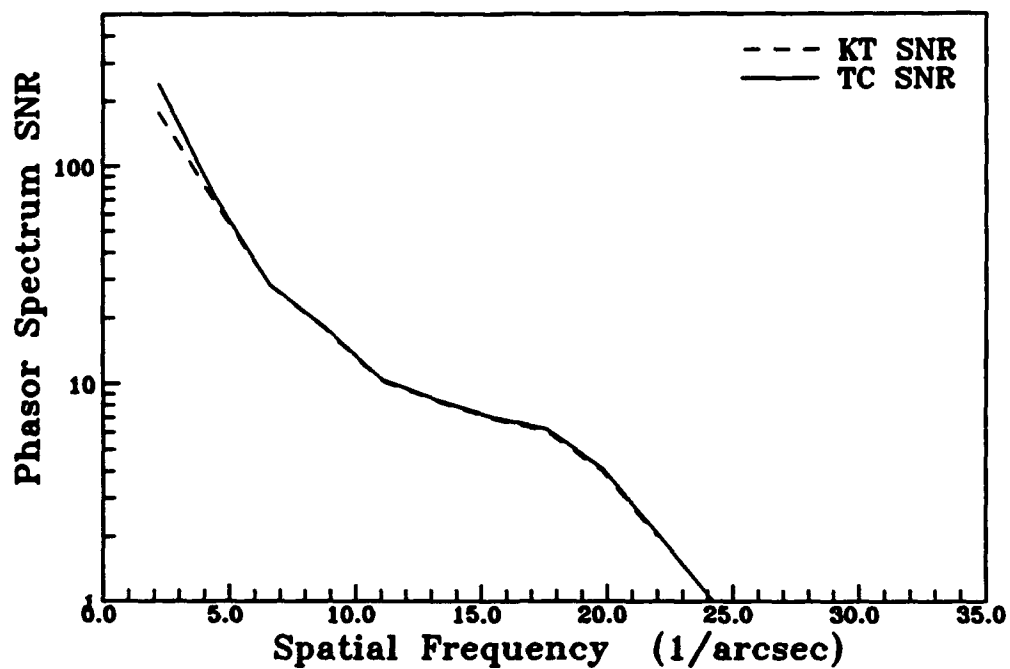


Figure A.56 Asteroid Phasor Spectrum SNR, OS = 2.

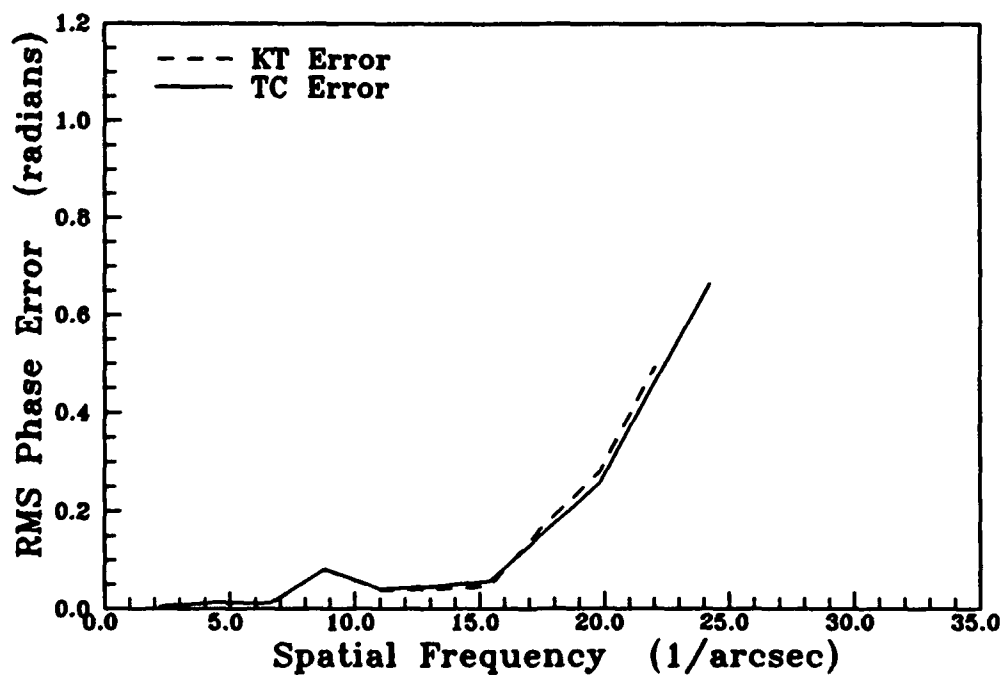
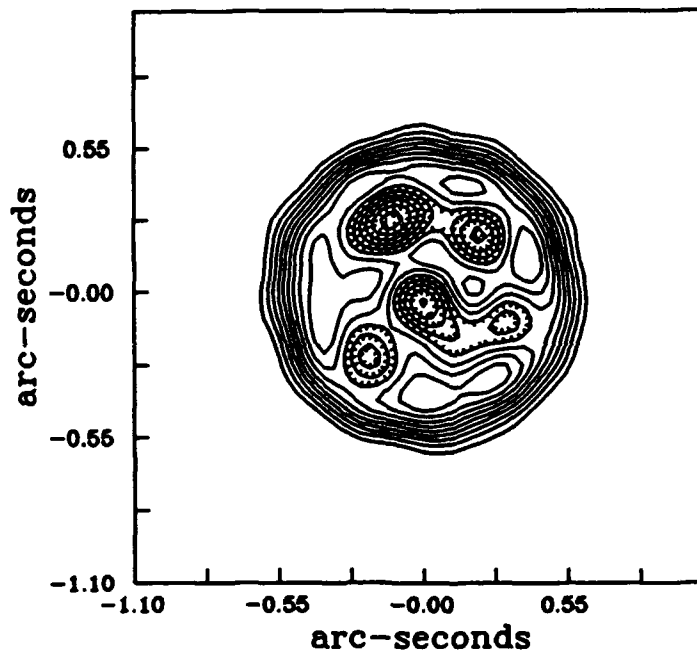
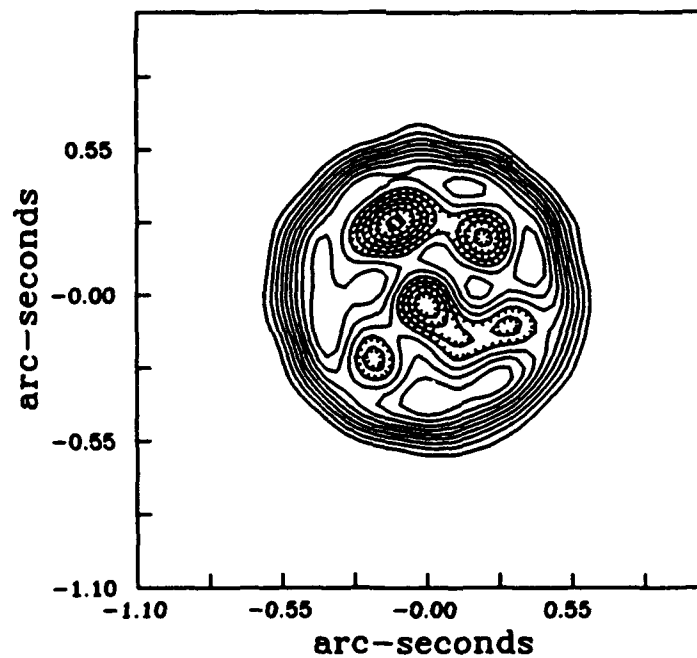


Figure A.57 Asteroid Azimuthal RMS Phase Error, OS = 2.



**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.

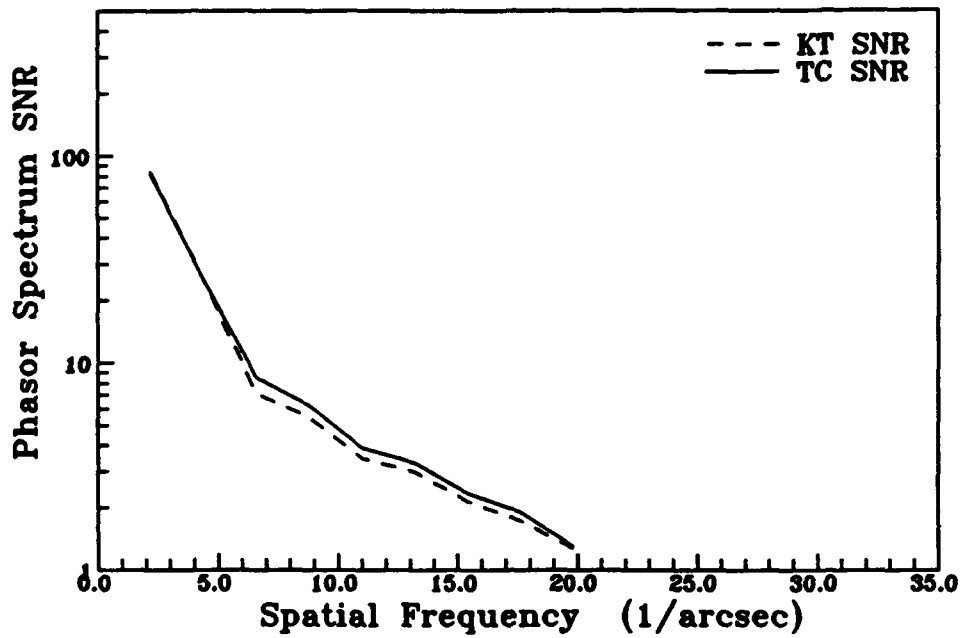


Figure A.60 Asteroid Phasor Spectrum SNR,  
 $r_0 = 0.013$ , OS = 2.

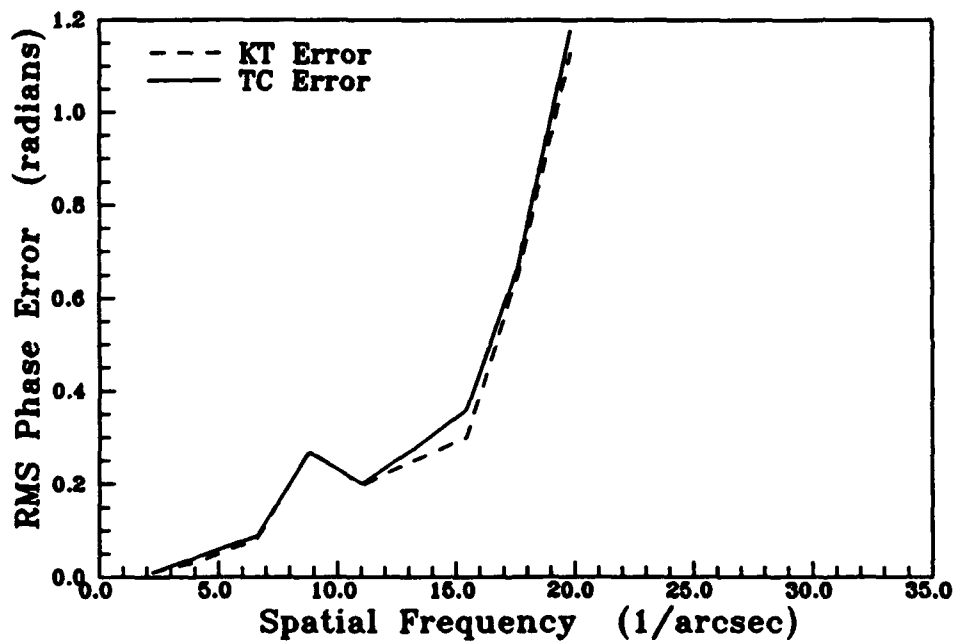
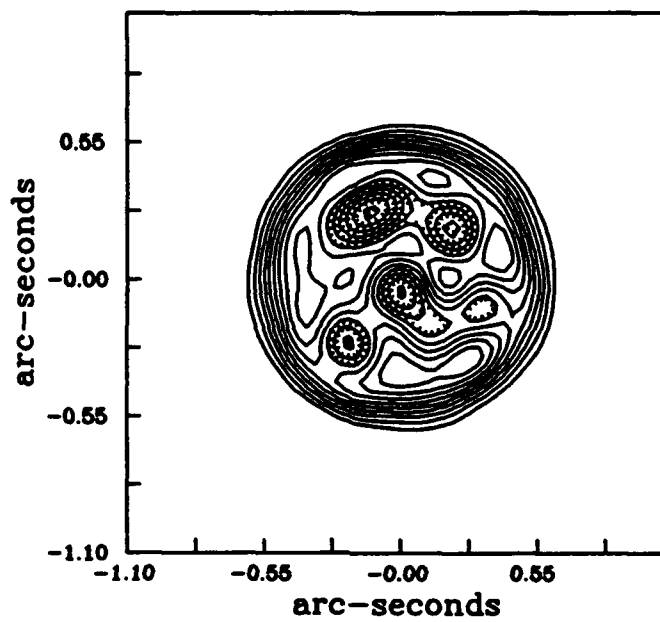
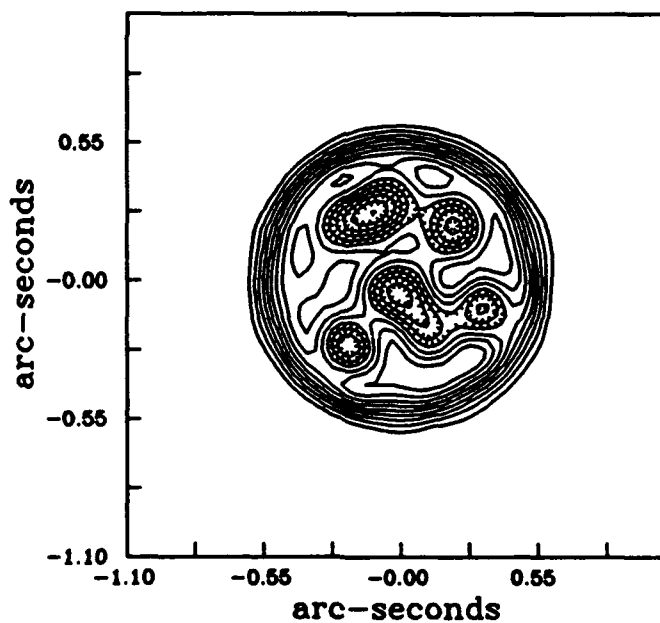


Figure A.61 Asteroid Azimuthal RMS Phase Error,  
 $r_0 = 0.103$ , OS = 2.



**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.

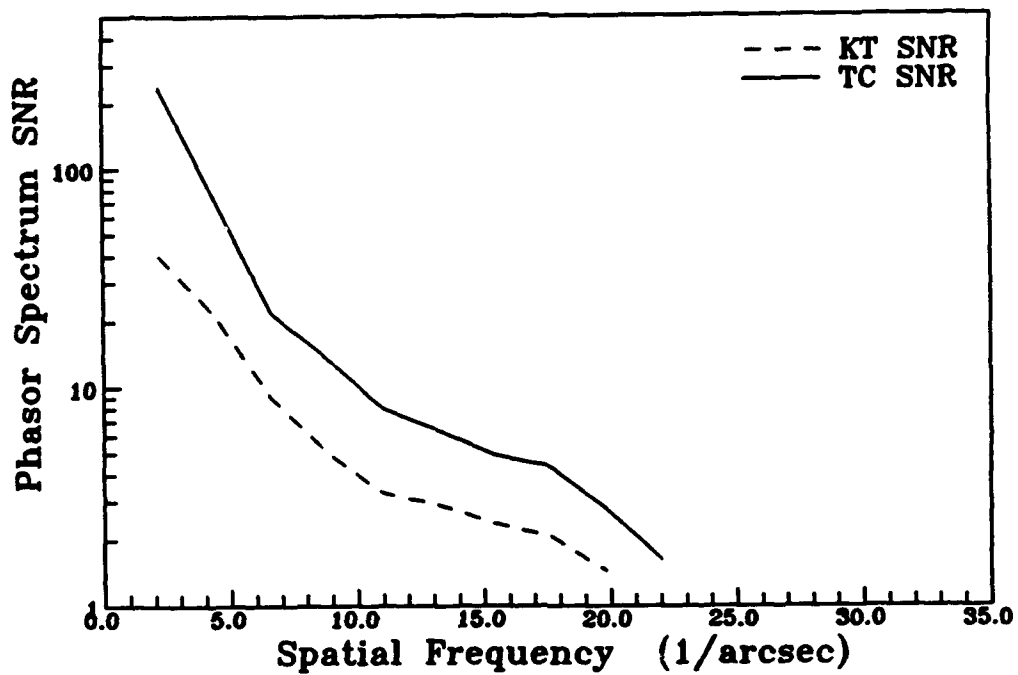


Figure A.64 Asteroid Phasor Spectrum SNR, Tilt.

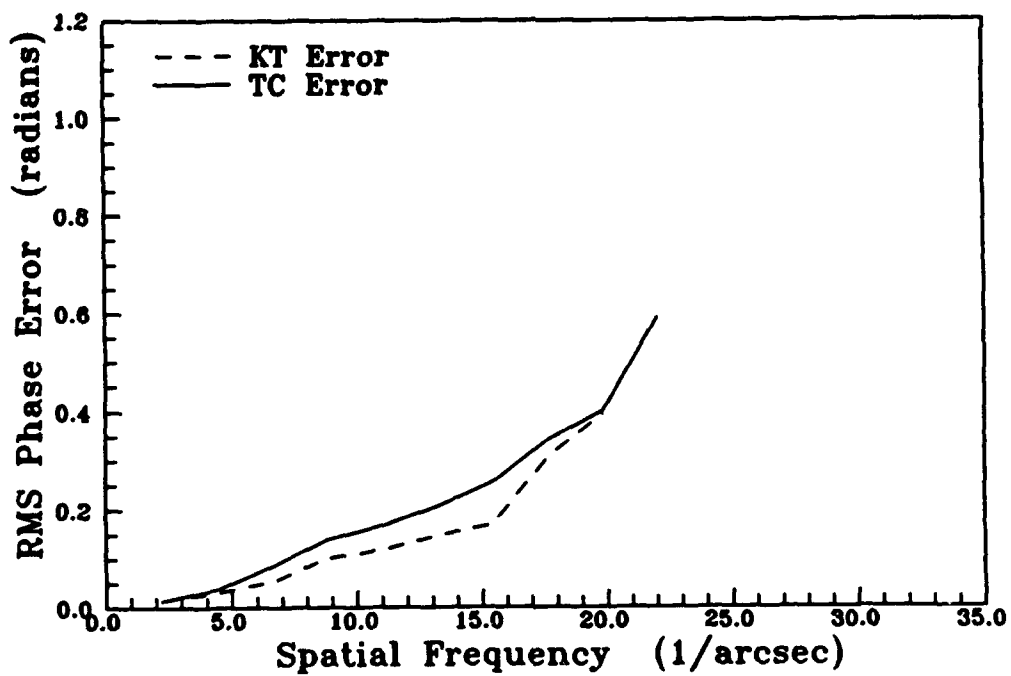
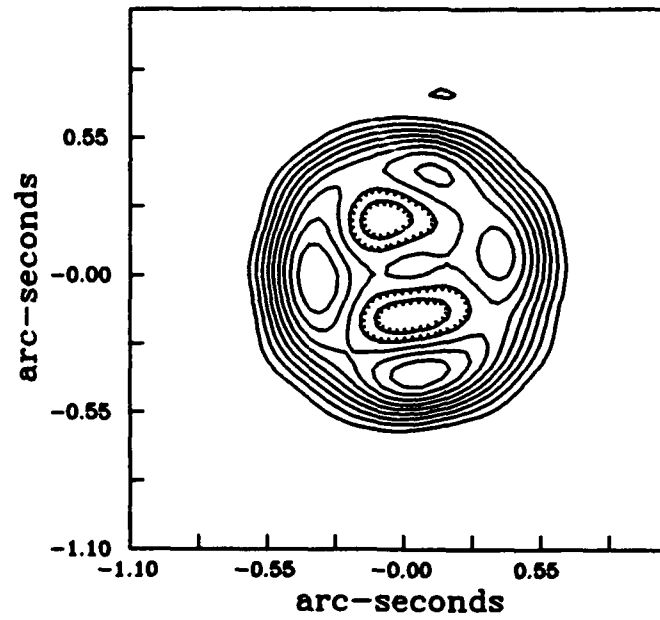
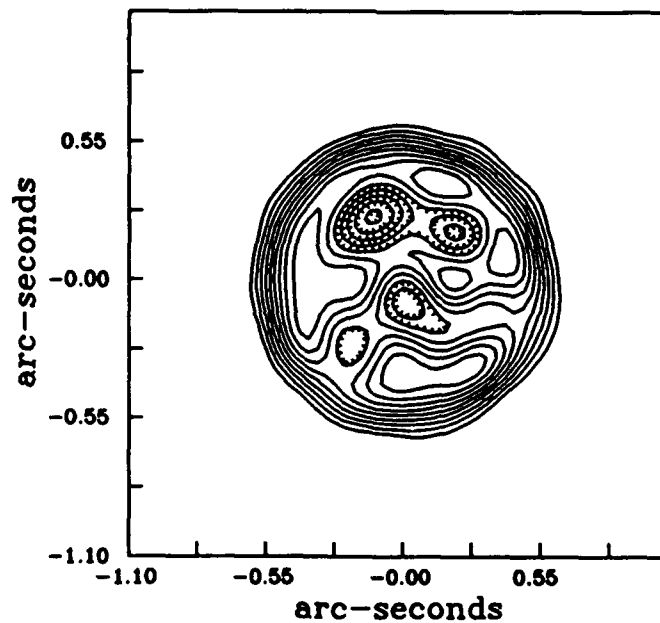


Figure A.65 Asteroid Azimuthal RMS Phase Error, Tilt



**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.

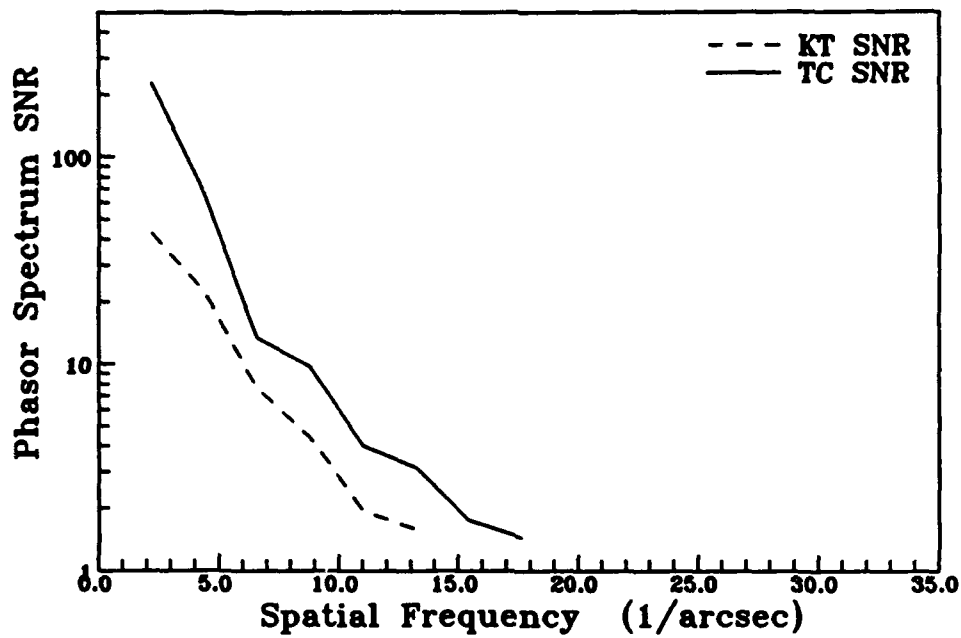


Figure A.68 Asteroid Phasor Spectrum SNR,  
Tilt,  $N_p = 10^4$ .

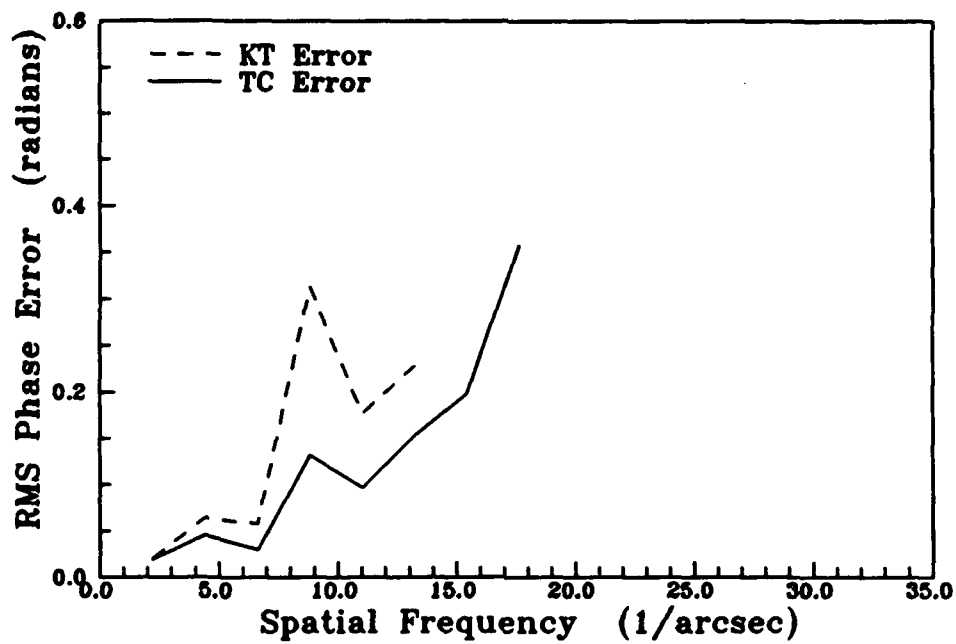
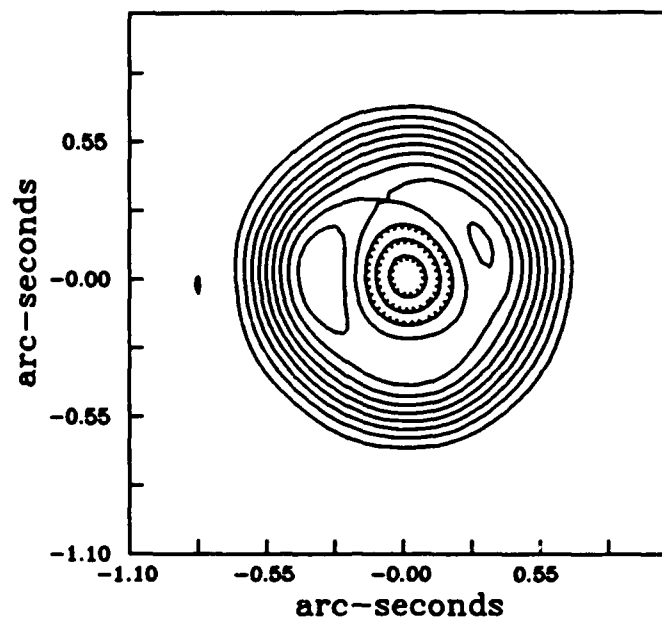
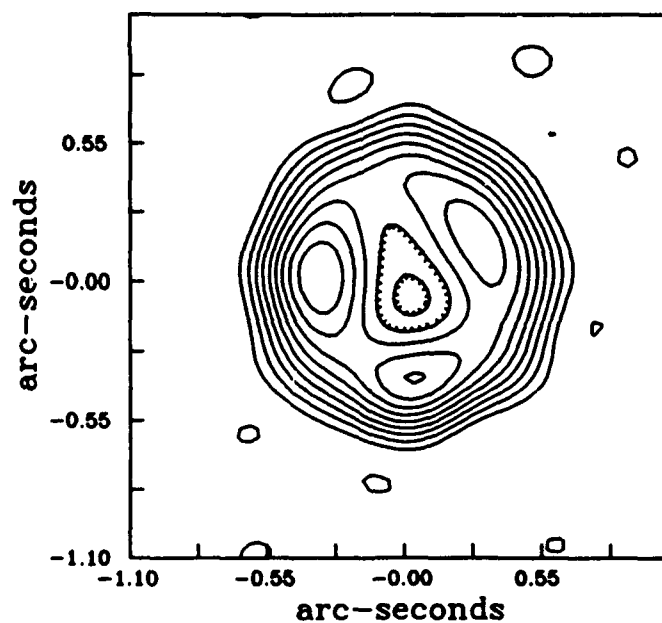


Figure A.69 Asteroid Azimuthal RMS Phase Error,  
Tilt,  $N_p = 10^4$ .



**Figure A.70** KT Recovered Asteroid,  
With Tilt,  $r_0 = 0.206$ ,  $N_p = 10^3$ ,  
 $N_f = 400$ , OS = 4.



**Figure A.71** TC Recovered Asteroid,  
With Tilt,  $r_0 = 0.206$ ,  $N_p = 10^3$ ,  
 $N_f = 400$ , OS = 4.



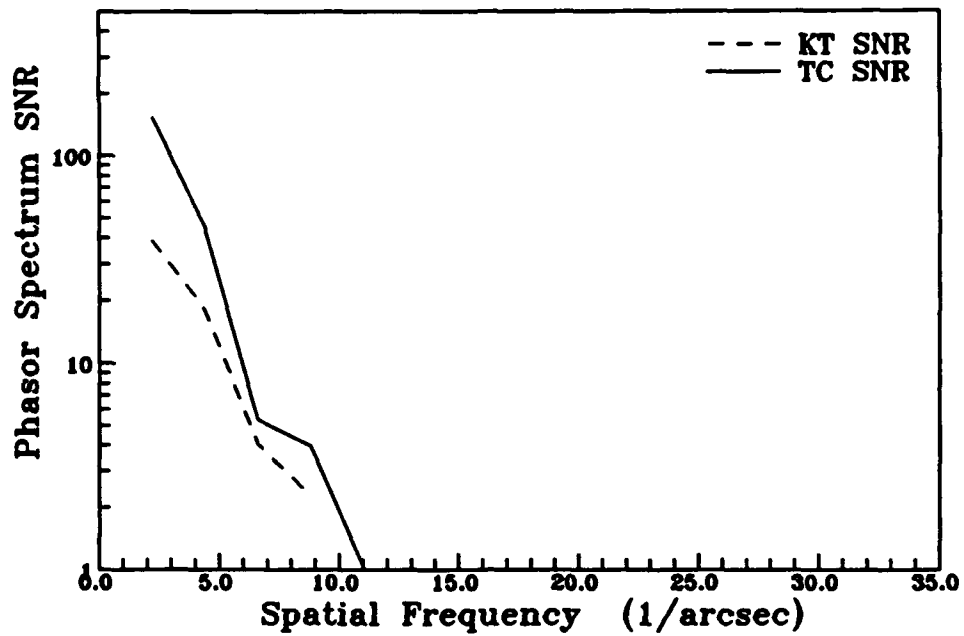


Figure A.72 Asteroid Phasor Spectrum SNR,  
Tilt,  $N_p = 10^3$ .

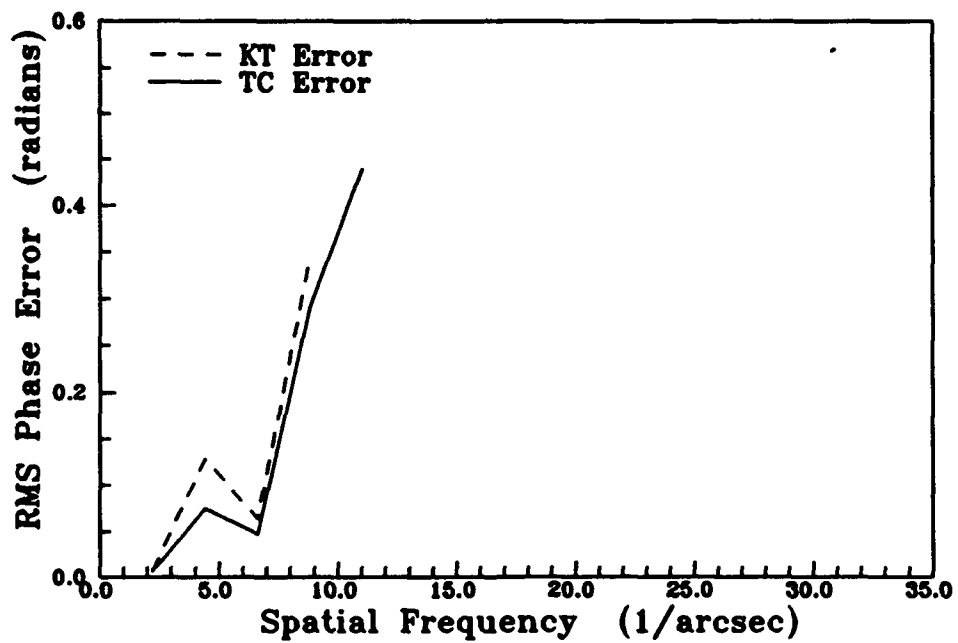
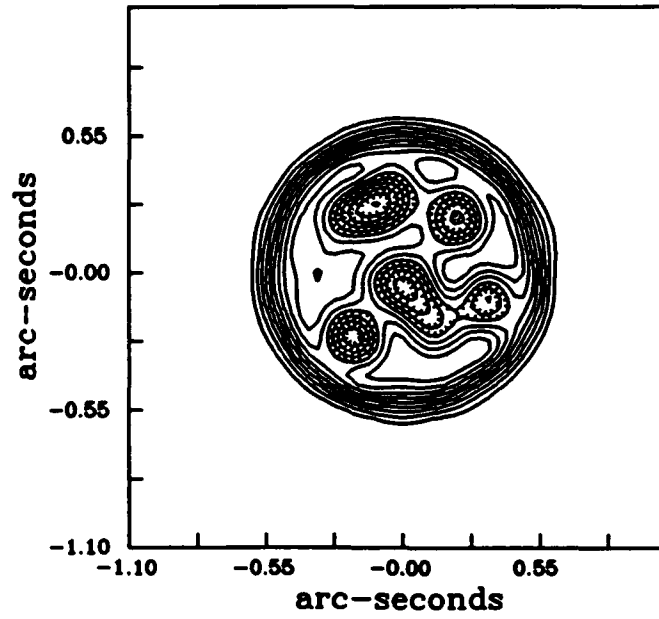
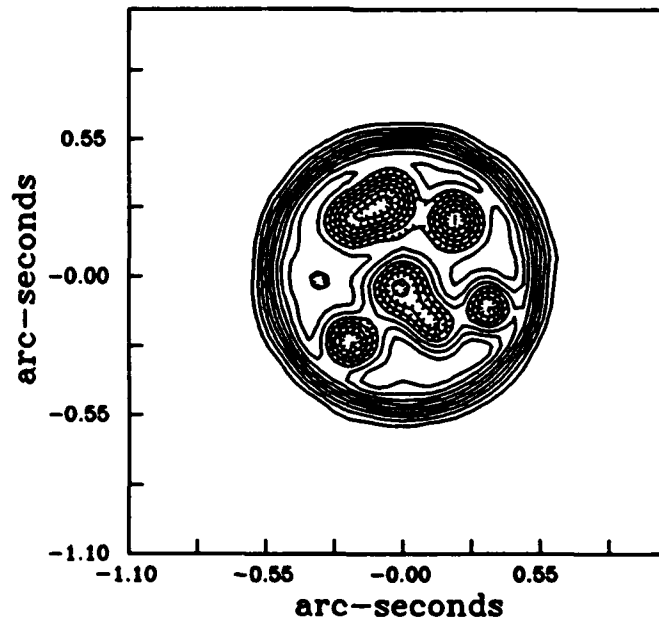


Figure A.73 Asteroid Azimuthal RMS Phase Error,  
Tilt,  $N_p = 10^3$ .



**Figure A.74** KT Recovered Asteroid,  
With Tilt,  $r_0 = 0.206$ ,  $N_p = 10^5$ ,  
 $N_f = 1600$ , OS = 4.



**Figure A.75** TC Recovered Asteroid,  
With Tilt,  $r_0 = 0.206$ ,  $N_p = 10^5$ ,  
 $N_f = 1600$ , OS = 4.

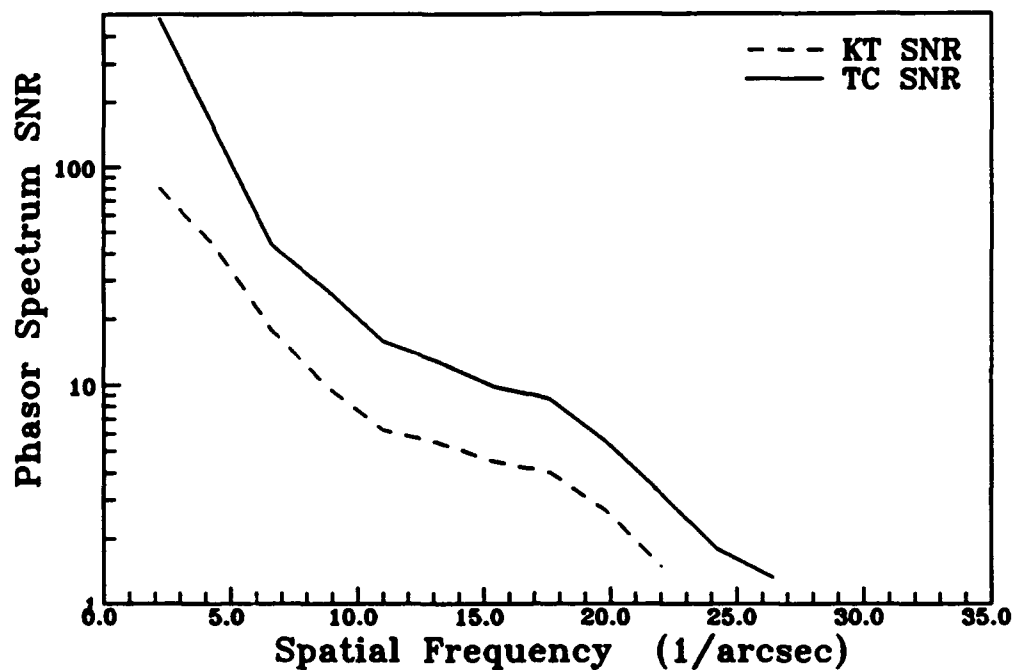


Figure A.76 Asteroid Phasor Spectrum SNR, Tilt,  $N_f = 1600$ .

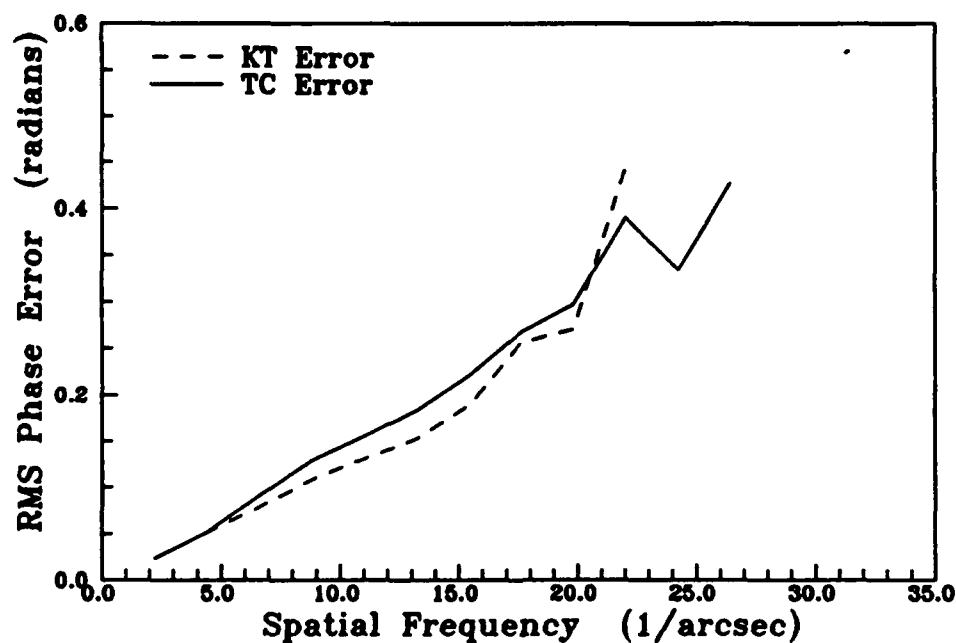
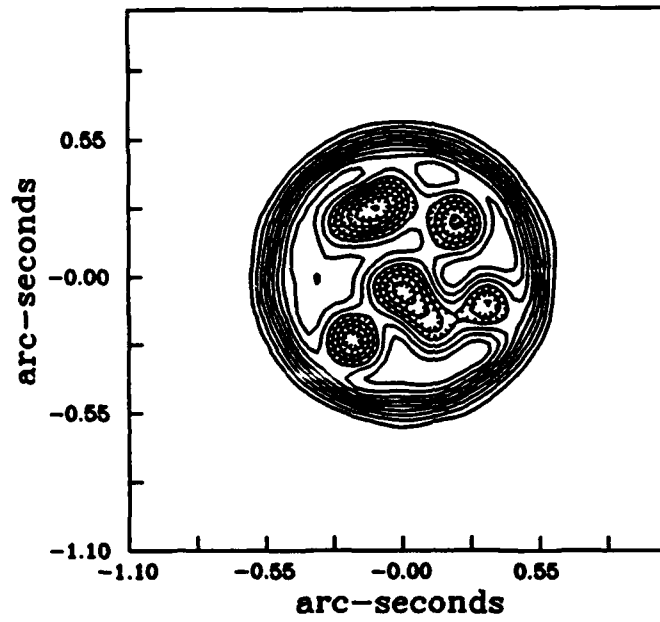
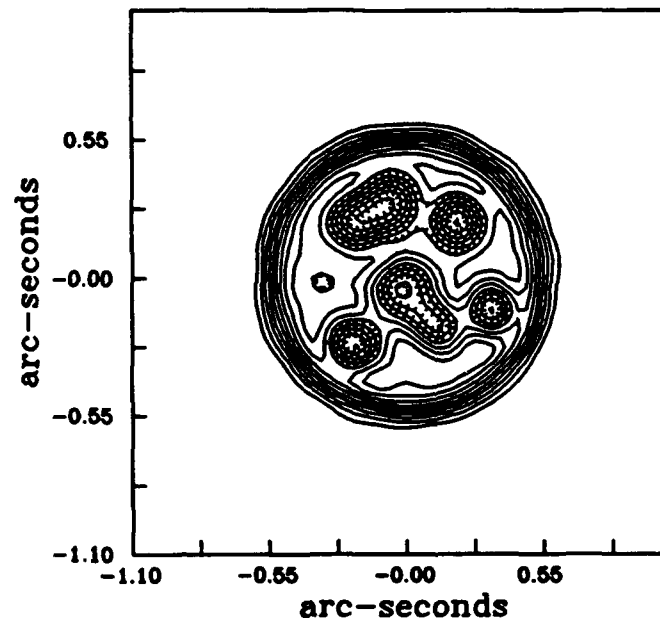


Figure A.77 Asteroid Azimuthal RMS Phase Error, Tilt,  $N_f = 1600$ .



**Figure A.78** KT Recovered Asteroid,  
With Tilt, Centroided,  $r_0 = 0.206$ ,  
 $N_p = 10^5$ ,  $N_t = 1600$ , OS = 4.



**Figure A.79** TC Recovered Asteroid,  
With Tilt, Centroided,  $r_0 = 0.206$ ,  
 $N_p = 10^5$ ,  $N_t = 1600$ , OS = 4.

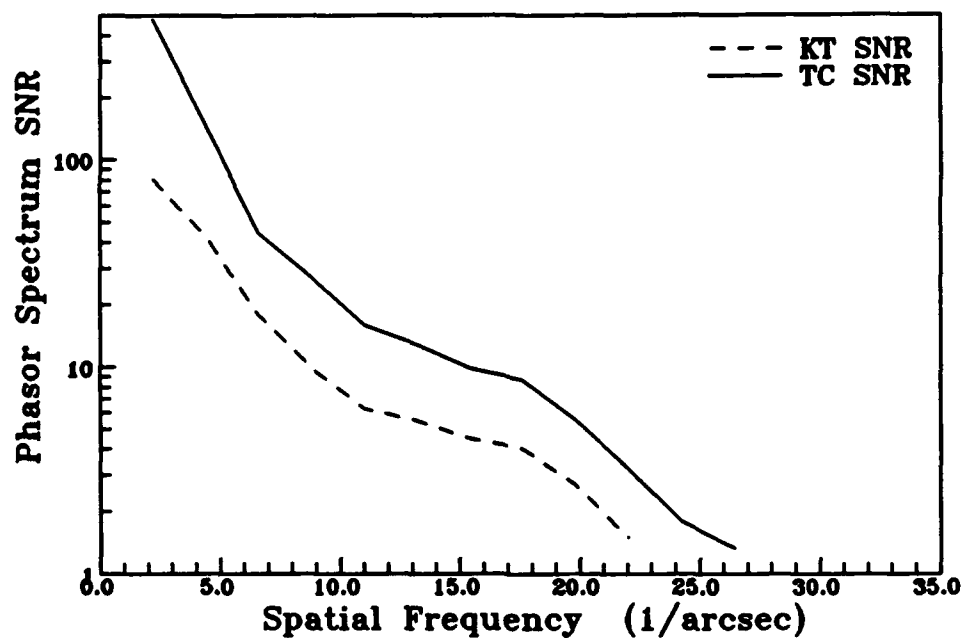


Figure A.80 Asteroid Phasor Spectrum SNR, Tilt, Centroiding,  $N_t = 1600$ .

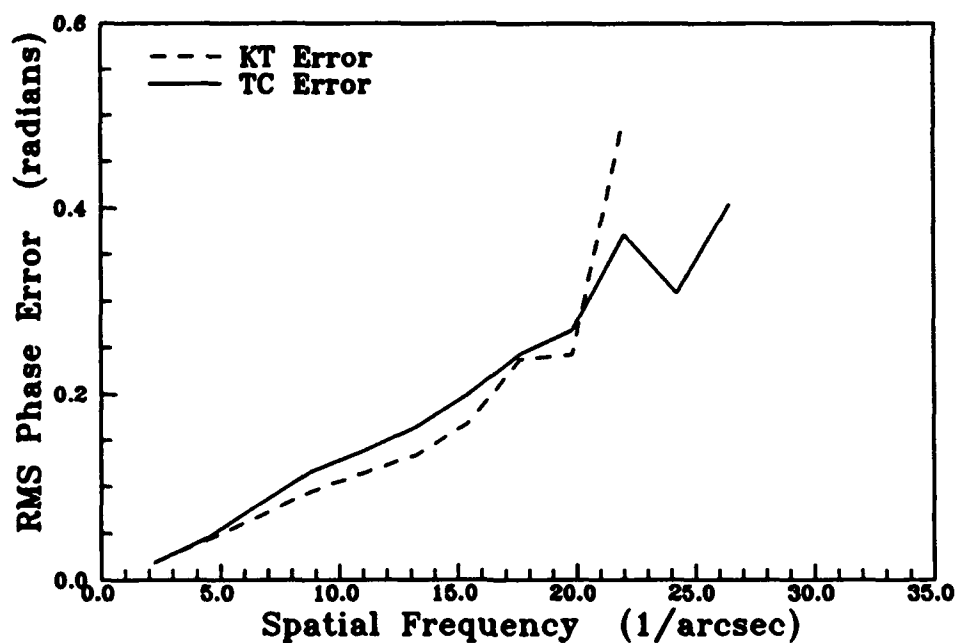
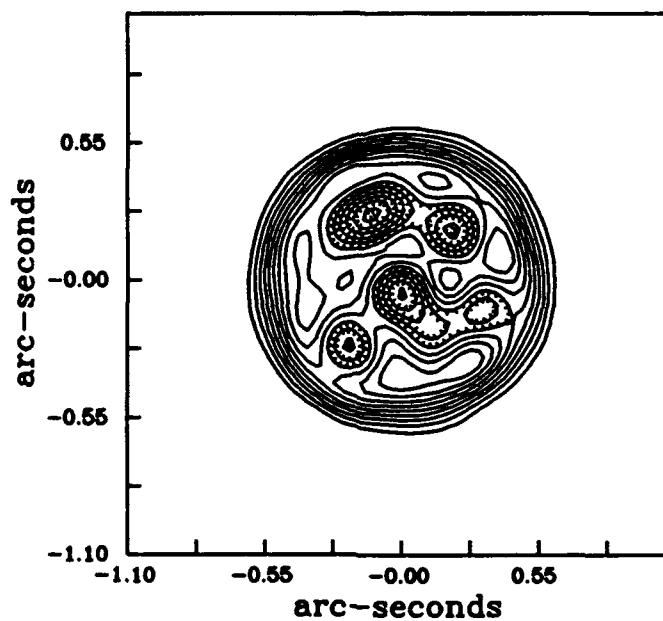
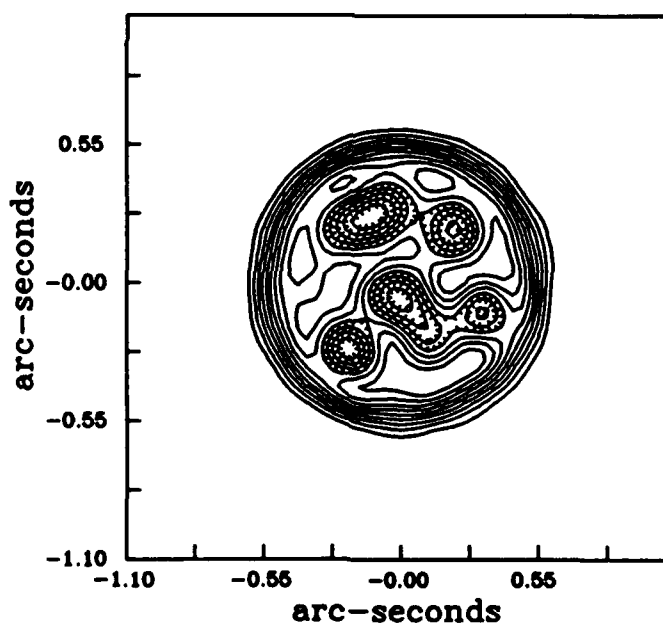


Figure A.81 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding,  $N_t = 1600$ .



**Figure A.82** KT Recovered Asteroid,  
With Tilt, Centroided,  $r_0 = 0.206$ ,  
 $N_p = 10^5$ ,  $N_t = 400$ , OS = 4.



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

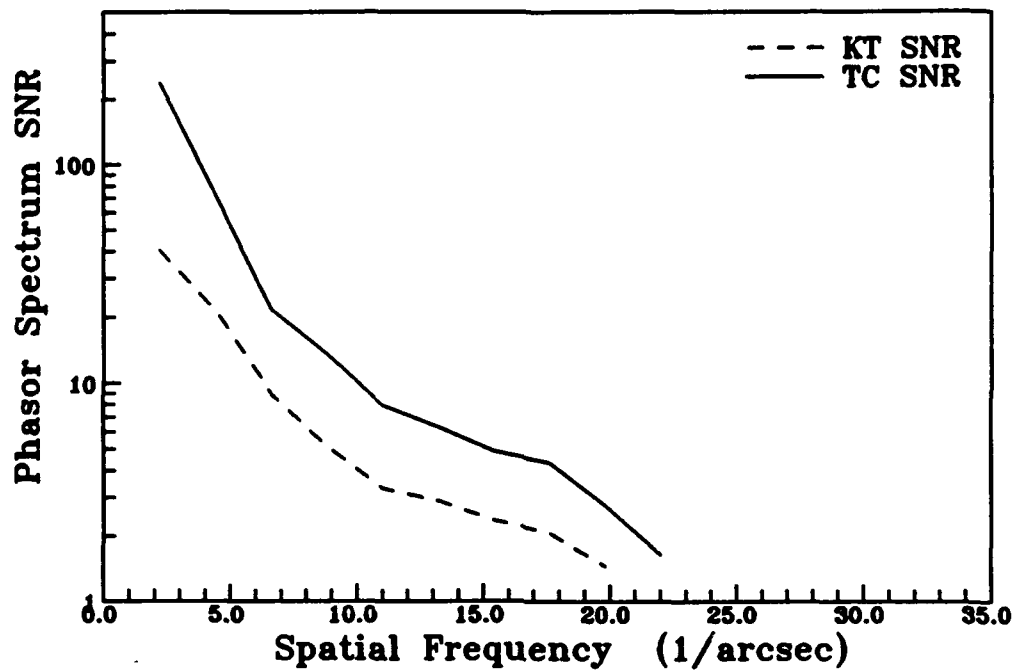


Figure A.84 Asteroid Phasor Spectrum SNR, Tilt, Centroiding.

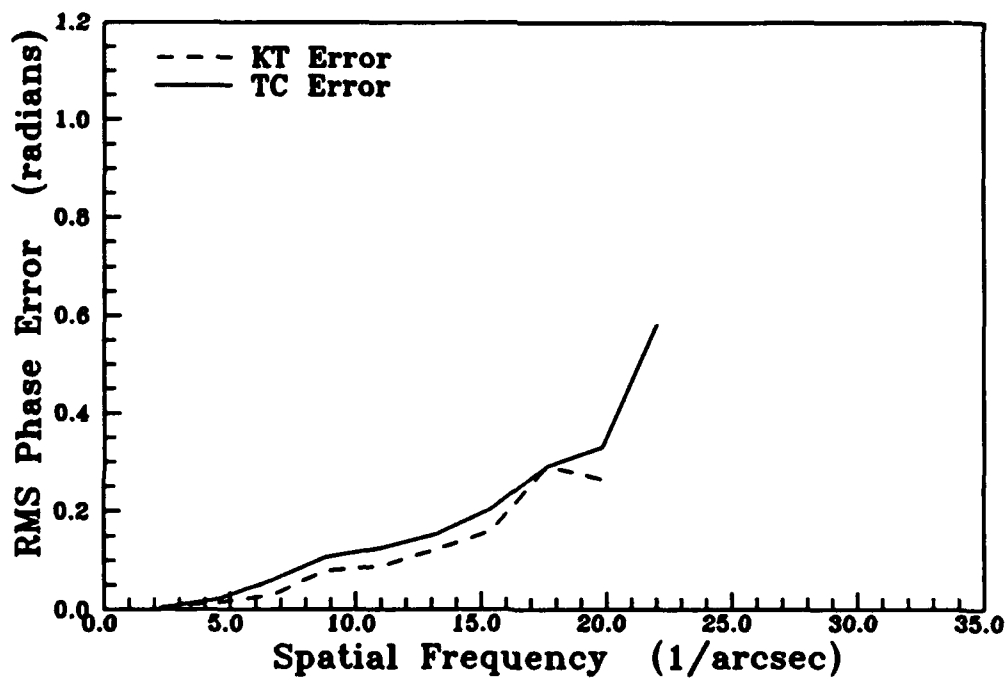
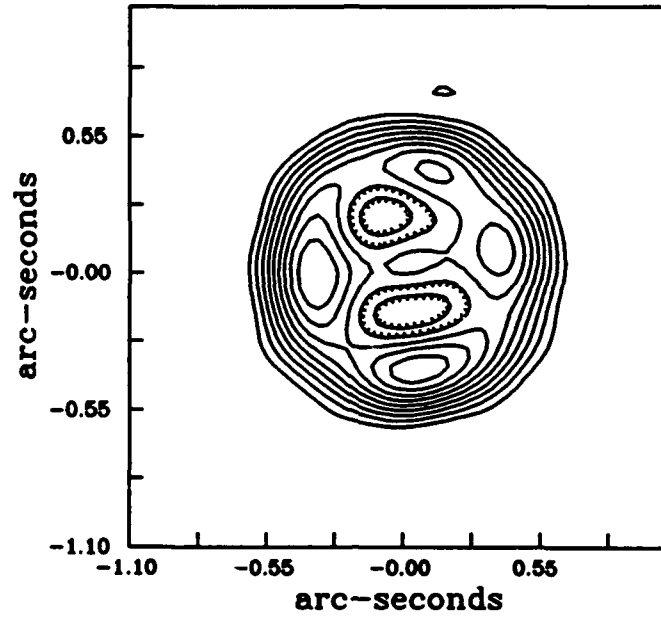
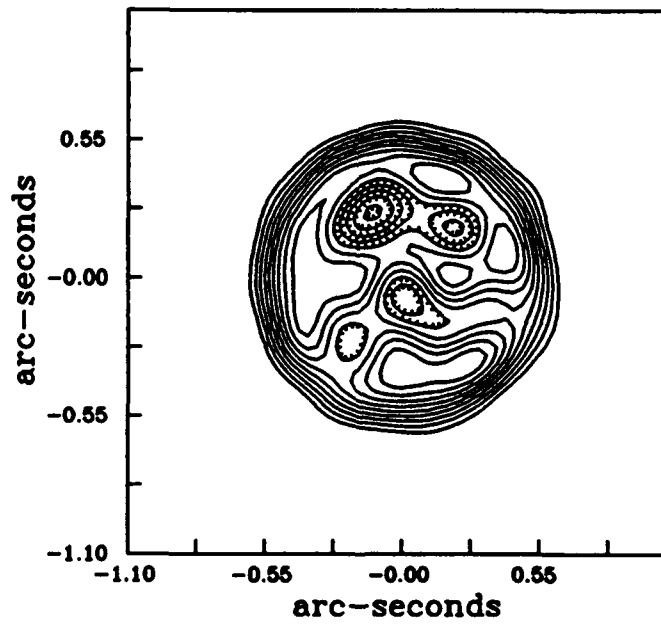


Figure A.85 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding.



**Figure A.86** KT Recovered Asteroid, With Tilt,  
Centroided,  $r_0 = 0.206$ ,  $N_p = 10^4$ ,  
 $N_f = 400$ , OS = 4.



**Figure A.87** TC Recovered Asteroid, With Tilt,  
Centroided,  $r_0 = 0.206$ ,  $N_p = 10^4$ ,  
 $N_f = 400$ , OS = 4.



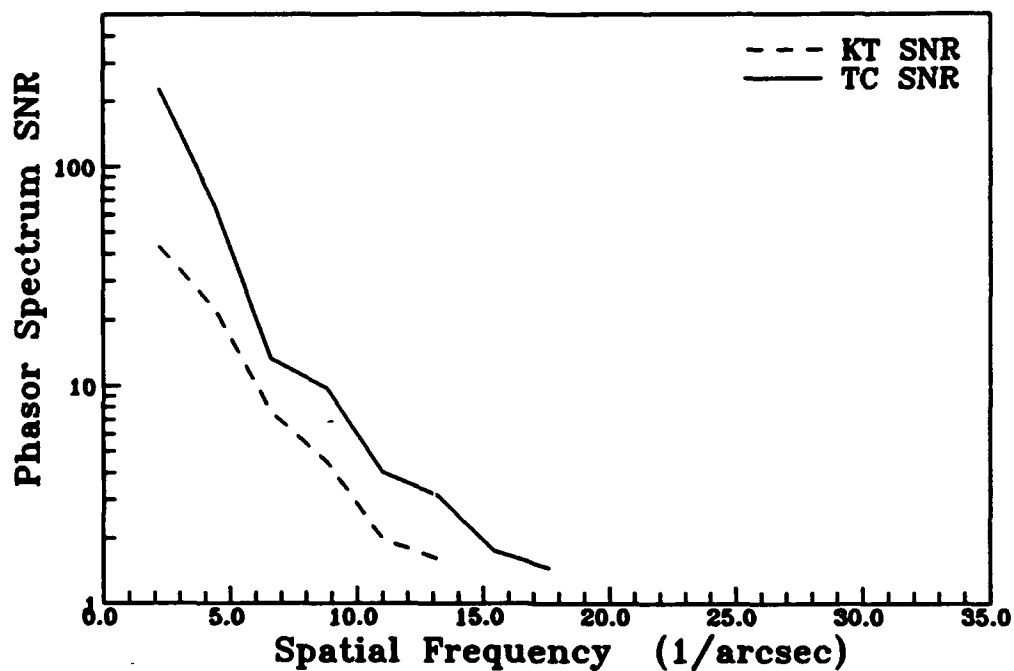


Figure A.88 Asteroid Phasor Spectrum SNR, Tilt, Centroiding,  $N_p = 10^4$ .

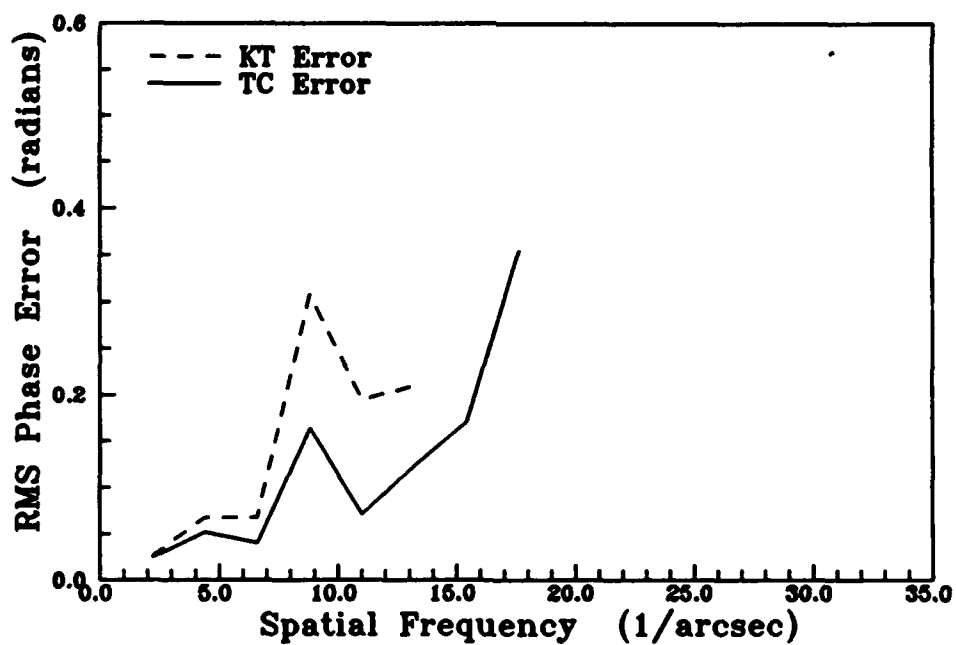
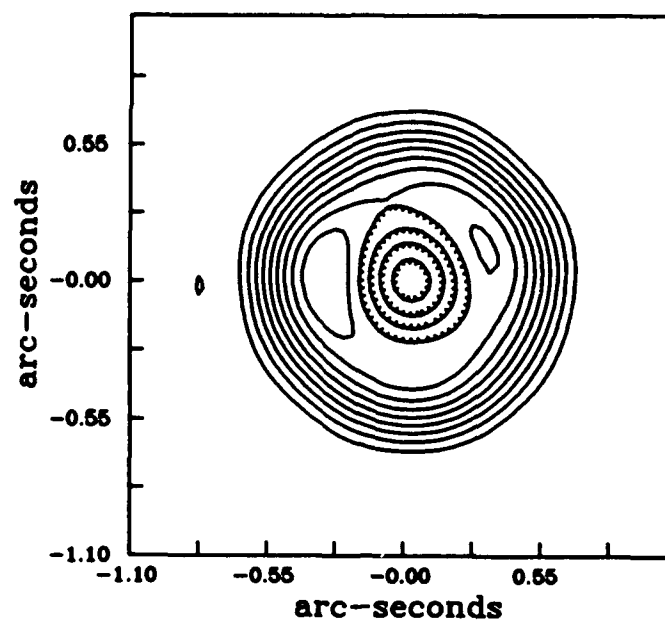
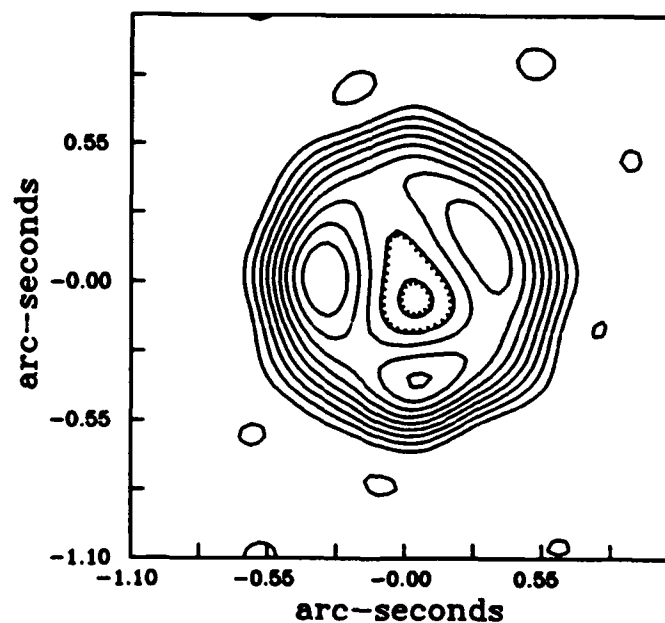


Figure A.89 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding,  $N_p = 10^4$ .



**Figure A.90** KT Recovered Asteroid, With Tilt,  
Centroided,  $r_0 = 0.206$ ,  $N_p = 10^3$ ,  
 $N_t = 400$ , OS = 4.



**Figure A.91** TC Recovered Asteroid, With Tilt,  
Centroided,  $r_0 = 0.206$ ,  $N_p = 10^3$ ,  
 $N_t = 400$ , OS = 4.

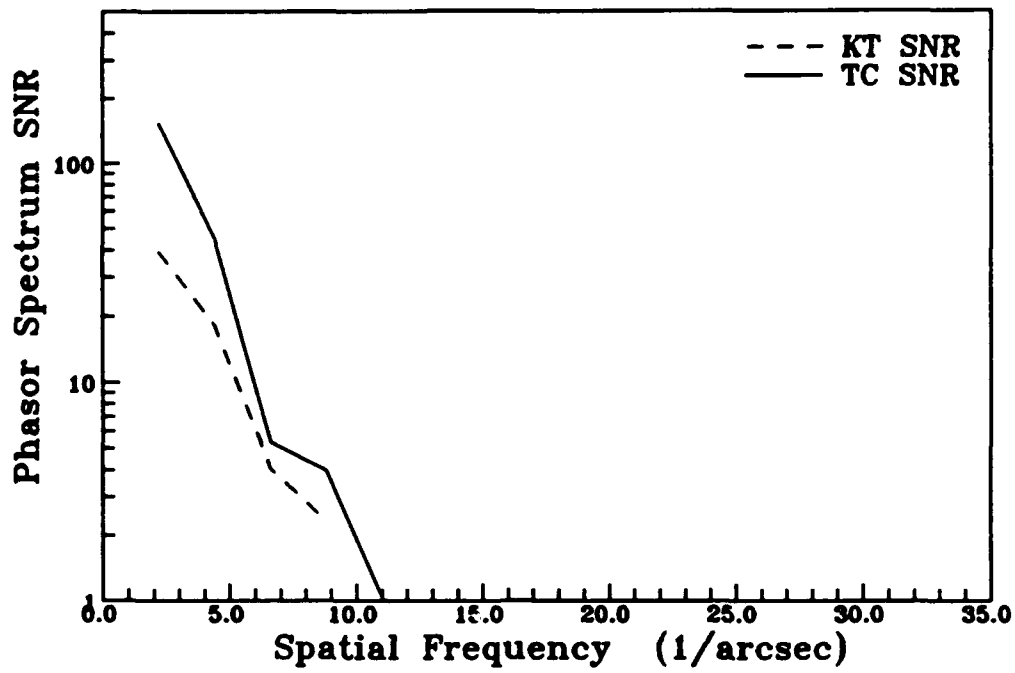


Figure A.92 Asteroid Phasor Spectrum SNR, Tilt, Centroiding,  $N_p = 10^3$ .

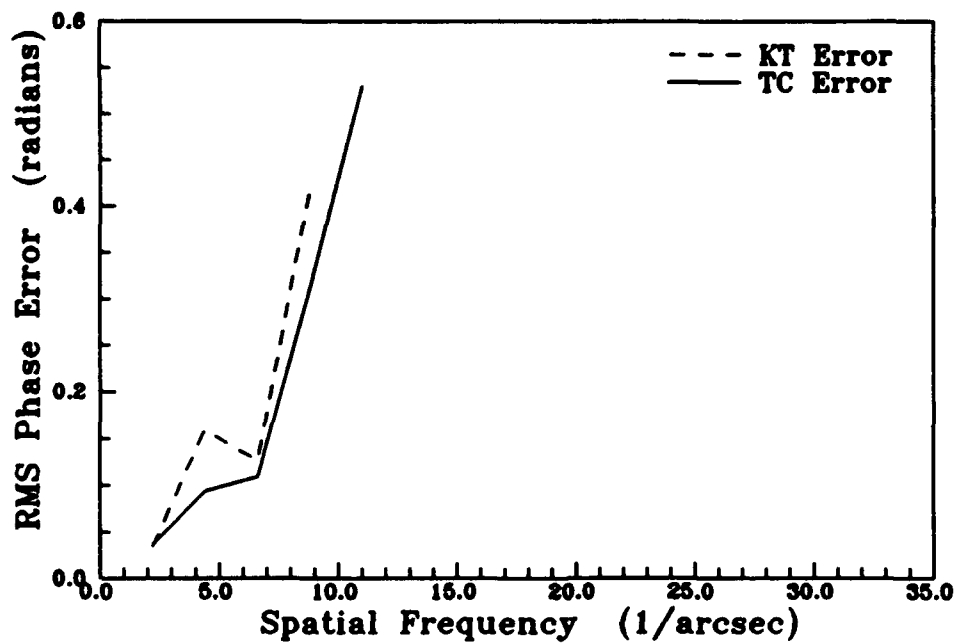
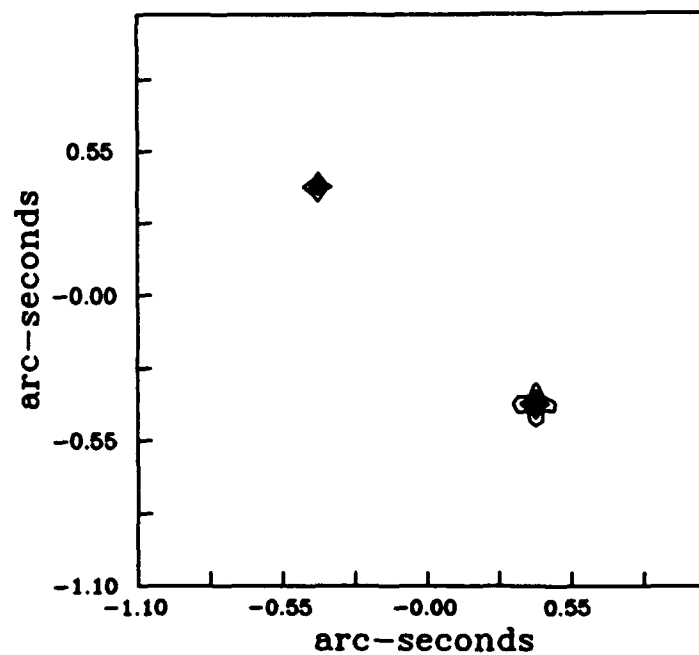
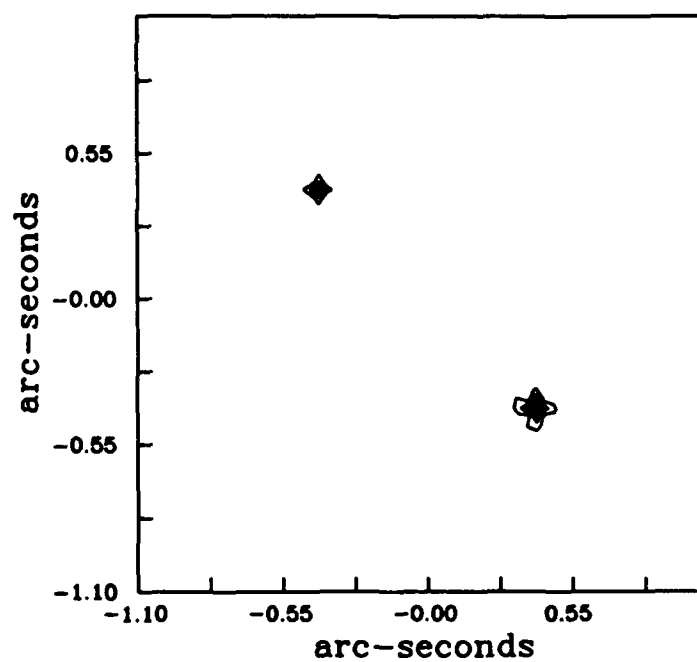


Figure A.93 Asteroid Azimuthal RMS Phase Error, Tilt, Centroiding,  $N_p = 10^3$ .



**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.

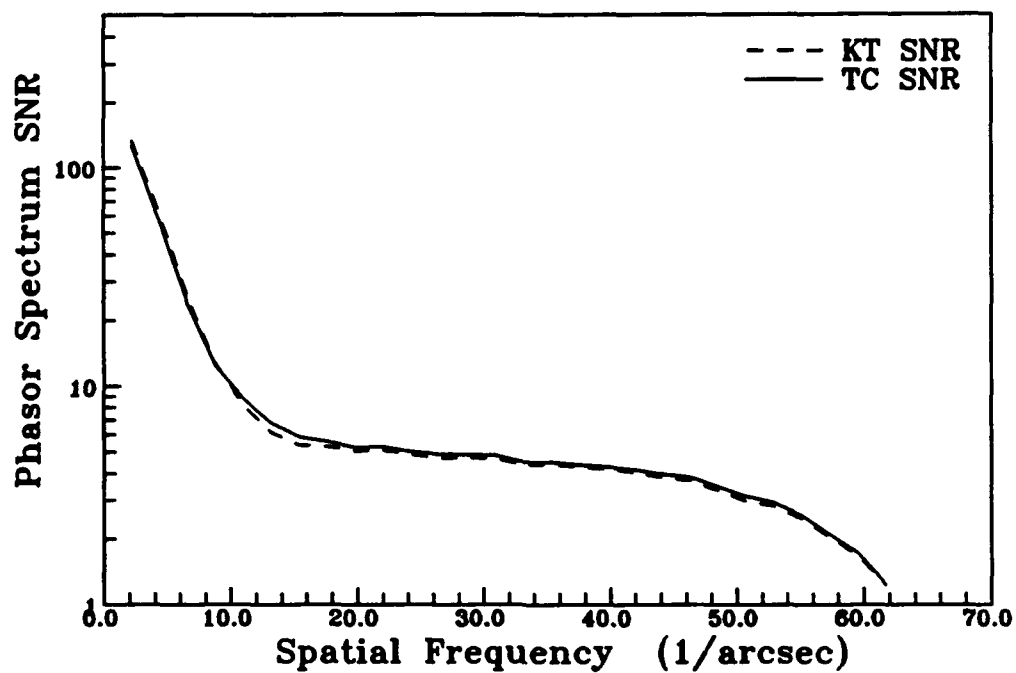


Figure A.96 Binary Star Phasor Spectrum SNR,  $N_p = 10^3$ .

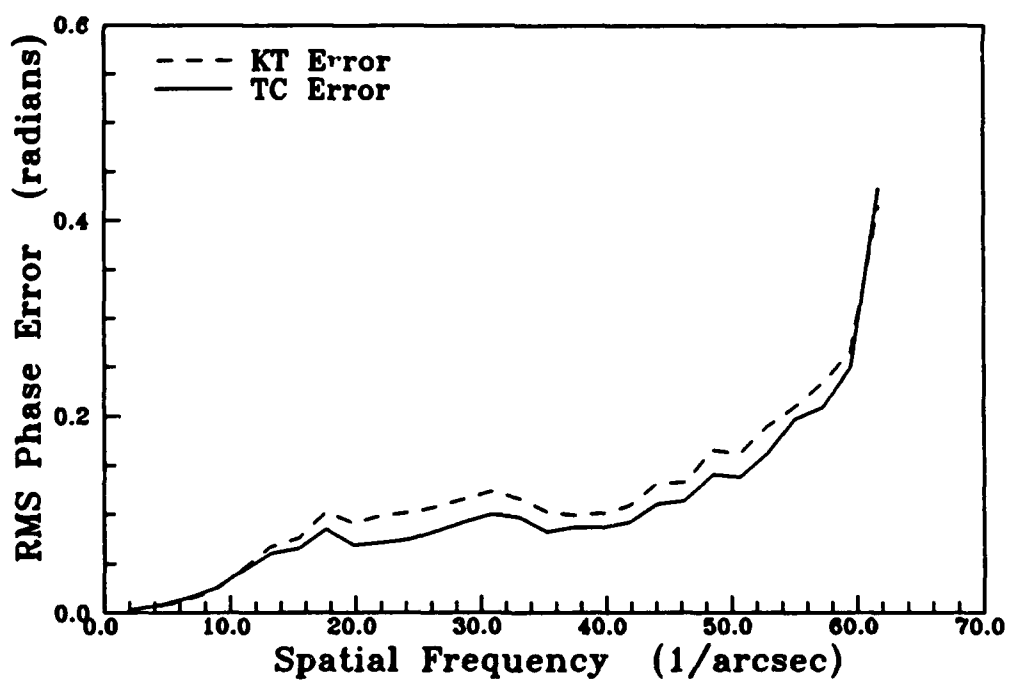
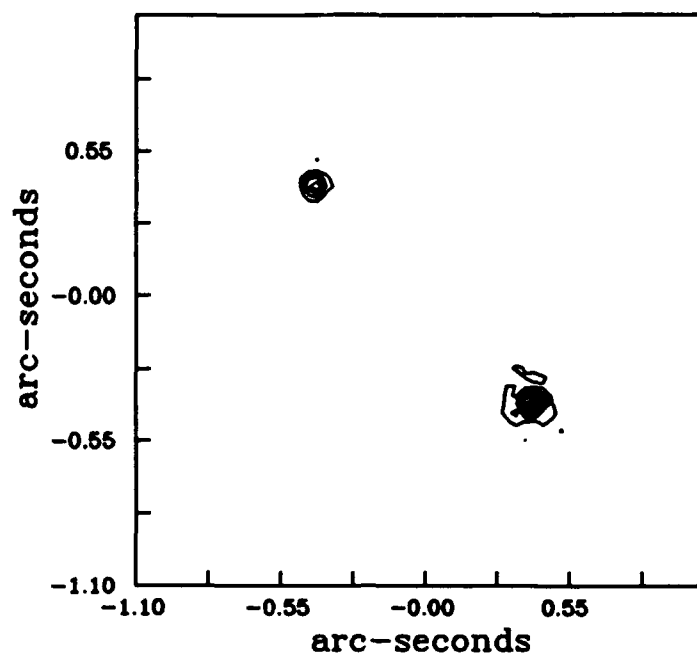
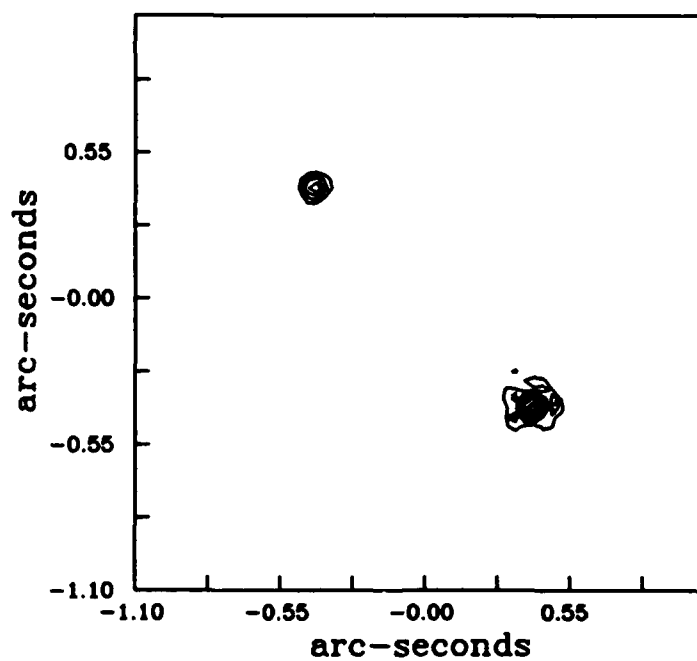


Figure A.97 Binary Star Azimuthal RMS Phase Error,  $N_p = 10^3$ .



**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.

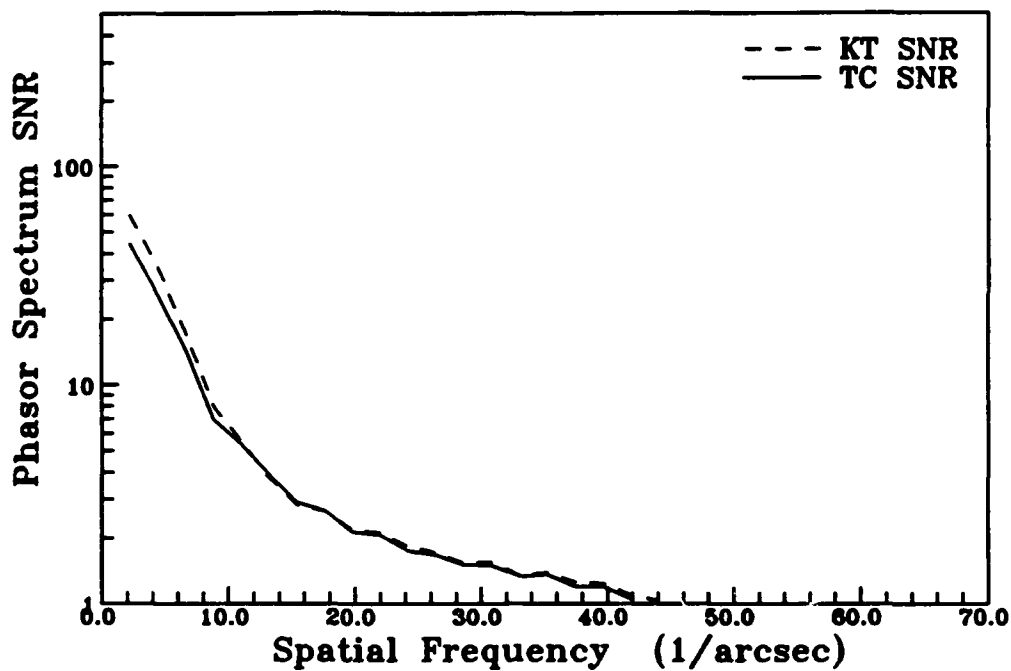


Figure A.100 Binary Star Phasor Spectrum SNR,  $N_p = 10^2$ .

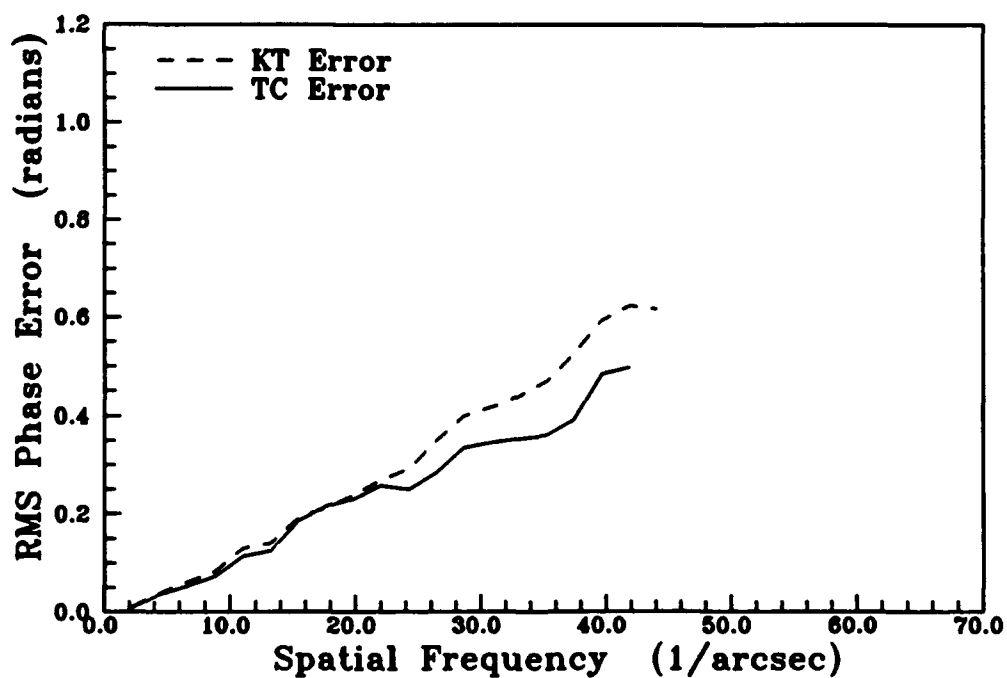
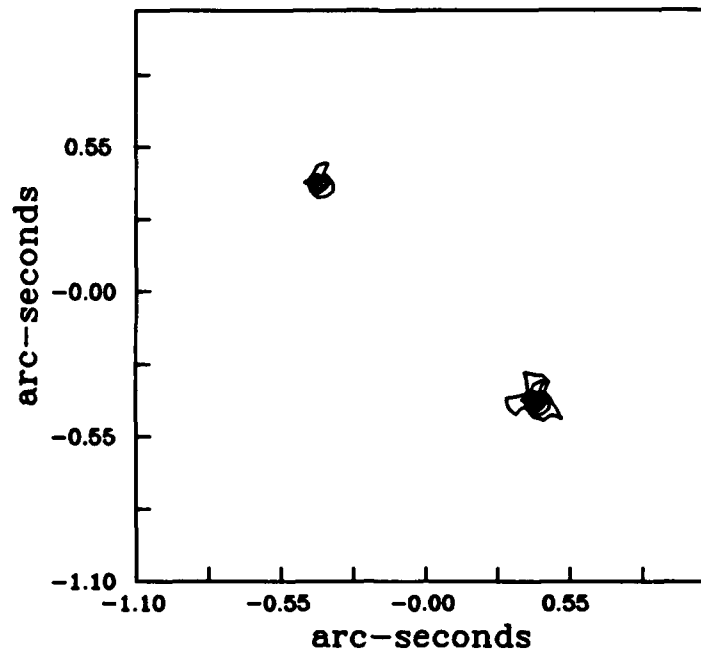
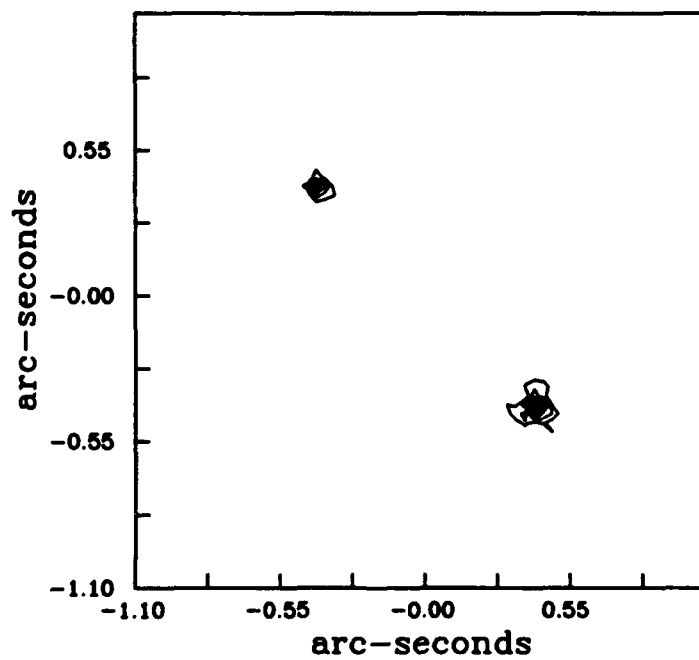


Figure A.101 Binary Star Azimuthal RMS Phase Error,  $N_p = 10^2$ .



**Figure A.102** KT Recovered Binary Star, No Tilt,  
 $r_0 = 0.103$ ,  $N_p = 10_3$ ,  $N_f = 400$ , OS = 4.



**Figure A.103** TC Recovered Binary Star, No Tilt,  
 $r_0 = 0.103$ ,  $N_p = 10_3$ ,  $N_f = 400$ , OS = 4.



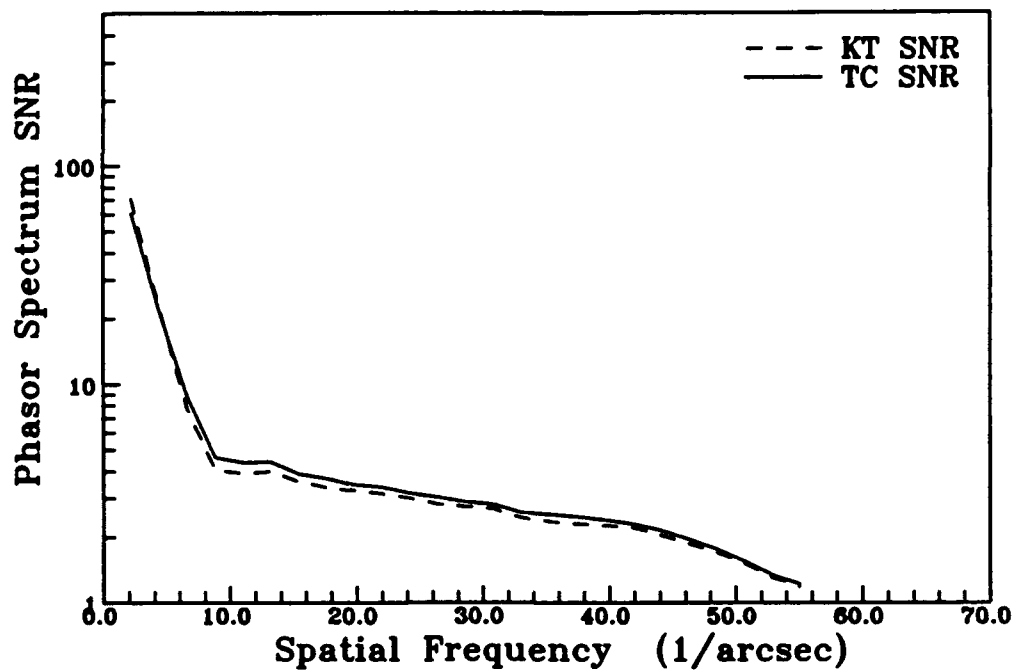


Figure A.104 Binary Star Phasor Spectrum SNR,  $r_0 = 0.103$ ,  $N_p = 10^3$ .

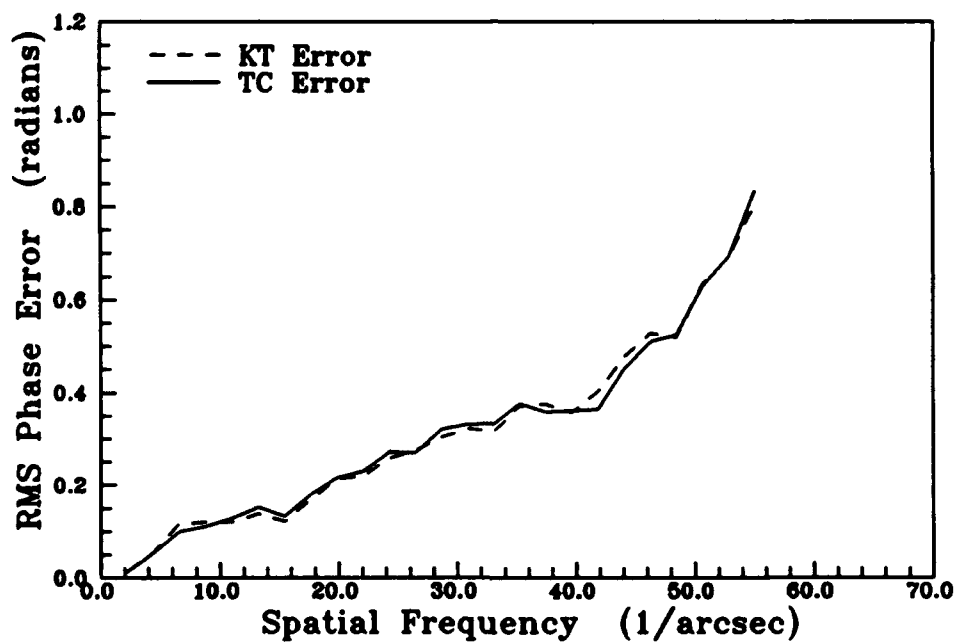
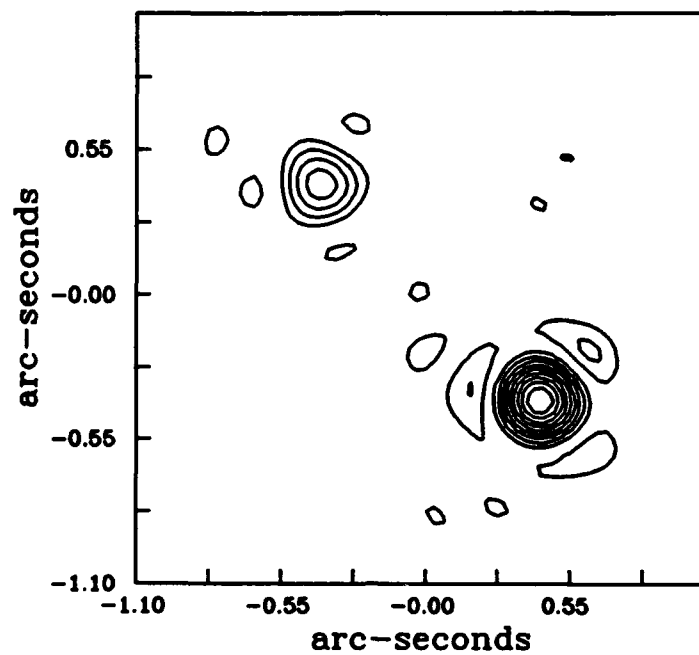
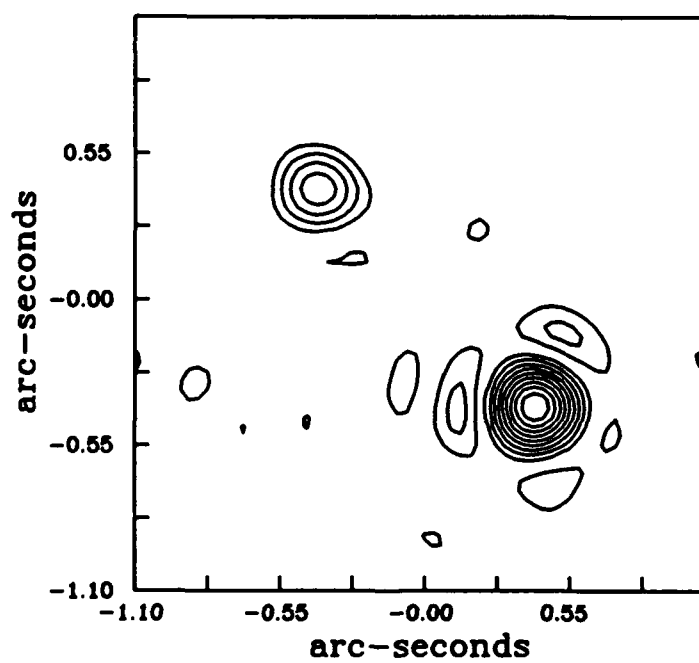


Figure A.105 Binary Star Azimuthal RMS Phase Error,  $r_0 = 0.103$ ,  $N_p = 10^3$ .



**Figure A.106** KT Recovered Binary Star, No Tilt,  
 $r_0 = 0.103$ ,  $N_p = 10^2$ ,  $N_t = 400$ , OS = 4.



**Figure A.107** TC Recovered Binary Star, No Tilt,  
 $r_0 = 0.103$ ,  $N_p = 10^2$ ,  $N_t = 400$ , OS = 4.

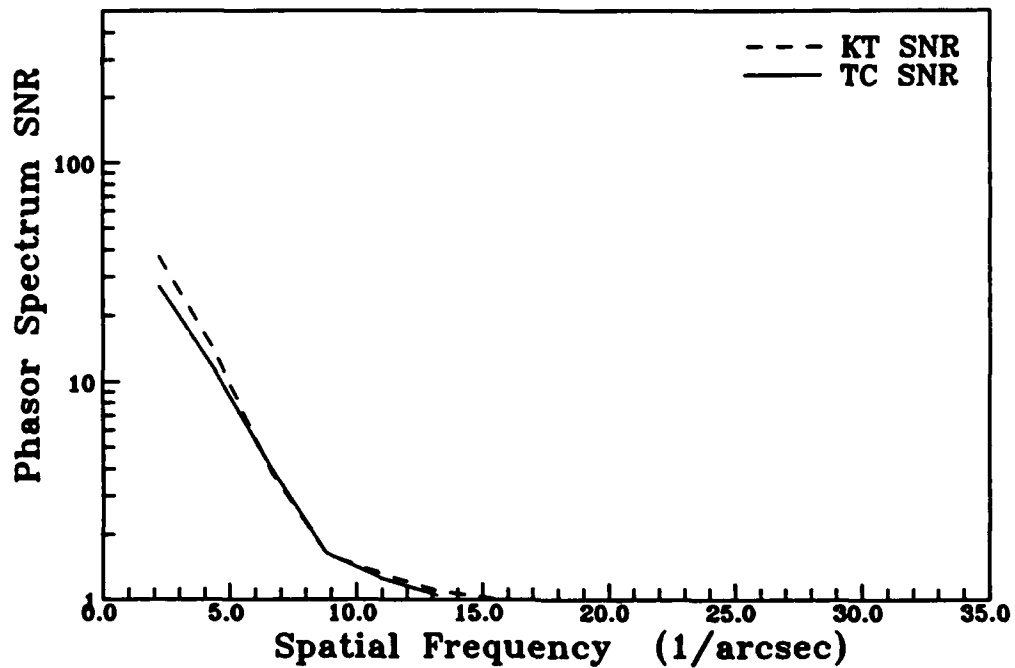


Figure A.108 Binary Star Phasor Spectrum SNR,  $r_0 = 0.103$ ,  $N_p = 10^2$ .

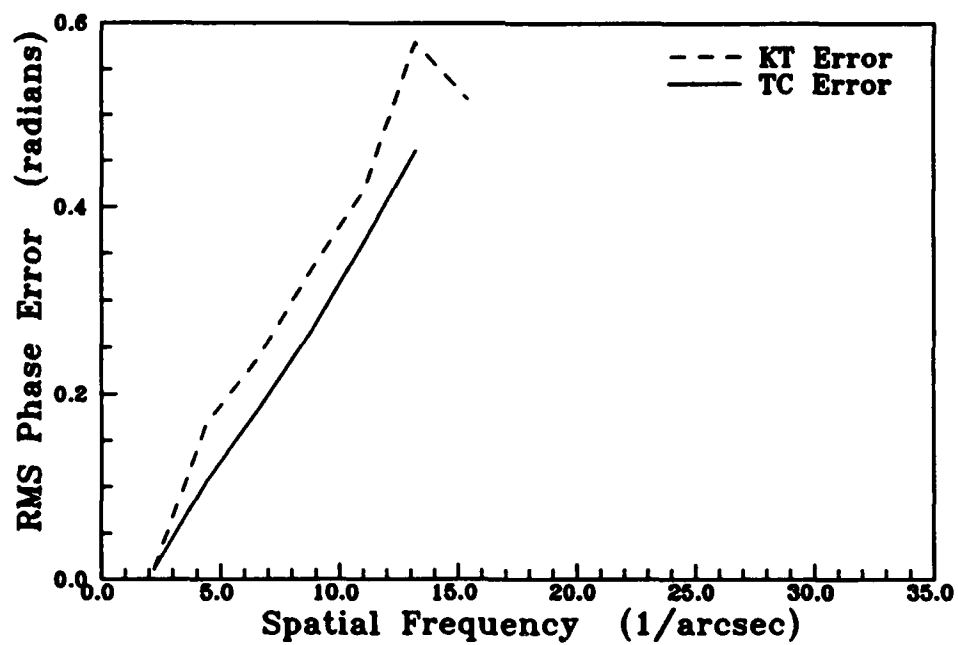


Figure A.109 Binary Star Azimuthal RMS Phase Error,  $r_0 = 0.103$ ,  $N_p = 10^2$ .

## APPENDIX B. KNOX-THOMPSON MAIN PROGRAM

c THIS PROGRAM CREATES ONE OF THREE IMAGES: A STAR, A  
c BINARY STAR AND AN ASTEROID. IT THEN DEGRADES THE  
c IMAGE BY SIMULATING ATMOSPHERIC CONDITIONS AND PHOTON  
c NOISE, AND THEN RECONSTRUCTS THE IMAGE USING THE  
c KNOX-THOMPSON ALGORITHM.

c AUTHOR: LT JAMES M. LACKEMACHER  
c COMPL. DATE: 26 OCTOBER 1990  
c REASON: COMPLETE REQUIREMENTS FOR A MASTERS  
c DEGREE IN PHYSICS.  
c GOAL: SIMULATE OBJECT, DEGRADE OBJECT,  
c RECONSTRUCT OBJECT USING KNOX-  
c THOMPSON AND TRIPLE-CORRELATION  
c METHODS, FILTER AND COMPARE.

### PROGRAM KNOXTHOMPSON

#### c MAIN PROGRAM COMPLEX VARIABLE LIST

c F n DIM ARRAY USED IN THE FOURIER TRANSFORM  
c GAUSSIAN n x n DIM ARRAY THAT REPRESENTS THE GAUSSIAN  
c PORTION OF THE ASTEROID  
c I n x n DIM ARRAY THAT REPRESENTS THE  
c DEGRADED IMAGE  
c IKT n x n x 5 x 9 ARRAY THAT REPRESENTS THE CROSS-  
c SPECTRUM OF THE IMAGE  
c IS n x n DIM ARRAY THAT REPRESENTS THE  
c DEGRADED POINT SOURCE  
c ISKT n x n DIM ARRAY THAT REPRESENTS THE MODULUS  
c SQUARED OF THE POINT SOURCE

```

c   OKT          n x n DIM ARRAY THAT REPRESENTS THE OBJECT
c               SPECTRUM OF THE RECONSTRUCTED IMAGE
c   OBJDATA      n x n DIM ARRAY THAT REPRESENTS THE OBJECT
c   KTPHASOR     n x n DIM ARRAY THAT REPRESENTS THE PHASOR OF
c               THE OBJECT IN THE RECONSTRUCTION PROCESS
c   PUPIL        n x n DIM ARRAY THAT REPRESENTS THE PUPIL
c               PORTION OF THE ASTEROID
c   TEMPDATA     n x n DIM ARRAY THAT IS USED AS A TEMPORARY
c               ARRAY

```

#### c                   MAIN PROGRAM REAL VARIABLE LIST

```

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

```

#### c                   MAIN PROGRAM INTEGER VARIABLE LIST

```

c   fwd          VALUE OF 1 FOR FORWARD FFT
c   icounter     COUNTER THAT COUNTS THE NUMBER OF
c               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
c               PIXELS THAT ARE AVERAGED IN THE KNOX-
c               THOMPSON PROCESS
c   mseed        VARIABLE USED TO ENSURE ONLY ONE PASS OF
c               INITIAL PART OF PHSUB SUBROUTINE

```

```

c      n          DIMENSION OF ONE SIDE OF 2-DIM ARRAY
c      nframes    TOTAL NUMBER OF SHORT EXPOSURE SNAPSHOTS
c      nphoton     TOTAL NUMBER OF PHOTONS IN SNAPSHOT
c      nyquist     EQUAL TO THE TELESCOPE PUPIL FUNCTION RADIUS
c                  DERIVED FROM THE RELATION:
c                  nyquist = (telescope diameter x
c                             number of pixels per r0)/r0

```

```

c                  MAIN PROGRAM

```

```

      PARAMETER(n=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), xvarrkt(n,n,5,9), xvarikt(n,n,5,9),
+ KTsnr(n,n), rsnr(n/2)

```

```

      INTEGER offset

```

```

      CHARACTER*16 file1, file2
      CHARACTER*1 cent

```

```

c      INITIALIZE MSED TO ALLOW ONLY ONE PASS THROUGH FIRST
c      PART OF PHSUB

```

```

      mseed = 1

```

```

c      INITIALIZE PROGRAM READING REQUIRED VARIABLES AND
c      CREATE THE DESIRED OBJECT

```

```

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

```

```

c      TRANSFORM THE OBJECT TO FREQUENCY SPACE

```

```

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

```

```

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

CALL Photons(OBJDATA,nphoton,n)

c  CREATE THE POINT SOURCE

CALL Star(STARDATA,n)

c  TRANSFORM THE POINT SOURCE TO FREQUENCY SPACE

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

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

CALL Photons(STARDATA,nphoton,n)

c  COMMENCE THE LOOP THAT COUNTS THE SNAPSHOTS

DO 10 icounter = 1, nframes

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

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

c  CENTROID THE DEGRADED IMAGE ONLY SINCE ONLY PHASE IS
c  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
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,I,IKT,KTsnnr,offset,xvarrkt,
+       xvarikt,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(KTsnnr,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

c THIS PROGRAM CREATES ONE OF THREE IMAGES: A STAR, A  
c BINARY STAR AND AN ASTEROID. IT THEN DEGRADES THE  
c IMAGE BY SIMULATING ATMOSPHERIC CONDITIONS AND PHOTON  
c NOISE, AND THEN RECONSTRUCTS THE IMAGE USING THE TRIPLE-  
c CORRELATION ALGORITHM.

c AUTHOR: LT JAMES M. LACKEMACHER  
c COMPL. DATE: 26 OCTOBER 1990  
c REASON: COMPLETE REQUIREMENTS FOR A MASTERS  
c DEGREE IN PHYSICS.  
c GOAL: SIMULATE OBJECT, DEGRADE OBJECT,  
c RECONSTRUCT OBJECT USING KNOX-  
c THOMPSON AND TRIPLE-CORRELATION  
c METHODS, FILTER AND COMPARE.

### PROGRAM TRIPLECORR

#### c MAIN PROGRAM COMPLEX VARIABLE LIST

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

c	IS	$n \times n$ DIM ARRAY THAT REPRESENTS THE DEGRADED
c		POINT SOURCE
c	ISBS	$n \times n$ DIM ARRAY THAT REPRESENTS THE MODULUS
c		SQUARED OF THE POINT SOURCE
c	OBS	$n \times n$ DIM ARRAY THAT REPRESENTS THE OBJECT
c		SPECTRUM OF THE RECONSTRUCTED IMAGE
c	OBJDATA	$n \times n$ DIM ARRAY THAT REPRESENTS THE OBJECT
c	PUPIL	$n \times n$ DIM ARRAY THAT REPRESENTS THE PUPIL
c		PORTION OF THE ASTEROID
c	TEMPDATA	$n \times n$ DIM ARRAY THAT IS USED AS A TEMPORARY
c		ARRAY IN THE FOURIER TRANSFORM

c                   MAIN PROGRAM REAL VARIABLE LIST

c	BSsnr	$n \times n$ DIM ARRAY THAT REPRESENTS THE SNR OF
c		EACH PHASOR
c	mod	$n \times n$ DIM ARRAY THAT REPRESENTS THE
c		MODULUS OF THE RECONSTRUCTED IMAGE
c	rsnr	$n/2$ DIM ARRAY THAT REPRESENTS THE SNR AS A
c		FUNCTION OF RADIUS
c	xvarrbs	$n \times n \times 5 \times 9$ DIM ARRAY THAT REPRESENTS THE
c		REAL PART OF THE VARIANCE OF THE BISPECTRUM
c	xvaribs	$n \times n \times 5 \times 9$ DIM ARRAY THAT REPRESENTS THE
c		IMAGINARY PART OF THE VARIANCE OF THE
c		BISPECTRUM

c                   MAIN PROGRAM INTEGER VARIABLE LIST

c	fwd	VALUE OF 1 FOR FORWARD FFT
c	icounter	COUNTER THAT COUNTS THE NUMBER OF
c		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 AVERAGED IN THE TRIPLE-
		CORRELATION PROCESS

```

c   mseed      VARIABLE USED TO ENSURE ONLY ONE PASS OF
c               INITIAL PART OF PHSUB SUBROUTINE
c   n          DIMENSION OF ONE SIDE OF 2-DIM ARRAY
c   nframes    TOTAL NUMBER OF SHORT EXPOSURE SNAPSHOTS
c   nphoton    TOTAL NUMBER OF PHOTONS IN SNAPSHOT
c   nyquist    EQUAL TO THE TELESCOPE PUPIL FUNCTION RADIUS
c               DERIVED FROM THE RELATION:
c               nyquist = (telescope diameter x
c                           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), xvarrrbs(n,n,5,9), xvaribs(n,n,5,9),
+ BSsnr(n,n), rsnr(n/2)

```

```

INTEGER offset

```

```

CHARACTER*16 file1, file2
CHARACTER*1 cent

```

```

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

```

```

mseed = 1

```

```

c   INITIALIZE PROGRAM READING REQUIRED VARIABLES AND
c   CREATE THE DESIRED OBJECT

```

```

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

```

```

c   TRANSFORM THE OBJECT TO FREQUENCY SPACE

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

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

    CALL Photons(OBJDATA,nphoton,n)

c   CREATE THE POINT SOURCE

    CALL Star(STARDATA,n)

c   TRANSFORM THE POINT SOURCE TO FREQUENCY SPACE

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

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

    CALL Photons(STARDATA,nphoton,n)

c   COMMENCE THE LOOP THAT COUNTS THE SNAPSHOTS

    DO 10 icounter = 1, nframes

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

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

c   CENTROID THE DEGRADED IMAGE ONLY SINCE ONLY PHASE IS
c   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
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,IBS,ISBS,mod,icounter,
+             nframes,nphoton,n)

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

      CALL BSrecon(BSPHASOR,I,IBS,IDBS,BSsnr,offset,
+             xvarrbs,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(BSsnr,rsnr,nyquist,n)

c   COMBINE THE MODULUS WITH THE PHASOR

      CALL Combine(OBS,mod,BSPHASOR,n)

```

c WRITE RECONSTRUCTED PHASE AND POWER SPECTRUM TO A  
c FILE

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

STOP  
END

## APPENDIX D. UNIVERSAL IMAGE RECOVERY SUBROUTINES

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

### SUBROUTINE LIST

```
SUBROUTINE Ast(ASTEROID,GAUSSIAN,PUPIL,F,TEMPDATA,
+             fwd,inv,ln,n)
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
10  CONTINUE
CALL FFT2D(GAUSSIAN,TEMPDATA,F,ln,fwd,n)
CALL FFT2D(PUPIL,TEMPDATA,F,ln,fwd,n)
```

```

DO 20 i = 1, n
  DO 20 j = 1, n
    ASTEROID(i,j) = GAUSSIAN(i,j) * PUPIL(i,j)
20  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 FFTing THE  
c ROWS THEN THE COLUMNS OF THE 2-D ARRAY. THIS S/R WAS  
c ACQUIRED FROM "DIGITAL IMAGE PROCESSING" BY GONZALEZ  
c 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 l = 1, ln  
le = 2\*\*l  
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)
20  CONTINUE
  CALL FFT(F,ln,n)
  DO 30 j = 1, n
    DATA(i,j) = F(j)
30  CONTINUE
10  CONTINUE
  DO 40 j = 1, n
    DO 50 i = 1, n
      F(i) = DATA(i,j)
50  CONTINUE
  CALL FFT(F,ln,n)
  do 60 i = 1, n
    DATA(i,j) = F(i)
60  CONTINUE
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,IS,IDATA,ISDATA,mod,icounter,
+                 nframes,nphoton,n)

c   THIS S/R DETERMINES THE MODULUS OF THE DEGRADED IMAGE
c   AND THE POINT SOURCE AND NORMALIZES THEM BY DIVIDING
c   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**
- c 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**
- c 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**
- c 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)
```



```

    TEMPDATA(i+n2,j+n2) = DATA(i,j)
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(j - (n2p1))
            y = float(i - (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)
```

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

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

### SUBROUTINE LIST

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

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

```

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
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 KT OUTPUT FILE NAME (16 CHAR'
    WRITE(*,*)'MAX):'
    READ(*,30) file1
    WRITE(*,*) '
    WRITE(*,*)'ENTER KT SNR FILE NAME (16 CHAR'
    WRITE(*,*)'MAX):'
    READ(*,30) file2
30  FORMAT(A16)
40  FORMAT(A1)
    RETURN
    END

```

```

SUBROUTINE KTrecon(KTPHASOR,I,IKT,KTsnr,offset,
+                 xvarrkt,xvarikt,icounter,
+                 nframes,nyquist,n)

```

```

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

```

```

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,
+ xvarrkt(-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
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
      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))
+         I(idi,jdj) = (1.0,0.0)
        IF (I(ii,jj).EQ.(0.0,0.0))
+         I(ii,jj) = (1.0,0.0)
        CTEMP = (I(idi,jdj) * DCONJG(I(ii,jj))) -
+         DCONJG(I(di,dj))
        IKT(idi,jdj,di,dj) = IKT(idi,jdj,di,dj) +
+         CTEMP
        xvarrkt(idi,jdj,di,dj) =
+         xvarrkt(idi,jdj,di,dj) +
+         (DREAL(CTEMP))**2.0
        xvarikt(idi,jdj,di,dj) =
+         xvarikt(idi,jdj,di,dj) +
+         (DIMAG(CTEMP))**2.0
        IF (icounter.EQ.nframes) THEN
          nsnr = nsnr + 1
          sigmar = dabs((xvarrkt(idi,jdj,di,dj) -
+            ((DREAL(IKT(idi,jdj,di,dj)))**2.0)/
+            dfloat(nframes))/dfloat(nframes - 1))
          sigmai = dabs((xvarikt(idi,jdj,di,dj) -
+            ((DIMAG(IKT(idi,jdj,di,dj)))**2.0)/
+            dfloat(nframes))/dfloat(nframes - 1))
          sigma = dsqrt(sigmar + sigmai)
          snr = (((ABS(IKT(idi,jdj,di,dj)))/
+            dfloat(nframes))/sigma) *
+            dsqrt(dfloat(nframes))
          KTsnr(ii,jj) = KTsnr(ii,jj) + snr
          PTEMP1 = IKT(idi,jdj,di,dj)/
+            ABS(IKT(idi,jdj,di,dj))
          PTEMP2 = KTPHASOR(idi,jdj)
          KTPHASOR(ii,jj) = KTPHASOR(ii,jj) +
+            ((snr**2.0) * (DCONJG(PTEMP1/PTEMP2)))
        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

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

### SUBROUTINE LIST

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

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

COMPLEX\*16 BSPHASOR(-n/2:n/2-1,-n/2:n/2-1), CTEMP,  
+ PTEMP2, IBS(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4),  
+ PTEMP1, I(-n/2:n/2-1,-n/2:n/2-1), IDBS(0:4,-4:4)  
REAL\*8 BSsnr(-n/2:n/2-1,-n/2:n/2-1), sigmar, sigmai,  
+ snr, xvarrbs(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4),  
+ sigma, xvaribs(-n/2:n/2-1,-n/2:n/2-1,0:4,-4:4)  
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

```

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))
+             I(idi,jdj) = (1.0,0.0)
            IF (I(ii,jj).EQ.(0.0,0.0))
+             I(ii,jj) = (1.0,0.0)
            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(di,dj)))**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

```

```

      IF (icounter.EQ.nframes) THEN
        nsnr = nsnr + 1
        sigmar = dabs((xvarrbs(idi,jdj,di,dj) -
+      ((DREAL(ABS(idi,jdj,di,dj)))**2.0)/
+      dfloat(nframes))/dfloat(nframes - 1))
        sigmai = dabs((xvaribs(idi,jdj,di,dj) -
+      ((DIMAG(ABS(idi,jdj,di,dj)))**2.0)/
+      dfloat(nframes))/dfloat(nframes - 1))
        sigma = dsqrt(sigmar + sigmai)
        snr = (((ABS(ABS(idi,jdj,di,dj)))/
+      dfloat(nframes))/sigma) *
+      dsqrt(dfloat(nframes))
        BSsnr(ii,jj) = BSsnr(ii,jj) + snr
        PTEMP1 = ABS(ABS(idi,jdj,di,dj))/
+      ABS(ABS(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
c  QUEARYING THE USER FOR INPUT. IT ALSO CREATES THE
c  OBJECT DESIRED BY THE USER BY CALLING EITHER ASTEROID,
c  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

```

```

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'
    WRITE(*,*)'(CHAR MAX):'
    READ(*,30) file1
    WRITE(*,*)' '
    WRITE(*,*)'ENTER TC SNR FILE NAME (16 CHAR'
    WRITE(*,*)'MAX):'
    READ(*,30) file2
30  FORMAT(A16)
40  FORMAT(A1)
    RETURN
    END

```

## APPENDIX G. ATMOSPHERIC DEGRADATION SUBROUTINES

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

### SUBROUTINE LIST

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

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

c ORDER OF EVENTS TO OBTAIN IMAGE:

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

c DICTIONARY:

- c DIAM - Telescope pupil diameter
- c ICOUNTER - Number of user chosen iterations of  
c phase screens to process
- c ISEED - Seed to be used for random number  
c generator
- c LAMBDA - Center wavelength
- c MU - Mean

c PIXSCALE - Corresponds to each pixel in cycles/m  
 c R0 - Variable characterizing the strength of atmospheric turbulence  
 c 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 (IWDIM = 5\*IDIM/2,IWDIM2 = 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,R0,INNERSCALE,  
 + CORR(LDIM2),SECDIM,OUTERSCALE,OBSCUR,RARRAY(LDIM)

INTEGER PCOUNT

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

COMMON /VARS/ZERO,RMAX,RMIN,CMINUS1,RCONST1,FCONST1,PI  
 COMMON /VARS2/DIAM,OBSCUR,LAMBDA,R0,SECDIM,  
 + PIXSCALE,TPFDIM,FILENAME

c 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).

```

11 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
- c array to be stored in array CORR. However, only call the filter function on
- c multiple runs if there are changing values for the width of the object and R0.

```

CALIFILT2(RARRAY,CORR,LDIM2,IDIM2,MULT1,MULT2,PIXSCALE,
+       ZERO,R0)

```

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

```

SIZE = (IDIM/2)-1
IF(SIZE.LT.TPFDIM) THEN
  WRITE(*,*) '
  IF(SIZE.LE.(.5*TPFDIM)) THEN
    WRITE(*,*)'ERROR. IMAGE ARRAY SIZE MUST BE
+       512x512.'

```



```

        ELSE
        WRITE(*,*)'ERROR. IMAGE ARRAY SIZE MUST BE
+      256x256.'
        ENDIF
        WRITE(*,*) '
        GOTO 400
        ENDIF

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

        CALL BGSCR2(BVAL,ZKLMAP,LINPTR,IDIM2)

c    End the if statement if first time through the subroutine.

        mseed = mseed + 1

    ENDIF

c    Get Gaussian numbers and then multiply these two numbers by the correlation
c    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*CORN(J),R2*CORN(J))
20    CONTINUE

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

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

        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,PI,PIXSCALE,TILT)

```

- c Calculate  $ae^{i \cdot \text{realization of phase screen}}$ . 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

```

- c Perform the Telescope Pupil Function (TPF) on the phase map to obtain the
- c coherent transfer function. First, call this routine to cut out chunk of phase
- c screen based on Telescope Pupil Function of  $D/\text{LAMBDA}$  to 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

```

- c Incoherent transfer function is completed. Now get the image in the fourier
- c domain by multiplying the object and the phase screen to get the image.

```

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

```

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

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

```
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

```

- c Error Statments

```

500 CONTINUE
  WRITE(*,*)' '
  WRITE(*,*)'ERROR, THIS FILE DOES NOT EXIST. REENTER.'
  WRITE(*,*)' '
  GOTO 11
800 CONTINUE
  END

```

## SUBROUTINE BGSCR2(bval,zklmap,linptr,nsid)

- c 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 = 'klrad.d'
```

- c Dictionary

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

```

10  format(6E4.9)
    open(lu4,file=noname,form='formatted',status='old')
    rewind (lu4)
    do 80 i=1,nr
        do 85 j=1,n4
            read(lu4,*) rmap(i,j)
85      continue
80      continue
    read(lu4,*) ir
    call kelp(zklmap,rmap,ir,ival,linptr,n3,n4,
+          nsid,nr,nkl)
    return
    end

```

SUBROUTINE FADD(bmap,zklmap,iq,nsid,nkl,sum)

c    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)

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

```

    REAL RARRAY(LDIM2),CORR(LDIM2),MULT1,MULT2,RCONST,
+    PIXSCALE,ZERO,R0,FCONST,R,ARRAYW
    INTEGER JOFFSET

```

```

c   Calculate the radial average of an input image.

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

```

- c Multiply the scaling array by the correlation filter function

```
RCONST = (-11.E0/12.E0)
DO 60 I = 1,LDIM2
  RARRAY(I) = (RARRAY(I)+(MULT2*MULT2))*RCONST
60 CONTINUE
ARRAYW = PIXSCALE*IDIM2
RCONST = (R0/ARRAYW)**(-5.E0/6.E0)
DO 70 I = 1,LDIM2
  RARRAY(I) = (RARRAY(I)*RCONST)*FCONST
70 CONTINUE
```

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

```
RARRAY(IDIM2*(IDIM2/2)+(IDIM2/2+1)) = ZERO
```

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

```
RCONST = -2.0*(MULT1*MULT1)
DO 90 I= 1,LDIM2
  CORR(I) = (EXP(CORR(I)/RCONST))*RARRAY(I)
90 CONTINUE

900 CONTINUE
RETURN
END
```

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

- c PURPOSE:
- c GET GAUSSIAN DISTRIBUTED RANDOM NUMBER WITH MEAN MU
- c AND STANDARD DEVIATION SIGMA

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

```
TWOPI = 2.e0*PI
```

```
Y1= uniform(iseed)
Y2= uniform(iseed)
```

c Calculate first random number

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

c Calculate the second random number

RNUM2 = MU + SIGMA\*SQRT(-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/
icent = 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)
    enddo
  enddo
enddo

```

```

        zklmap(i,j,iq) = zz*top
        endif
20    continue
10    continue
15    continue
return
end

```

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

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

```

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

WRITE(*,*) '
WRITE(*,*) 'THE DEFAULT VALUES FOR THIS PROGRAM ARE AS
+     FOLLOWS: '
WRITE(*,*) '
WRITE(*,*)     TELESCOPE DIAMETER:         ',DIAM
WRITE(*,*)     CENTER WAVELENGTH:         ',LAMBDA
WRITE(*,*)     R0:                         ',R0
WRITE(*,*)     TELESCOPE SECONDARY DIAMETER:',OBSCUR
20  CONTINUE
WRITE(*,*) '
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(*,*) '
    WRITE(*,*) 'YOUR ANSWER HAS TO BE EITHER Y OR N.
+     REENTER'
    FLAG = ''
    GOTO 20
ENDIF

```

```

30  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.'L').AND.(ANSWER.NE.'l')).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(*,*)' '
        WRITE(*,*)'THE CURRENT VALUES ARE: '
        WRITE(*,*)' '
        WRITE(*,*)'          TELESCOPE DIAMETER: ',DIAM
        WRITE(*,*)'          CENTER WAVELENGTH: ',LAMBDA
        WRITE(*,*)'          R0: ',R0
        WRITE(*,*)'          SECONDARY DIAMETER ',OBSCUR
        WRITE(*,*)' '
        WRITE(*,*)' '
        IF((ANSWER.EQ.'A').OR.(ANSWER.EQ.'a'))THEN
            WRITE(*,*)'INPUT THE FOLLOWING'
            WRITE(*,*)' '
            WRITE(*,*)'ENTER NEW TELESCOPE DIAMETER:'
            READ(*,100) DIAM
            WRITE(*,*)' '
            WRITE(*,*)'ENTER NEW CENTER WAVELENGTH:'
            READ(*,100) LAMBDA
            WRITE(*,*)' '
            WRITE(*,*)'ENTER NEW R0:'
            READ(*,100) R0
            WRITE(*,*)' '
            WRITE(*,*)'ENTER NEW TELESCOPE SECONDARY DIAM:'

```

```

    READ(*,100) OBSCUR
    ELSEIF((ANSWER.EQ.'D').OR.(ANSWER.EQ.'d'))THEN
        WRITE(*,*)' '
        WRITE(*,*)'ENTER NEW TELESCOPE DIAMETER:'
        READ(*,100)DIAM
    ELSEIF((ANSWER.EQ.'W').OR.(ANSWER.EQ.'w'))THEN
        WRITE(*,*)' '
        WRITE(*,*)'ENTER NEW CENTER WAVELENGTH:'
        READ(*,100) LAMBDA
    ELSEIF((ANSWER.EQ.'R').OR.(ANSWER.EQ.'r'))THEN
        WRITE(*,*)' '
        WRITE(*,*)'ENTER THE NEW R0:'
        READ(*,100)R0
    ELSEIF((ANSWER.EQ.'S').OR.(ANSWER.EQ.'s'))THEN
        WRITE(*,*)' '
        WRITE(*,*)'ENTER THE NEW SECONDARY DIAMETER:'
        READ(*,100) OBSCUR
    ENDIF
50  CONTINUE
    WRITE(*,*)' '
    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(*,*)' '
        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,*)DIAM,OBSCUR,LAMBDA,R0
    CLOSE(UNIT=10)
    ENDIF

    WRITE(*,*)' '
    WRITE(*,*)'ENTER THE NUMBER OF PIXELS PER R0:'
    READ(*,*)PIXELNUM

```

```

WRITE(*,*)' '
WRITE(*,*)'ENTER SEED VALUE:'
READ(*,*)ISEED
PIXSCALE = R0/PIXELNUM
TPFDIM = nint(DIAM*PIXELNUM/R0)
SECDIM = nint(OBSCUR*PIXELNUM/R0)
60  CONTINUE
65  CONTINUE
WRITE(*,*)' '
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(*,*)' '
    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,zklmap,linptr,iseed,
+              nkl,r0,n1,n2,n3,n4,idim2,pi,pixscale,tilt)

```

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

- c 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 zklmap(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

```

- c 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(mu,sigma,y,yi,pi,iseed)
sum = cmplx(Y,YI)
sum = sum*(dr0**.83333333)
call inprod(bmap,zklmap,i,idim2,nkl,zap)

```

- c 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,zklmap,i,idim2,nkl,sum)

```

- 10 CONTINUE
- RETURN
- END

**SUBROUTINE TPF2A(PS1,PS2,IDIM,IDIM2,TPFDIM,SECDIM)**

- c   **Function:** The function of this subroutine is to cut out a portion of array with  
c       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**

- c   **Zero out ps1 to ensure that no remnants 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**

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

**TPFDIM2 = TPFDIM/2**  
**SECDIM2 = SECDIM/2**

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

- c   **Now insert the phase screen into the larger array. The smaller array is the data**  
c       **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-5,2.50662827465d0/
  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
c   deviate)

c   Author

c   numerical recipes, p207, routine poidev.for

```



c Modifications

c 13 nov, 1987: pat fitch

c 1988: erik m johansson - fixed integer conversion problem with maxint

c Routines called

c uniform

c gammln

c (c) Copyright 1987 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.

real pi

parameter (pi=3.141592654)

c maxint is the largest real number which can be converted to an integer without

c resulting in an arithmetic error (32 bits)

real maxint

parameter (maxint = .214748352e10)

real pexpmean, oldmean, t, em, sq, abxm, y, gammln,

+ uniform

integer times

data oldmean /-1./

if(pmean.lt.12.0)then

c Use direct computation method

if(pmean.ne.oldmean) then

c New mean, calculate the exponential

oldmean = pmean

pexpmean= exp(-pmean)

endif

em = -1.0

t = 1.0

2 em = em + 1.0

```

    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)

c    Natural log of gamma function = gammln

        pexpmean= pmean*abxm - gammln(pmean+1.)
    endif

    times = 0

1    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-gammln(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    K. Park and Keith W. Miller, Communications of the ACM, October 1988, Vol  
c    31, No 10, pp 1192 - 1201 routine is listed on p 1195

c    modifications

c    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    (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

c The function

```
k = seed / q
seed = a * (seed - k * q) - k * r
if (seed .lt. 0) seed = seed + m
uniform = seed * h
return
end
```

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

c THIS PROGRAM WAS DEVELOPED BY THE THESIS AUTHOR AND  
c DEVELOPS A LOW PASS FILTER IN THE FREQUENCY  
c DOMAIN TO FILTER THE RECONSTRUCTED OBJECT DATA FILE  
c FOR BOTH THE KNOX-THOMPSON AND TRIPLE-CORRELATION  
c METHODS. ADDITIONALLY, THIS PROGRAM DETERMINES THE  
c AZIMUTHAL RMS PHASE ERROR FOR THE INPUT FOURIER  
c SPECTRA.

c THE FOLLOWING SUBROUTINES ARE REQUIRED FROM  
c UNIVERSAL SUBROUTINES IN APPENDIX D:

c       Complexconj  
c       FFT  
c       FFT2D  
c       NormFFT  
c       Quadswap

c   AUTHOR:        LT JAMES M. LACKEMACHER  
c   COMPL. DATE:   26 OCTOBER 1990  
c   REASON:        COMPLETE REQUIREMENTS FOR A MASTERS  
c                   DEGREE IN PHYSICS  
c   GOAL:          SIMULATE OBJECT, DEGRADE OBJECT,  
c                   RECONSTRUCT OBJECT USING KNOX-THOMPSON  
c                   AND TRIPLE-CORRELATION METHODS FILTER  
c                   THE DATA THE COMPARE THE TWO METHODS.

### PROGRAM FREQFILTER

#### MAIN PROGRAM COMPLEX VARIABLE LIST

c   BSDATA        n x n DIM ARRAY REPRESENTING THE INPUT  
c                   TRIPLE-CORRELATION RECONSTRUCTED OBJECT IN

c		FREQUENCY SPACE
c	BSLP	n x n DIM ARRAY REPRESENTING THE LOW PASS
c		FILTERED TRIPLE-CORRELATION RECONSTRUCTED
c		OBJECT IN FREQUENCY SPACE
c	F	n DIM ARRAY USED IN THE FOURIER TRANSFORM
c	KTDATA	n x n DIM ARRAY REPRESENTING THE INPUT
c		KNOX-THOMPSON RECONSTRUCTED OBJECT IN
c		FREQUENCY SPACE
c	KTLP	n x n DIM ARRAY REPRESENTING THE LOW PASS
c		FILTERED KNOX-THOMPSON RECONSTRUCTED
c		OBJECT IN FREQUENCY SPACE
c	TEMPDATA	n x n DIM ARRAY THAT IS USED AS A TEMPORARY
c		ARRAY IN THE FOURIER TRANSFORM
c	TRUDATA	n x n DIM ARRAY REPRESENTING THE TRUTH DATA

c                   MAIN PROGRAM REAL VARIABLE LIST

c	berror	n x n DIM ARRAY THAT REPRESENTS THE SQUARE
c		PHASE ERROR FOR THE TRIPLE-CORRELATION
c		RECONSTRUCTED IMAGE
c	bslpmmod	n x n DIM ARRAY THAT REPRESENTS THE MODULUS
c		OF THE LOW PASS FILTERED TRIPLE-CORRELATION
c		RECONSTRUCTED IMAGE
c	bsr	n/2 DIM ARRAY THAT REPRESENTS THE RADIAL
c		FREQ VALUE IN INVERSE ARCSECONDS FOR THE
c		TRIPLE-CORRELATION IMAGE
c	bssnr	n/2 DIM ARRAY THAT REPRESENTS THE INPUT SNR
c		VALUES OF TRIPLE-CORRELATION IMAGE
c	bstrunc	TRIPLE-CORRELATION RADIAL TRUNCATION VALUE
c	kterror	n x n DIM ARRAY THAT REPRESENTS THE SQUARE
c		PHASE ERROR FOR THE KNOX-THOMPSON
c		RECONSTRUCTED IMAGE
c	ktlpmmod	n x n DIM ARRAY THAT REPRESENTS THE MODULUS
c		OF THE LOW PASS FILTERED KNOX-THOMPSON
c		RECONSTRUCTED IMAGE
c	ktr	n/2 DIM ARRAY THAT REPRESENTS THE RADIAL
c		FREQ VALUE IN INVERSE ARCSECONDS FOR THE
c		KNOX-THOMPSON IMAGE
c	ktsnr	n/2 DIM ARRAY THAT REPRESENTS THE INPUT SNR
c		VALUES OF KNOX-THOMPSON IMAGE

c	kttrunc	KNOX-THOMPSON RADIAL TRUNCATION VALUE
c	lambda	WAVELENGTH OF QUASI-MONOCHROMATIC LIGHT
c	rBSError	n/2 DIM ARRAY THAT REPRESENTS THE AZIMUTHAL
c		RMS PHASE ERROR OF THE TRIPLE-CORRELATION
c		RECONSTRUCTED IMAGE
c	rKTerror	n/2 DIM ARRAY THAT REPRESENTS THE AZIMUTHAL
c		RMS PHASE ERROR OF THE KNOX-THOMPSON
c		RECONSTRUCTED IMAGE
c	R0	COHERENCE LENGTH

c                   MAIN PROGRAM INTEGER VARIABLE LIST

c	fwd	VALUE OF 1 FOR FORWARD FFT
c	iktcoun	NUMBER OF KT PIXELS WITH SNR GREATER THAN 1.0
c	inv	VALUE OF -1 FOR INVERSE FFT
c	itcoun	NUMBER OF TC PIXELS WITH SNR GREATER THAN 1.0
c	ln	2^ln FOR USE WITH FFT SUBROUTINE
c	n	DIMENSION OF ONE SIDE OF 2-DIM ARRAY
c	nyquist	EQUAL TO THE TELESCOPE PUPIL FUNCTION
c		DERIVED FROM THE FOLLOWING FORMULA:
c		nyquist = (telescope diameter x number of pixels/r0)/r0
c	numpix	THE NUMBER OF PIXELS PER R0

c                   MAIN PROGRAM INTEGER VARIABLE LIST

c	end	CHAR INPUT TO DETERMINE WHETHER TO STOP THE
c		PROGRAM
c	mean	CHAR INPUT TO DETERMINE WHETHER TO FIND THE
c		AMSPE
c	rite	CHAR INPUT TO DETERMINE WHETHER TO WRITE
c		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), bsipmod(n,n), kterror(n,n),  
+ bserror(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,kttrunc,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

```

```

c   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

c   READ IN DATA

        CALL Readtrufile(TRUDATA,n)

c   DETERMINE SQUARE PHASE ERROR

        CALL AMSPE(KTDATA,BSDATA,TRUDATA,
+          kterror,bserror,nyquist,n)

c   DETERMINE AZIMUTHAL AVERAGE OF THE RMS PHASE ERROR

        CALL AMSPEcalc(kterror,bserror,rKTerror,rBSerror,
+          nyquist,n)

c   WRITE AMSPE TO A FILE FOR EACH CORRELATION TECHNIQUE

        CALL Writerrfile(rKTerror,rBSerror,nyquist,lambda,
+          R0,numpix,iktcount,itccount,n)

    ENDIF

    STOP
    END

```

## SUBROUTINE LIST

SUBROUTINE AMSPE(KTDATA, BSDATA, TRUDATA,  
+ kterror, bserror, nyquist, n)

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

```

COMPLEX*16 KTDATA(n,n), BSDATA(n,n), TRUDATA(n,n)
REAL*8 kterror(n,n), bserror(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
  
```

```

60      IF (bsphase.GT.pi2) THEN
          bsphase = bsphase - pi2
          GOTO 60
        ENDIF
70      IF (bsphase.LT.-pi2) THEN
          bsphase = bsphase + pi2
          GOTO 70
        ENDIF
        kterror(i,j) = truphase - ktphase
80      IF (kterror(i,j).GT.pi2) THEN
          kterror(i,j) = kterror(i,j) - pi2
          GOTO 80
        ENDIF
90      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
        IF (kterror(i,j).LT.-pi)
+         kterror(i,j) = kterror(i,j) + pi2
        kterror(i,j) = (kterror(i,j))**2.0

        bserror(i,j) = truphase - bsphase
100     IF (bserror(i,j).GT.pi2) THEN
          bserror(i,j) = bserror(i,j) - pi2
          GOTO 100
        ENDIF
110     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
        IF (bserror(i,j).LT.-pi)
+         bserror(i,j) = bserror(i,j) + pi2
        bserror(i,j) = (bserror(i,j))**2.0
        ENDIF
10     CONTINUE
        RETURN
        END

```

```

SUBROUTINE AMSPEcalc(kterror,bserror,rKTerror,
+                   rBSerror,nyquist,n)

```

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

```

REAL*8 kterror(n,n), bserror(n,n), rKTerror(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
        rKTerror(r) = rKTerror(r) + kterror(i,j)
        rBSerror(r) = rBSerror(r) + bserror(i,j)
      ENDIF
20  CONTINUE
    rKTerror(r) = dsqrt(rKTerror(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)

- c 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)

- c THIS S/R READS THE RECONSTRUCTED DATA FROM A FILE

```
COMPLEX*16 KTDATA(n,n), BSDATA(n,n)
CHARACTER*16 ktfile, bsfile
WRITE(*,*) '
WRITE(*,*)'ENTER INPUT KT RECON FILE NAME (16 CHAR'
WRITE(*,*)'MAX):'
READ (*,30) ktfile
WRITE(*,*) '
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):'

```

```

      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,R0,n)

- c THIS S/R WRITES THE DATA TO A FILE

```

REAL*8 ktlpdata(n,n), bslpdata(n,n)
REAL conv, lambda, R0, x, y
CHARACTER*16 ktlpfile, 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)

```

```

        WRITE(30,60) x, y, ktldata(i,j)
10    CONTINUE
        DO 20 i = 1, n
            DO 20 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)
                WRITE(40,60) x, y, bsldata(i,j)
20    CONTINUE
50    FORMAT(A16)
60    FORMAT(F7.4,2x,F7.4,2x,F7.4)
        RETURN
        END

```

```

SUBROUTINE Writerrfile(rKTerror,rBSError,nyquist,
+                    lambda,R0,numpix,iktcount,
+                    itccount,n)

```

c 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(*,*) '
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, iktcount
    x = r * numpix * (lambda/R0) * (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<br>Cameron Station<br>Alexandria, Virginia 22304-6145  | 2 |
| 2. | Library Code 52<br>Naval Postgraduate School<br>Monterey, California 93943-5002   | 2 |
| 3. | Professor Donald L. Walters, Code PH/We<br>Department of Physics<br>Naval Postgraduate School<br>Monterey, California 93943-5000    | 5 |
| 4. | Professor David S. Davis, Code PH/Dv<br>Department of Physics<br>Naval Postgraduate School<br>Monterey, California 93943-5000       | 1 |
| 5. | Professor Karlheinz E. Woehler, Code PH/Wh<br>Department of Physics<br>Naval Postgraduate School<br>Monterey, California 93943-5000 | 1 |
| 6. | Capt Charles L. Matson<br>Weapons Lab/ARCI<br>Kirtland Air Force Base<br>Albuquerque, New Mexico 87117                              | 1 |
| 7. | Capt Michael C. Roggemann<br>Weapons Lab/ARCI<br>Kirtland Air Force Base<br>Albuquerque, New Mexico 87117                           | 1 |

8. LT James M. Lackemacher  
USS Gallery (FFG-26)  
FPO, AA 34093-1482

2