A Monte Carlo ray trace tool for predicting contrast in naval scenes including the effects of polarization

Joseph Maniscalco

Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Dr. J.R. Mahan, Chairman
Dr. H.H. Robertshaw
Dr. B. Vick

October 31, 2002
Blacksburg, Virginia

Keywords: Monte Carlo ray-trace, polarization, sea-slope Statistics

A Monte Carlo ray trace tool for predicting contrast in naval scenes including the effects of polarization

Joseph Maniscalco

(ABSTRACT)

The survivability of U.S. warships has become a higher priority than ever before. Two ways to improve survivability are to either avoid damage, or to continue to operate after damage has been incurred. This thesis concentrates on the first line of defense, which involves the first of these two approaches. Specifically, this thesis evaluates the extent of threat due to optical contrast with the ocean background.

As part of this effort, an MCRT tool was created that allows the user to vary the shape and surface properties of a ship. A reverse MCRT was performed in order to reduce the processing time required to get accurate results. Using this MCRT tool, the user can determine the theoretical contrast with the ocean surface that would be seen at any viewing angle with and without a polarization filter. The contrast due to differential polarization and a change in viewing angle is estimated to determine the extent of threat. These results can be determined for both daytime and nighttime conditions by specifying if the ray trace is in the infrared or visible light range. The location of the sun for daytime conditions, and the temperature of the surfaces for nighttime conditions, can all be adjusted by the user.

In order to get an accurate estimation of the signal power coming from the ocean surface, a great deal of time and effort was spent modeling the ocean surface. Many studies have been done concerning the slope statistics of an ocean surface, some more informative than others. This thesis takes two of the most complete studies and brings them together to get accurate slope statistics in both along-wind and crosswind directions.

An original idea by the author was used to give a typical shape to the waves of the simulated ocean surface. The surface properties of the ship were determined using Fresnel's equations and the complex index of refraction of water at the particular wavelengths of interest.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. J.R. Mahan for his guidance and enthusiasm, which was such a great motivation to me. I am also thankful for his determination in keeping me funded during the last year of my degree.

I would like to express my appreciation to Dr. Vick and Dr. Robertshaw for serving on my advisory committee. I am also grateful that my committee members were able to meet for my defense on such short notice.

I would also like to thank Greg Pickett for writing a utility program that aided me in creating many of the figures shown in this thesis.

Last, but not least, I would like to thank my parents for their support and motivation, especially during the last few months before my defense.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 Introduction

1.1 Motivation

In the past few decades, the American public has become increasingly sensitive to wartime casualties, and the cost of U.S. warships continues to increase as technology advances. For example, the USS Cole, the destroyer damaged in the Yemeni terrorist attack in 2000, cost $250 million to repair. For these reasons, and many others, the survivability of U.S. warships has become a higher priority than ever before. The two ways to increase survivability are to either avoid damage, or to continue to operate after damage has been incurred [King, 2001]. The focus of this thesis is on the first of these two approaches. Specifically, this thesis evaluates the extent of the threat due to optical contrast with the ocean background. The reflectivity of the ocean surface varies with incident angle and polarizes the light that it reflects, while the current ship surfaces do not [Cooper, Lentz and Walker, 2002]. This contrast due to differential polarization and viewing angle is evaluated to determine the extent of the threat.

1.2 Research goals

Stealth is becoming a fundamental design and operation aspect of warships, and so it must be part of the design process from the beginning. The main goal of this research is to create a tool that can be used to aid in the design of stealthy ships throughout the design process. Even though the program presented in this thesis is capable of evaluating any ship geometry, only the surface properties of the ship are discussed in detail. The basic idea behind this design tool is to examine the light emitted and reflected from the ocean and ship to an optical sensor in order to generate a predicted image. Figure 1.1 shows an example of a predicted image of three DD(X) destroyers stationary in a seaway.

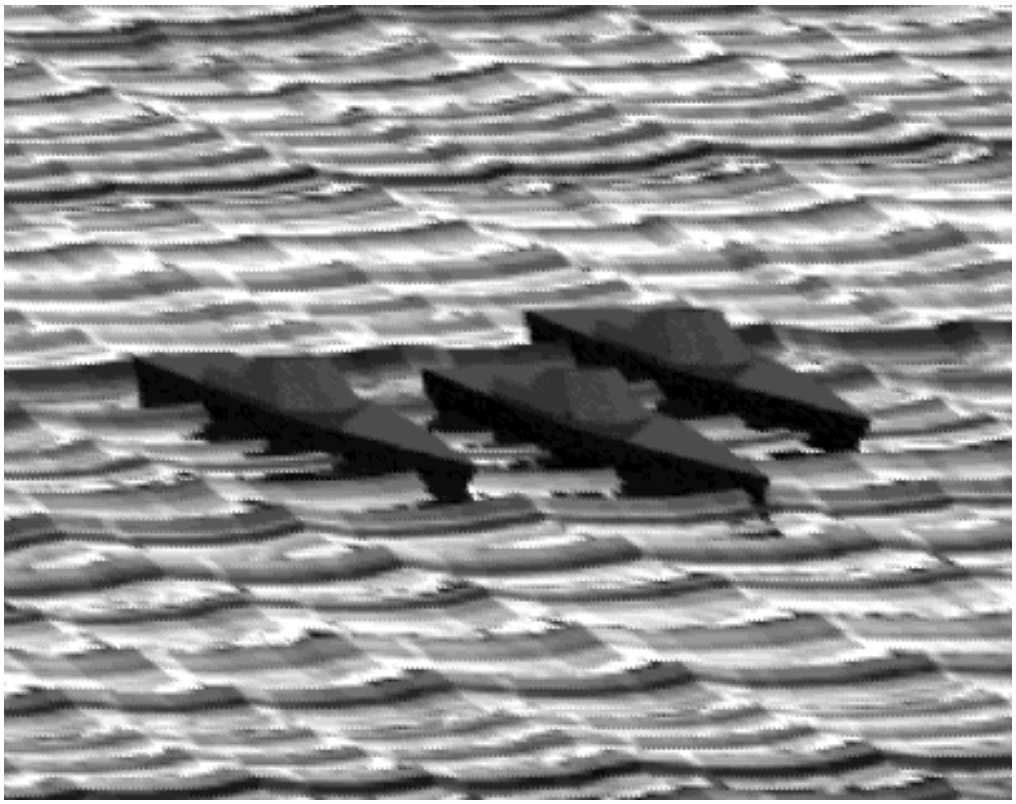


Figure 1.1 Predicted visible image of three DD(X) destroyers

Since the design of U.S. warships can only be as good as the tools we use to create the design, it is crucial that the latest and most accurate methods to create these tools be used. The design tool presented here uses a reverse Monte Carlo ray-trace (MCRT) method. The MCRT method has been shown to produce less than one percent error when used correctly. The two most important factors that determine the accuracy of a ray trace are the accuracy with which geometry and material properties are represented [Mahan, 2002]. For this particular application, it was critical to create an ocean surface whose geometry and optical properties closely represent those of a real seaway. Less concern was given for the geometry and optical properties of the ship, since these factors vary among ship concepts, and this MCRT tool was created to help determine their values. The ship geometry chosen to test this design tool is the DD(X) destroyer. This concept was selected so that next generation warships could be evaluated for infrared and

visible stealth. Figure 1.2 shows a rendered view of the geometry that was generated for this study.

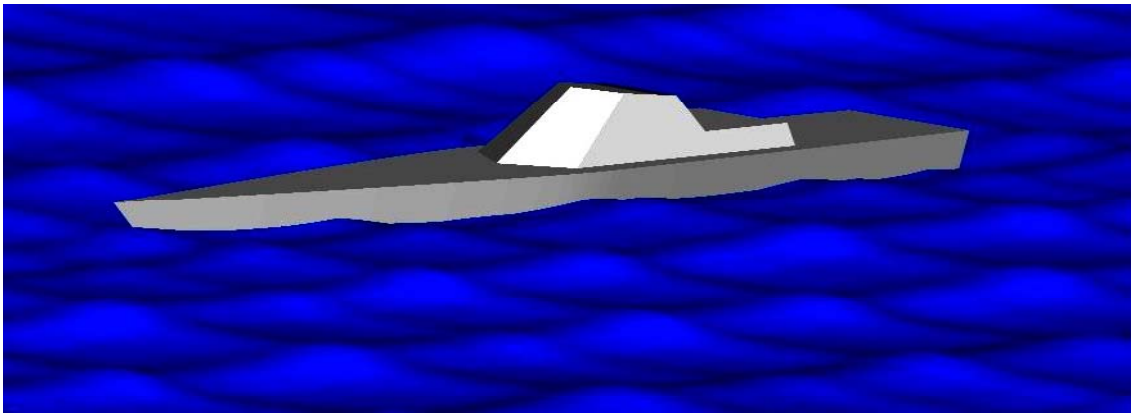


Figure 1.2 Next-generation warship geometry

1.3 Organization

Chapter 2 covers all the theory used to create the design tool, including the method for creating the geometry, polarized light theory, and MCRT theory. The section on geometry covers the statistics of a real ocean surface and how the statistics are used to simulate a typical ocean surface. The method for creating the ship geometry used to produce the results in Chapter 4 is also briefly discussed. Chapter 2 gives an introduction to polarized light theory, including what it is and how it is useful in detecting ships at sea. The last section of Chapter 2 fully explains the MCRT theory used for the design tool. Every step of tracing an energy bundle, from emission to absorption, is explained. The method for organizing the information gained from this analysis is also described in this section of Chapter 2.

Chapter 3 describes how to use each program and how each program is organized. This is done through the use of a users' manual and a programmers' manual. Both of these manuals, integral to the thesis, treat all four of the programs making up the MCRT design tool. The first program, **ocean**, is used to create the geometry for the ocean surface. The second program, **ship**, is used to create the DD(X) ship geometry at a specified location and orientation with respect to the ocean surface. The third program,

called **MCRT**, traces a user-specified number of rays and stores all relevant information about each ray. The final program, **Post**, creates the predicted image of the ship and ocean surface from the information obtained from the ray trace.

Chapter 4 gives the results of all the different test cases considered in this effort. The first section of Chapter 4 covers the visible light ray traces, while the second section covers the infrared ray traces. The surface properties of the ship and the viewing direction of the optical sensor are varied for both types of ray trace. The visible-light ray trace also shows the effects of variation in the location of the sun. The different test cases are discussed and the role played by polarization is identified.

Finally, Chapter 5 summarizes the results and makes conclusions based on the test cases evaluated in Chapter 4. Other possible applications of these programs are discussed, and future work that could expand the capabilities of these programs is suggested.

CHAPTER 2 Computer model theory

This chapter treats the theory used to create the computer model referred to in Chapter 1. Three main topics are discussed: geometry, polarization theory, and MCRT theory. The section on geometry covers the statistics of a real ocean surface and how the statistics have been used to simulate a typical ocean surface. Ship geometry is also briefly discussed in that section. The next section gives an introduction to polarized light theory, including what it is and how it is useful in detecting ships. The MCRT theory used for both the ocean surface and the ship surface is then developed. The last part of this chapter deals with keeping track of the polarization of light during a ray trace.

2.1 Geometry of ship and ocean surfaces

This section provides a detailed description of how all the geometry for the computer model was created. Two main steps are needed to create the ocean surface. The first is to determine the slope statistics that can be stored in the form of probability density functions (PDF's). The second step is to use those statistics to create a realistic, typical ocean surface. The final product is a statistically correct ocean formed by an array of rectangular facets. The ship, which is discussed in the last part of this section, has a more complex geometry and was created entirely using triangular facets.

2.1.1 Ocean Statistics

The topography of the ocean surface exerts a strong influence on how sunlight is reflected to an optical sensor. Sea-surface slopes are primarily composed of short, low amplitude waves that depend strongly on wind speed and air-sea temperature difference [Shaw and Churnside, 1997]. The method used in this thesis for modeling the ocean surface is to create a surface that is statistically equivalent to a real ocean surface. Several papers present measurements of sea-surface slope statistics, but the most complete set of measurements were obtained by Cox and Munk [1954] using their classic

sun-glitter photographs. The measurements were organized as slope probability density functions (PDF's) that vary as a function of wind speed ranging from 1 to 14 m/s. Statistics of the ocean surface along both the direction of the wind and perpendicular to the direction of the wind are reported. Unfortunately, these data do not include information on the air-sea temperature dependence. In a more recent paper, Shaw and Churnside [1997] obtained measurements in the along-wind direction using a scanning-laser glint sensor. These data are presented as slope PDF's as a function of wind speed, with air-sea temperature difference as a parameter. Shaw and Churnside present no discussion of the measurements taken in the crosswind direction, so a combination of these two data sets is used in this thesis. The along-wind PDF's from Shaw and Churnside, and the crosswind PDF's from Cox and Munk are used in the present effort. It is important to note that the accuracy of the surface statistics decreases as the magnitude of the air-sea temperature difference increases. This is particularly true for negative (air colder than water) stability. Since Cox and Munk made their measurements when the air-sea temperature difference was near zero or slightly positive, the crosswind statistics are strictly valid only for these temperatures. Figure 2.1 shows the slope distribution of the ocean surface for both crosswind and along-wind directions.

Although both PDF's appear to have a Gaussian distribution, the along-wind PDF has some significant deviations from a standard Gaussian distribution. These deviations carry important information about the surface roughness and can be described by the coefficients of skewness and kurtosis. To obtain an analytical estimate of the slope distribution, a Gram-Charlier series [Shaw and Churnside, 1997] was used. The PDF's were first expanded in terms of a normalized slope,

$$\eta = \frac{\theta - \bar{\theta}}{\sigma} \tag{2.1}$$

where θ is the slope with respect to nadir, $\bar{\theta}$ is the mean slope angle, and σ is the measured PDF standard deviation. The probability density function was then represented as the series

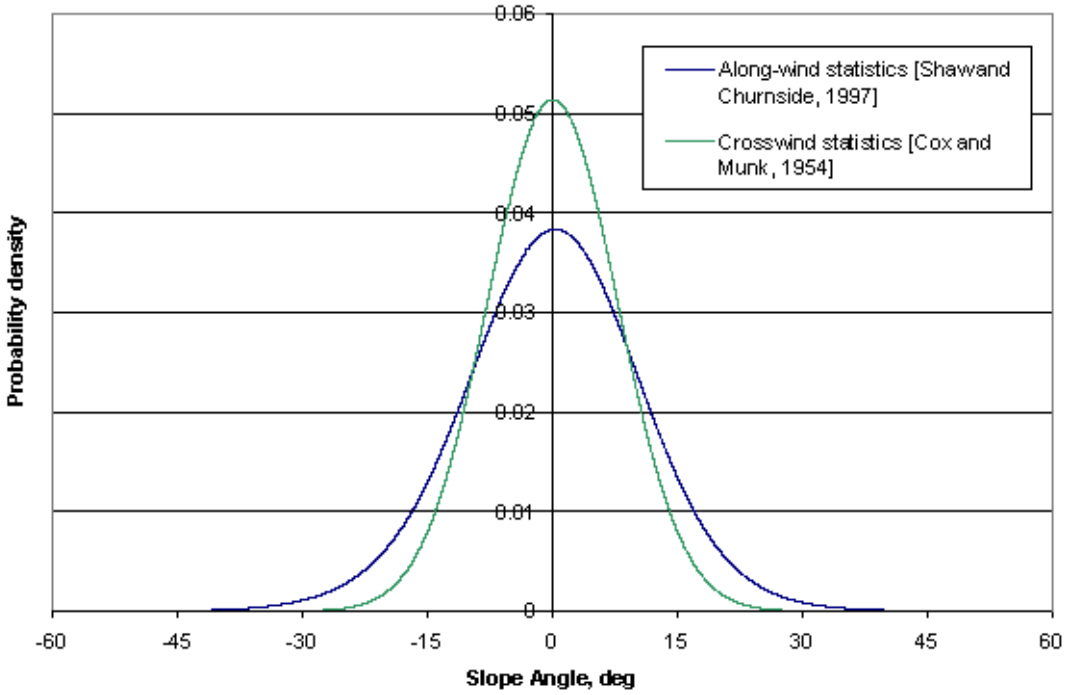


Figure 2.1 Along-wind sea slope statistics according to Shaw and Churnside [1997] and crosswind sea slope statistics according to Cox and Munk [1954]

$$P_{AW}(\eta) = \sum_{n=0}^{N} \left(\frac{c_n}{n!} \right) H_n(\eta) G_n(\eta) \tag{2.2}$$

where $G(\eta)$ is a zero-mean, unit-variance Gaussian distribution,

$$G(\eta) = \left(\frac{1}{\sqrt{2\pi}} \right) \exp\left(\frac{-\eta^2}{2} \right) \tag{2.3}$$

$H_n(\eta)$ are nth-order polynomials listed in Table 2.1, and c_n are the expansion coefficients that were calculated using the collected data. The coefficients $c_1 = c_2 = 0$, and c_3 and c_4 are given by

$$c_3 = -0.2349 + 4.812\sigma^2 \tag{2.4}$$

$$c_4 = 5.505 - 31.12\sigma^2 \tag{2.5}$$

where σ is the standard deviation found using

$$\sigma^2 = .00449U \tag{2.6}$$

where U is the wind speed in m/s. This equation is only valid for an air-sea temperature difference of zero, which is the situation assumed for the ocean surface used in this thesis. Since the crosswind PDF's do not contain any skewness or kurtosis, the probability density function can be simply written as

$$P_{CW}(\eta) = G_n(\eta) \tag{2.7}$$

where $G_n(\eta)$ is the Gaussian distribution given in Equation 2.3.

Table 2.1 Hermite polynomials of order n [Shaw and Churnside, 1997]

n	**$H_n(\eta)$**
0	1
1	η
2	η^2 - 1
3	η^3 - 3η
4	$\eta^4 - 6\,\eta^2 + 3$

2.1.2 Ocean Surface

The main goal in creating the statistical ocean surface model was to have a statistically correct surface while giving the waves a realistic shape. A typical shape for the short choppy waves that were modeled, as observed by the author, is shown in Figure 2.2.

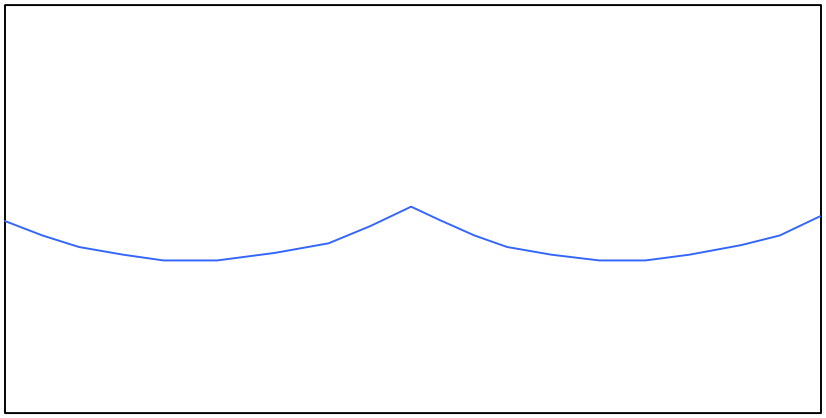


Figure 2.2 Typical shape of an ocean wave

Waves having the general shape illustrated in Figure 2.2 dominate the sea-surface statistics and therefore are the only type of waves modeled in this effort. To put the information from the PDF into a useful form, it is integrated to give a function that varies from zero to unity as shown in Figure 2.3. Then in order to force the wave to conform to the desired shape, the curve in Figure 2.3 is divided into sections that correspond to different parts of the wave. The method used to obtain the shape in Figure 2.2 involved organized random number generation using these sections. This means that the uniformly distributed random numbers used were limited to one section at a time, and therefore to a specific range of slopes, corresponding to the section of the wave being generated. For example, if 40 facets are used to describe a wave, ten random numbers from Section 1, ten from Section 2, and so forth, were used. Each facet was then assigned a length and assembled in the order chosen. The result was shown in Figure 2.2. This method was used to generate a series of waves in both the crosswind and the along-wind directions. The number of facets used for each wave was minimized to get the maximum number of waves using a least amount of facets while still maintaining a smooth wave shape. Each wave has between 8 and 16 facets, randomly determined, so that the height and length of the waves are not uniform. The final ocean surface generated used the crosswind wave series to determine all of the starting points for the along-wind wave series. Figure 2.4 graphically shows how the ocean is generated.

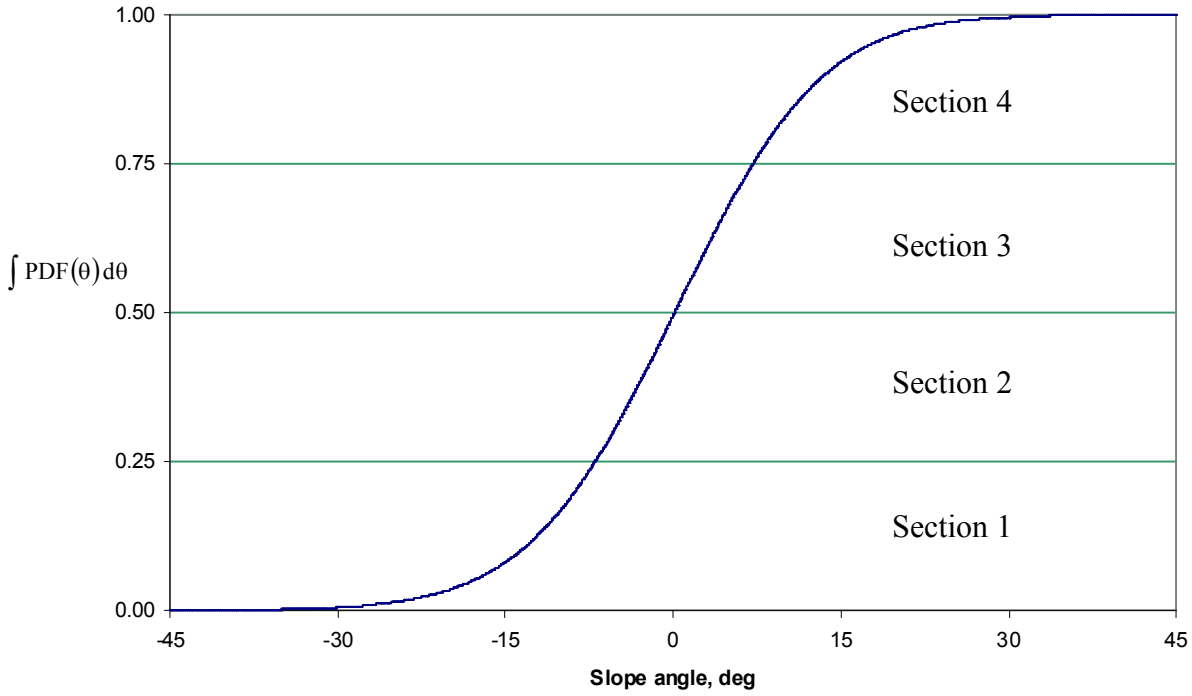


Figure 2.3 Division of sea waves into sections

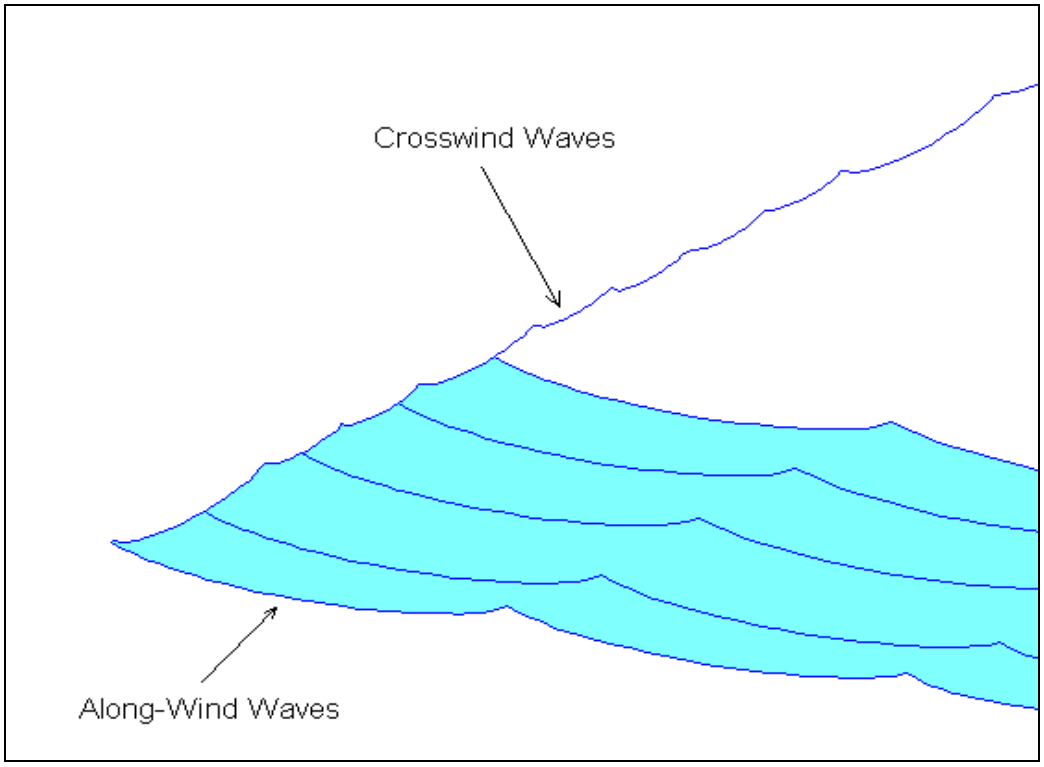


Figure 2.4 Assembly of the waves into a seaway

All the along-wind waves are identical to ensure the original crosswind wave is retained throughout the surface. This guarantees that the crosswind statistics are maintained. The final result is illustrated in Figure 2.5 for a wind speed of 8 m/s and a temperature difference of 0.0 degrees.

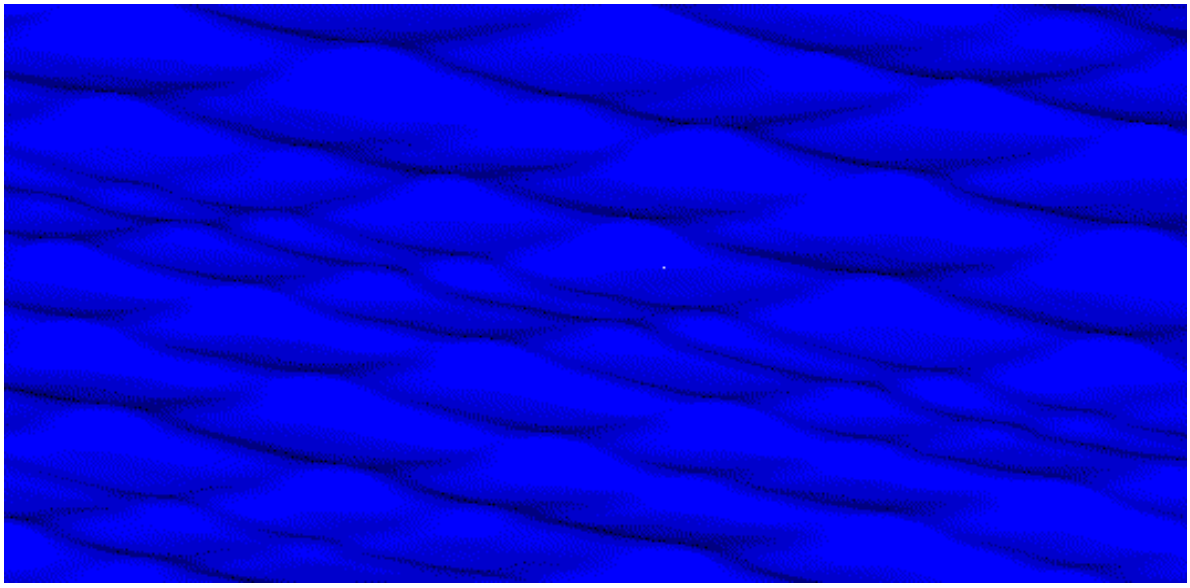


Figure 2.5 Rendered view of a typical seaway corresponding to a wind speed of 8 m/s and a temperature difference of 0.0 degrees

2.1.3 Ship geometry

The geometry of the ship used for this computer model was created to closely resemble the new class of DD(X) U.S. Navy destroyers. The coordinates for all the facets of the ship were approximated based on a rendered concept drawing [http://www.naval-technology.com/projects/dd21/]. Figure 2.6 shows the picture that was used to approximate the geometry.

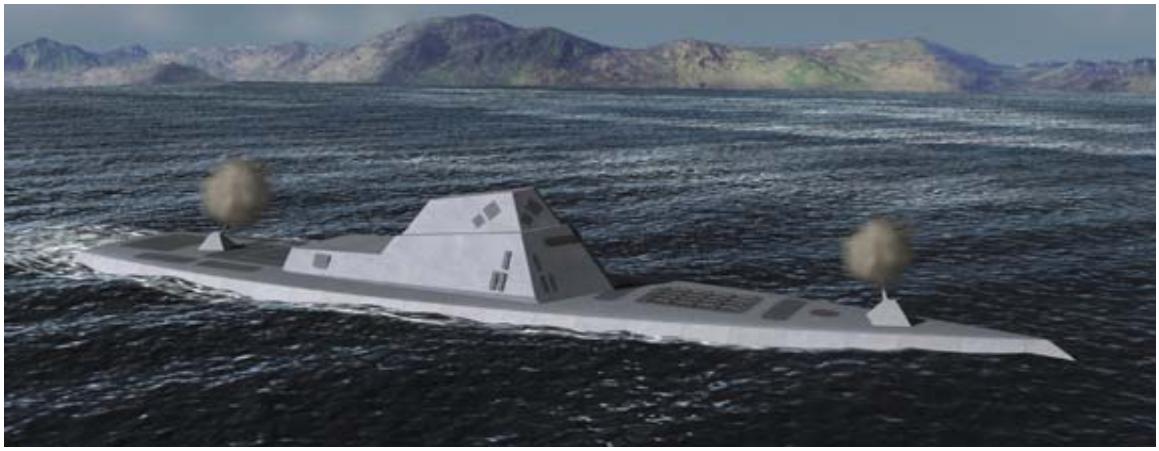


Figure 2.6 Rendered concept drawing of the DD(X) destroyer

Figure 2.7 shows a rendered view of the ship that was used in the computer model, stationary, in a confused seaway.

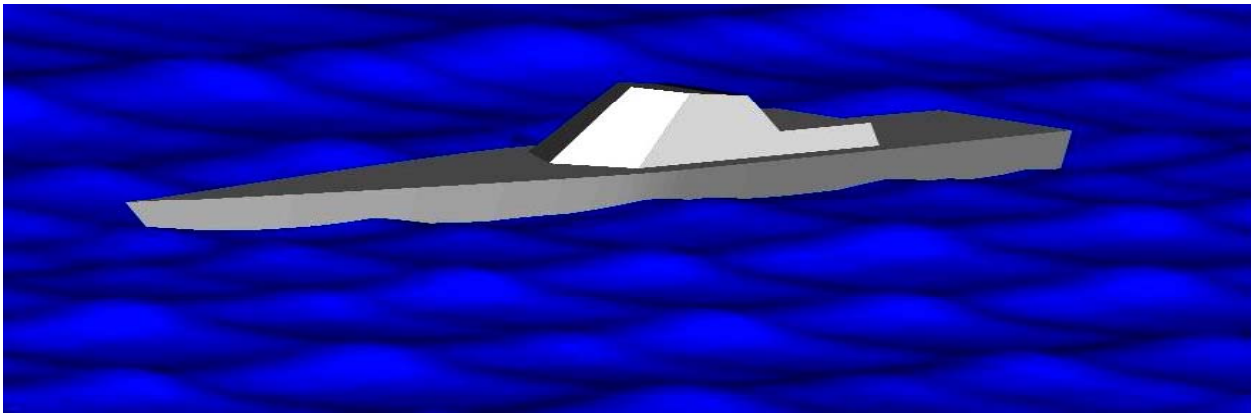


Figure 2.7 Rendered drawing of the next-generation destroyer concept used in the computer model

2.2 Polarization

Light emitted from the sun is considered to be unpolarized, or randomly polarized. This type of light is equivalent to two perpendicular electromagnetic waves propagating along the same vector [Mahan, 2002]. Figure 2.8 is a graphical representation of the electronic component of these two waves traveling along the propagation vector.

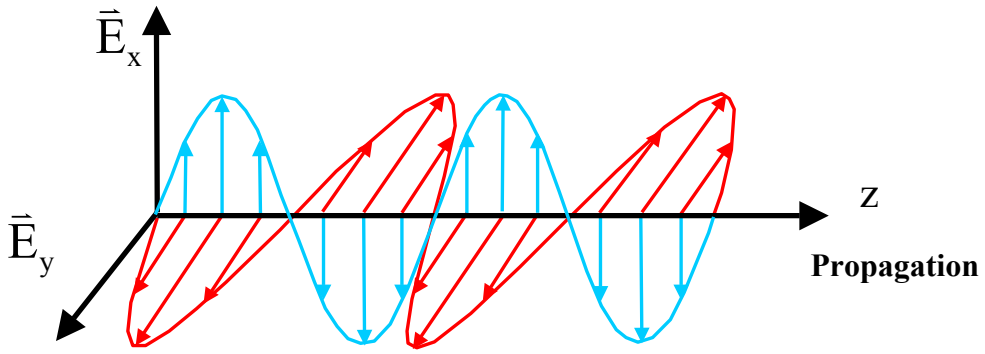


Figure 2.8 Representation of the two electrical components of an unpolarized wave

When one of these waves has a larger amplitude than the other, the light is considered polarized. Several different mechanisms are responsible for polarizing light, including transmission, refraction, scattering, and reflection [http://www.physicsclass-room.com/Class/light/U12L1e.html#refln]. The only mechanism considered in the current effort is polarization by reflection.

Light will become polarized when the molecular structure of a surface is such that a wave whose electric field component is perpendicular to the surface reflects with a different amplitude than a wave whose electric field component is parallel to the surface. This will occur to some degree with all non-metallic surfaces, including seawater [http://www.physicsclassroom.com/Class/light/U12L1e.html#refln]. The degree of polarization will depend on the difference in the reflectivities associated with the two components of polarization and the angle of incidence with the surface. Figure 2.9 shows these two reflectivities along with the net reflectivity of an ideal seawater surface at 400 nm.

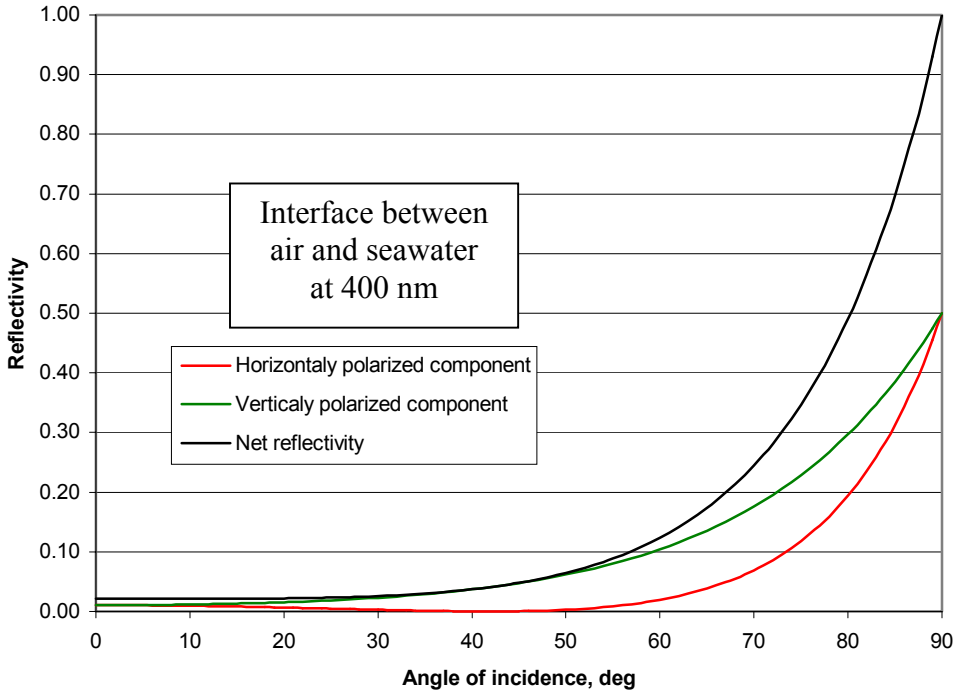


Figure 2.9 Reflectivity of an ideal plane seawater surface at 400 nm

Figure 2.10 is a graphical representation of a reflection in which the ray is completely polarized in the vertical direction. The green vectors represent the electric wave component parallel to the surface and the red vectors represent the electric wave component perpendicular to the surface.

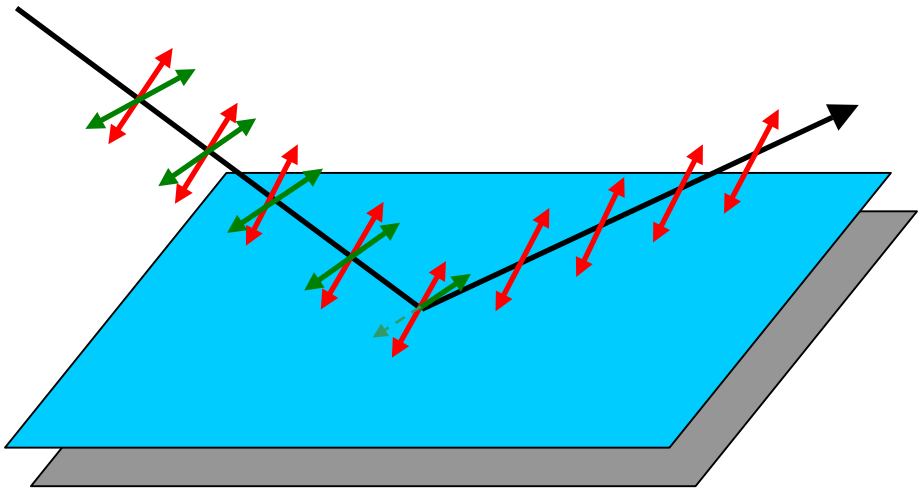


Figure 2.10 Illustration of an idealized polarized reflection

2.3 The Monte Carlo ray trace

In general, a Monte Carlo ray trace involves tracing energy bundles along their straight-line paths, called rays. The direction of the ray depends on the properties of the object that it is emitted or reflected from. Each energy bundle is reflected until it is absorbed by a surface. The purpose of the ray trace is to calculate radiation distribution factors. These distribution factors hold all the information needed to perform any radiation heat transfer analysis. The first part of this section discusses the methods that were used to formulate a computer model of this process. The next section explains how each surface was treated during a ray trace. The three subsequent sections explain how the rays were traced; including the method for emitting rays, the method for determining the point of intersection of a ray and a surface, and finding the direction of reflection for a ray. The last section explains how the distribution factors were calculated and used for each polarization.

2.3.1 The Monte Carlo ray-trace (MCRT) method

The ray trace that is used for the computer model developed in this effort is called a reverse MCRT. This means that all rays begin at the final destination of a ray and are traced back to the origin of that ray. This type of ray trace was chosen because the only rays that are important are those that end at the optical sensor; therefore, it is much more efficient to trace from the optical sensor back to the scene [Turk, 1994]. Two different methods, the traditional method and the line-of-sight method, were used to formulate this computer model. The traditional method involves tracing rays down to the ocean and the ship where they are either absorbed or reflected up to the sky without loss of power. In this case, surface properties control whether the ray is reflected or absorbed. When using the line-of-sight method, all the rays are reflected up to the sky but their power is reduced by a factor that is determined by the surface properties. The traditional method was used for the infrared ray trace in which the ocean, ship and sky are all emitters. The line-of-sight method was used for the visible-light ray trace where only the sky is an emitter. For this case the line-of-sight method greatly reduces the number of energy bundles that need to be emitted because those absorbed by the ocean or the ship are not traced.

2.3.2 Surface properties

Three types of "surface" are of interest in this computer model: the ocean, the ship, and the sky. All relevant information concerning these surfaces is discussed in this section.

2.3.2.1 Ocean surface properties

The ocean reflects all rays in the specular direction [Sidran, 1981] and is the only surface in this model that polarizes light. A specular reflection, also referred to as mirror-like, is graphically shown in Figure 2.11.

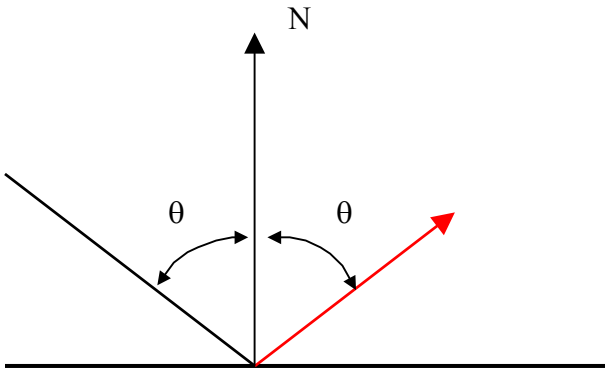


Figure 2.11 Illustration of specular reflection

The reflectivity of the ocean surface varies with the incident zenith angle of the ray and was determined following a formulation given in standard radiation and optics texts of which Mahan [2002] is an example. The reflectivity of the water for the two polarizations of radiation are given by

$$\rho_p = \left(\frac{n\cos(\theta_i) - \cos(\theta_t)}{n\cos(\theta_i) + \cos(\theta_t)} \right)^2 \tag{2.8}$$

and

$$\rho_s = \left(\frac{\cos(\theta_i) - n\cos(\theta_t)}{\cos(\theta_i) + n\cos(\theta_t)} \right)^2 \tag{2.9}$$

where the subscript *p* indicates that the orientation of the electric field is parallel to the incident surface, the subscript *s* indicates that the orientation of the electric field is perpendicular to the incident surface, θ_i is the incident angle of the ray measured from the normal, *n* is the complex index of refraction of water, and θ_t is an artificial angle defined as

$$\theta_t = A\cos\left(\sqrt{1 - \left(\frac{\sin(\theta_i)}{n} \right)^2} \right) \tag{2.10}$$

The complex index of refraction of water is documented for various wavelengths by Hale [1973]. According to Hobson [1971], the complex index of refraction of water can be used to calculate the reflectivity of seawater with little or no error. The reflectance of visible light has almost no error and the reflectance of infrared light has an error of approximately five percent.

2.3.2.2 Ship surface properties

The ship is assumed to reflect following the diffuse-specular approximation applied to the properties of the paint on the ship [Mahan, 2002]. This means that a certain percentage of the incident rays are reflected in a specular manor and the rest are reflected diffusely, that is

$$\rho = \rho^s + \rho^d \tag{2.11}$$

where ρ is the sum of ρ^s, the specular component of the reflectivity and ρ^d, the diffuse component of the reflectivity. Figure 2.12 illustrates a diffuse-specular reflection.

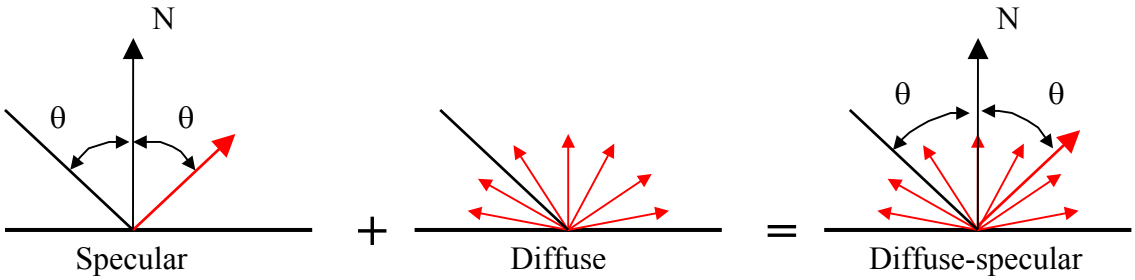


Figure 2.12 Illustration of diffuse-specular reflection

The surface properties of the ship are specified using the reflectivity, ρ, and the specularity ratio,

$$r^s = \frac{\rho^s}{\rho^s + \rho^d} \tag{2.12}$$

Since these ray traces keep separate track of both polarizations of light and the ship surface does not polarize light, the total reflectivity must be divided into equal parallel and perpendicular reflectivities following

$$\rho_s = \rho_p = 0.5\rho \tag{2.13}$$

2.3.2.3 Sky power

The source strength of the sky varies with location, and increases with proximity to the sun. This distribution of sky power is necessary for the visible-light ray trace, since the intensity of the sky is used to determine the initial power of each ray. Most of the sky is nearly uniform, but significant changes occur within the solar aureole [Grant, Heisler, and Gao, 1995]. The intensities in this region around the sun are shown in Figure 2.13 [Green, 1976]. An exponential curve,

$$I_{sky} = 0.15 + 0.0379\,\Psi^{-0.7491} \tag{2.14}$$

was fit to these data so that an equation could be used to approximate the intensity of the circumsolar sky. In Equation 2.13, Ψ is the viewing angle of a given location relative to the angle of the sun, as shown in Figure 2.14.

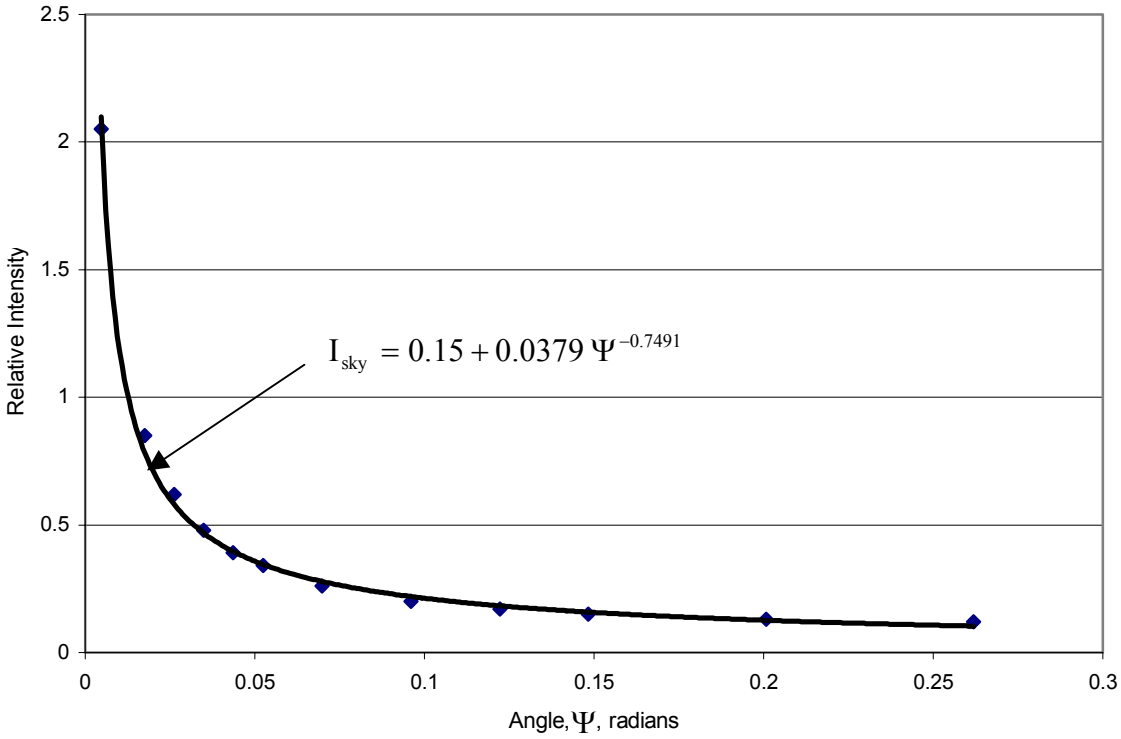


Figure 2.13 Solar aureole data and an exponential curve fit to the data [Green, 1976]

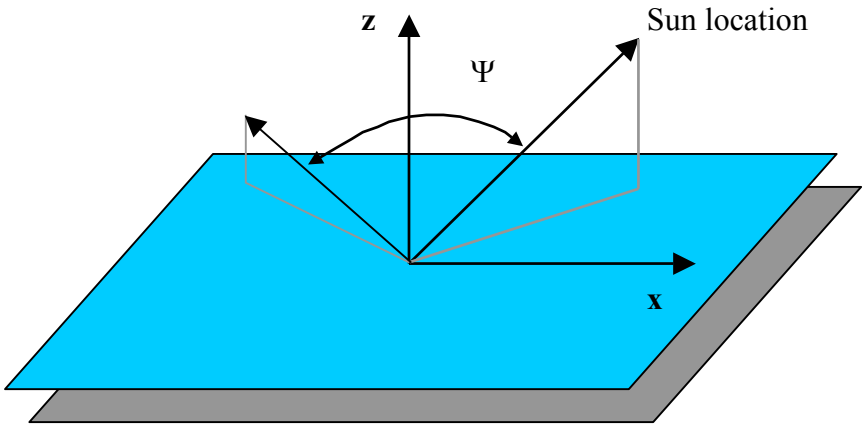


Figure 2.14 Definition of the angle Ψ

2.3.3 Emitting rays

All rays that are traced in the computer model formulated in this effort begin at the optical sensor and are traced through a virtual screen down to the surfaces below. The virtual screen, shown in Figure 2.15, is used to create an image after all the rays have been traced [Turk, 1994].

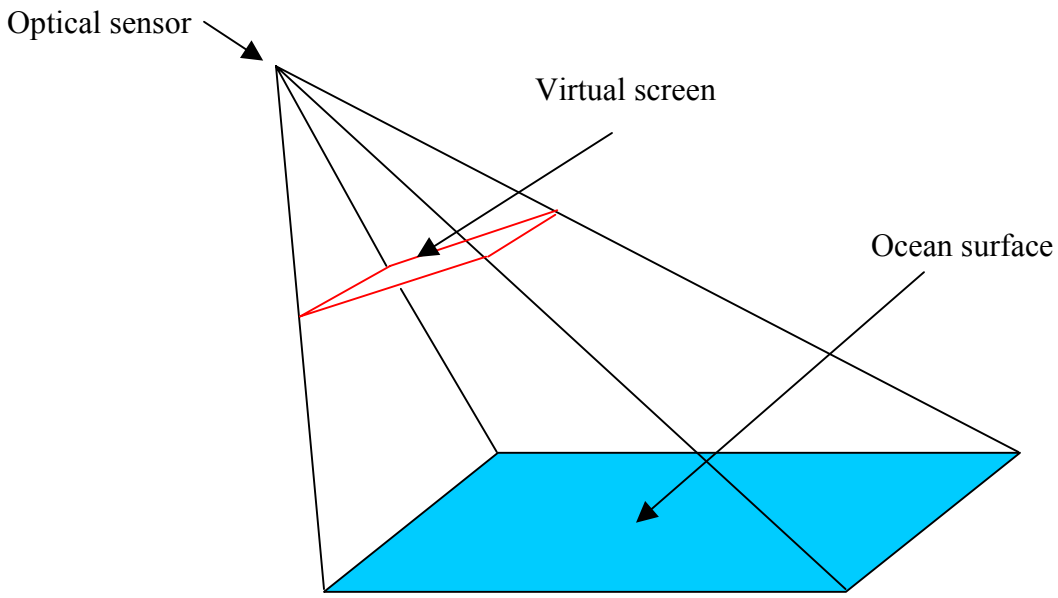


Figure 2.15 Optical sensor and screen location relative to the ocean surface

The rays are emitted using a raster scan method based on equally spaced along both dimensions of the screen. The screen is oriented in such a way that the ray passing through the center of the ocean is normal to the screen. Each of the facets on the virtual screen can be thought of as pixels on a real screen. All of the facets on the virtual screen are square and equal in size, as shown in Figure 2.16. The actual virtual screen used in the program uses a grid of 100 by 100 facets. Each screen pixel will ultimately be assigned a value based on the final power of the rays that pass through it.

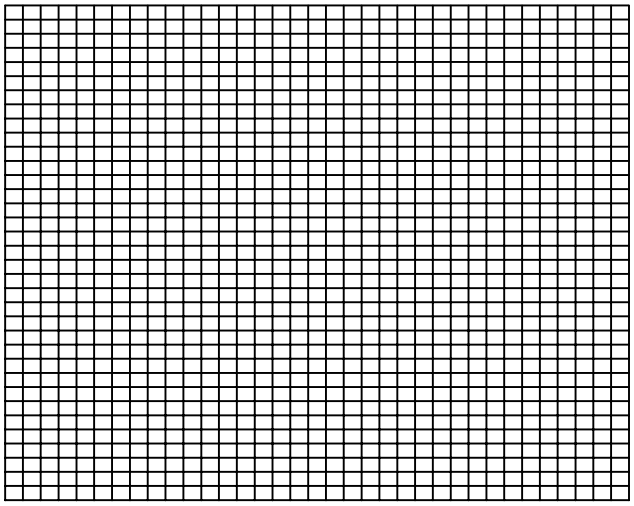


Figure 2.16 Facets that make up the virtual screen

2.3.4 Locating the point of intersection

Each time a ray is emitted or reflected, every facet in the scene must be checked to determine if the ray will intersect it. The first step in locating the point of intersection is to solve simultaneous equations for the line that represents the ray and the plane containing the facet [Mahan, 2002]. The equation for each plane is determined using

$$\vec{N} \cdot \vec{P} + d = 0 \tag{2.14}$$

where $\vec{N}$ is the normal vector of the plane, $\vec{P}$ is a vector locating any point on the plane, and d is the distance from the plane to the origin of the coordinate system. The equation for the ray is represented using

$$\vec{R}_p = \vec{R}_o + \vec{R}_d \, T \tag{2.15}$$

where $\vec{R}_o$ is a vector locating the point of origin of the ray, $\vec{R}_d$ is a vector defining the direction of the ray, $\vec{R}_p$ is a vector locating a point on the ray, and T is the distance from

the point located by $\vec{R}_o$ to the point located by $\vec{R}_p$. By using Equations 2.14 and 2.15, an equation for the distance T can be found as

$$T = \frac{d + \vec{N} \cdot \vec{R}_o}{\vec{N} \cdot \vec{R}_d} \tag{2.16}$$

Using this value for T and Equation 2.15, the point of intersection, located by $\vec{R}_p$, can be determined. Once this point is found, a check must be performed to determine if the point is located on the facet. Using a method presented by Baudouel [1990], a point on the plane is defined using two variables, α and β, as shown in Figure 2.17.

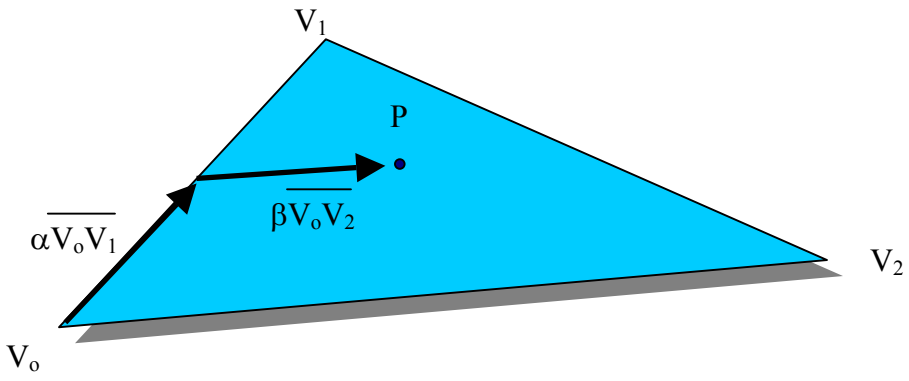


Figure 2.17 Locating a point on a facet

The values of α and β can be obtained for using

$$x_p - x_o = \alpha(x_1 - x_o) + \beta(x_2 - x_o) \tag{2.17}$$

$$y_p - y_o = \alpha(y_1 - y_o) + \beta(y_2 - y_o) \tag{2.18}$$

$$z_p - z_o = \alpha(z_1 - z_o) + \beta(z_2 - z_o) \tag{2.19}$$

Before solving for these variables the dominant plane must be determined to eliminate one of the equations. The dominant plane, and therefore the equation to be eliminated, is chosen according to Table 2.2.

Table 2.2 Rules for Reducing in the number of equations

Maximum ($\|N_x\|$, $\|N_y\|$, $\|N_z\|$)	**Dominant Plane**	**Equation to eliminate**
$\|N_x\|$	YZ	Equation 2.17
$\|N_y\|$	XZ	Equation 2.18
$\|N_z\|$	XY	Equation 2.19

Once the two remaining equations are solved, then in order for the intersection point to be located on the facet, it must be true that

$$\text{a) } \alpha \geq 0 \qquad \text{b) } \beta \geq 0 \qquad \text{c) } \alpha + \beta \leq 1 \tag{2.20}$$

If the ray intersects more than one facet, the correct facet is the forward candidate that is nearest to the point of origin. In Figure 2.18, the correct intersection point is point 2. Point 1 is closest, but it is in the negative direction and is therefore a back candidate. Points 3 and 4 are both further away than point 2.

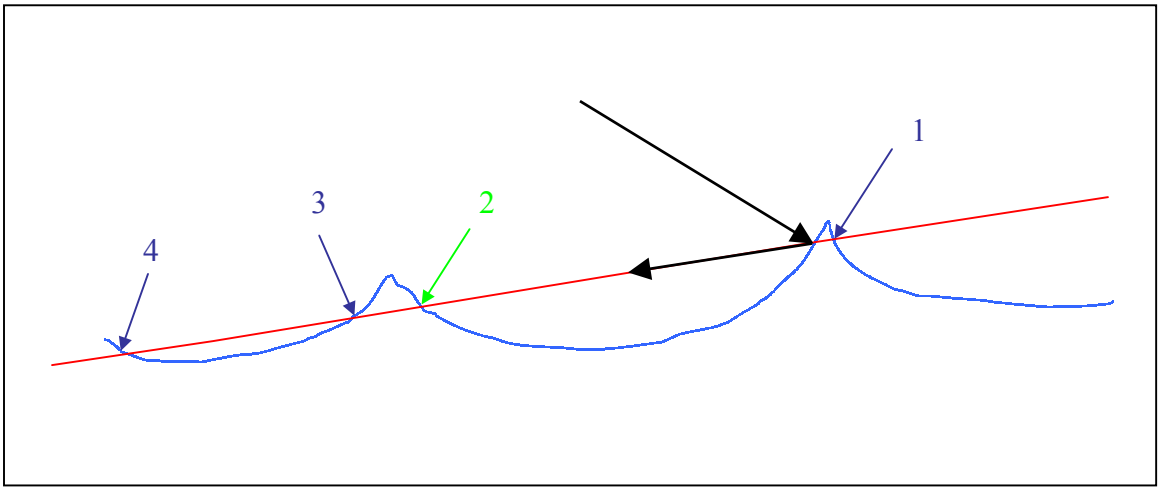


Figure 2.18 Finding the correct intersection point

If the ray does not intersect any facets, then it must intersect the sky, in which case the ray trace is terminated for that particular ray.

2.3.5 Determining the direction of reflection

The direction of reflection of any ray is dependant on the properties of the surface from which it is reflected. Two types of reflection are possible in the computer model: specular and diffuse-specular. The direction of the reflected ray for a specular reflection is found using

$$\vec{V}_r = \vec{V}_i + 2\,(\vec{V}_i \cdot \vec{N})\,\vec{N} \tag{2.21}$$

where $\vec{V}_r$ is a vector representing the direction of the reflected ray, $\vec{V}_i$ is a vector representing the direction of the incident ray, and $\vec{N}$ is the normal vector [Mahan, 2002]. For a diffuse-specular reflection, the type of reflection, diffuse or specular, must first be determined before the direction of reflection can be found. A uniformly distributed random number between zero and unity is drawn and its value is compared to that of the specularity ratio. If the random number is less than the specularity ratio, the reflection is specular; otherwise it is diffuse. The direction of reflection is then found using Equation 2.21 as described earlier. If the random number is greater than or equal to the specularity ratio, the reflection is diffuse, in which case the direction of diffuse radiation is found using

$$\phi = 2\pi R_\phi \tag{2.22}$$

$$\theta = \sin^{-1}(\sqrt{R_\theta}) \tag{2.23}$$

where R_ϕ and R_θ are uniformly distributed random numbers between zero and unity, θ is the angle between the reflected ray and the unit normal vector, and ϕ is the rotational angle around the normal vector relative to a tangent vector [Sparrow and Cess, 1966]. These two angles are shown in Figure 2.19.

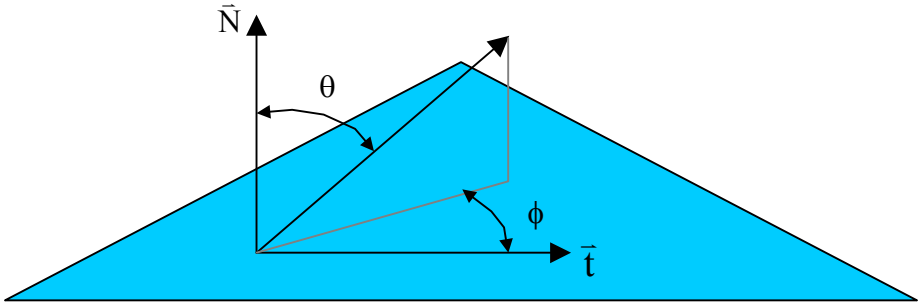


Figure 2.19 Direction of a diffuse reflection relative to unit normal and tangent vectors

2.3.6 Distribution factors

The distribution factor D_{ijk} is defined as the fraction of energy within wavelength interval $\Delta\lambda_k$ that is emitted from facet *i* and absorbed by facet *j*. This section treats the two methods used to estimate the distribution factors in the computer model. Once the distribution factors are known, the distribution of flux that passes through the virtual screen can be calculated. An image can then be created using this flux distribution. The details of the process used to determine the distribution factors are found in Mahan [2002].

2.3.6.1 Line-of-sight method

When using the line-of-sight method, all rays will continue to reflect until they reach the sky, but the rays leave behind energy with each reflection. As they rays are reflected, the power that the ray carries is reduced in both polarizations by an amount that is proportional to the absorptivity. The original energy of each ray is directly

proportional to the intensity of the sky where it intersects with the sky "surface". The distribution factor for the line-of-sight method is estimated as

$$D^{k}_{i,Sky} \cong \frac{P^{k}_{i,Sky}}{P^{k}_{i}} \tag{2.24}$$

where $P^{k}_{i,Sky}$ is the total power within wavelength interval $\Delta\lambda_k$ emitted by the sensor that passes through pixel *i* of the virtual screen and reaches the sky, and P^{k}_{i} is the total power within wavelength interval $\Delta\lambda_k$ that is emitted from the sensor that passes through pixel *i* of the virtual screen. Since this was a reverse ray trace and the rays are, in reality, emitted from the sky rather than the sensor, the rest of this section therefore refers to rays being emitted from the sky. The average relative power of the rays that reach each pixel on the screen is calculated using

$$P^{k}_{Sky,i} = \varepsilon^{k}_{Sky} A_{Sky} I_{Sky} D^{k}_{Sky,i} \tag{2.25}$$

where ε^{k}_{Sky} is the emissivity of the sky within wavelength interval $\Delta\lambda_k$, A_{Sky} is the area of the sky, A_i is the area of facet *i*, I_{Sky} adjusts all the rays to account for the difference in initial power, and $D^{k}_{Sky,i}$ is the reciprocal of $D^{k}_{i,Sky}$, which is calculated using the reciprocity relation

$$D^{k}_{Sky,i} = D^{k}_{i,Sky} \frac{A_i \varepsilon^{k}_{i}}{A_{Sky} \varepsilon^{k}_{Sky}} \tag{2.26}$$

By combining Equations 2.25 and 2.26, and knowing that ε^{k}_{i} is equal to unity, the relative power at pixel *i* can be simply written

$$P^{k}_{Sky,i} = A_i I_{Sky} D^{k}_{i,Sky} \tag{2.27}$$

Since the signal strength at each pixel is measured as a power flux, Equation 2.27 must be converted to

$$P''^{k}_{Sky,i} = I_{Sky} D^{k}_{i,Sky} \tag{2.28}$$

2.3.6.2 Traditional method

The traditional MCRT method uses energy bundles carrying equal power that do not lose power when they are reflected. In order to keep track of polarization, the power in each polarization was first reduced by an amount proportional to the absorptivity. The power in each polarization was then multiplied by a factor that would restore the total power to its original value. In this case the distribution factor may be estimated

$$D_{ijk} \cong \frac{N_{ijk}}{N_{ik}} \tag{2.29}$$

where N_{ijk} is the total number of rays emitted within wavelength interval $\Delta\lambda_k$ that pass through pixel i on the virtual screen and are absorbed by facet j in the scene, and N_{ik} is the total number of rays emitted within wavelength interval $\Delta\lambda_k$ that pass through pixel i. The average relative power flux within wavelength interval $\Delta\lambda_k$ that passes through pixel i on the screen is calculated using

$$P''_{ik} = \sum_{j=1}^{n} \varepsilon_{jk} \frac{A_j}{A_i} D_{jik} e_{B,k}(T) \tag{2.30}$$

Using a similar equation to the one used in Equation 2.26, Equation 2.30 can be simplified to

$$P''_{ik} = \sum_{j=1}^{n} D_{ijk} e_{B,k}(T_j) \tag{2.31}$$

where $e_{B,k}$ is the blackbody function integrated over wavelength interval $\Delta\lambda_k$, T_j is the temperature of facet *j* in the scene, and *n* is the total number of facets in the scene.

2.3.6.3 Displaying results

Each pixel on the screen is assigned two values that are proportional to the average relative power flux, or signal strength, that passes through that pixel. One of these values represents the horizontally polarized light signal strength, and the other represents the vertically polarized light signal strength. The results can then be displayed by specifying a height, a color, or both, proportional to the signal strength. Figure 2.20 shows an example where both height and color are both used.

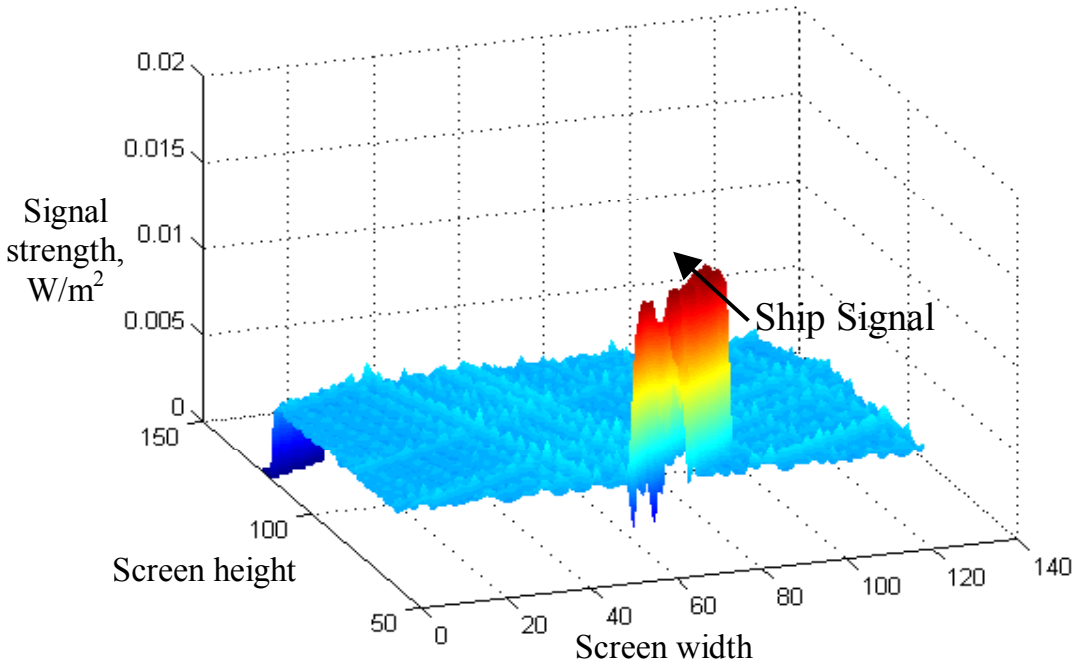


Figure 2.20 Optical signature using height and color to show signal strength in terms of W/m^2

2.4 Summary

This chapter presents all the theory used to create the computer model and display its results. The method for creating all of the geometry is covered in the first section. This includes the statistics for the ocean surface, the method for creating the ocean surface, and the method for creating the ship that was used in the model. The next section introduces polarization theory, and the final section covers all aspects of both types of ray trace used. The difference between the two ray-trace models is explained, followed by the means for treating each surface in the model. An explanation is given of how the rays are traced; including ray emission, determining points of intersection, and finding the direction of a reflected ray. Finally, estimation of the distribution factors and their use to form an image is developed.

CHAPTER 3 Program manuals

This chapter provides a detailed description of how the program is organized and how to use it. This is done through the use of a users' manual and a programmers' manual. Each of these manuals treats all four of the programs making up the computer model. The first program, called **ocean**, is used to create the geometry for the ocean surface. The program **ship** is used to create the ship geometry at a specified location and orientation with respect to the ocean. The program **MCRT** traces a user-specified number of rays and stores all relevant information about each ray. The final program, **post,** does two things. First, all of the information stored from the ray trace is adjusted to account for initial conditions. Each pixel on the screen is then assigned a value based on these results. A listing of all four programs may be found in the appendix.

3.1 Users' manual

The users' manual provides a detailed description of how each of the programs is used and how the programs are used together to model an array of cases. Table 3.1 shows all the parameters that can be changed by the user for each program. The parameters that are highlighted in orange apply only to the visible-light ray trace, and those in red apply only to the infrared ray trace.

Table 3.1 – A listing of user specified parameters for each program

Ocean	Ship	MCRT	Post
Air-Sea ΔT	Rotation	Ray trace type	Number of pixels
Wind speed	X-offset	Ship reflectivity	Sun azimuth
Number of facets	Y-offset	Ship specularity	Sun zenith
Size of facets		Sensor azimuth	Ship temperature
		Sensor zenith	Ocean temperature
		Number of rays	

The programs are separated in such a way that not all of the programs need be re-executed each time a parameter is changed. For example, the user can change the location of the sun while only re-executing the **post** program, or the user can change the location of the sensor with out re-executing the **ocean** and **ship** programs.

3.1.1 Program ocean

This program is capable of creating an infinite number of different ocean surfaces, but only one was studied for the effort discussed in this thesis. All four of the user-specified parameters were held constant for each ray trace that was performed. The first two parameters control the shape of the waves and the last two control the size and number of waves. When the program is executed, the command "Enter the air-sea temperature difference" is displayed. This value is the result of subtracting the ocean surface temperature from the air temperature where all temperatures are in degrees celsius. Once this value is entered, the computer prompts the user to enter the wind speed in meters per second. The user is then prompted to enter the size of the facets, in meters, that make up the ocean surface. All the facets are square, and so only one dimension needs to be entered. The last parameter entered is the number of facets in each direction. The same number is used for both directions, but it is important to note that the patch of ocean surface of interest is not necessarily square. Because the facets are oriented at different angles, the dimension of the ocean surface in each direction is dependant on the shape of the waves.

3.1.2 Program ship

This program allows the user to create a given ship geometry at any location and orientation on the ocean surface. The first parameter to input is the angle of orientation in degrees. An angle of zero would position the ship in the positive x-direction, and a positive angle will rotate the ship counter-clockwise, as shown in Figure 3.1.

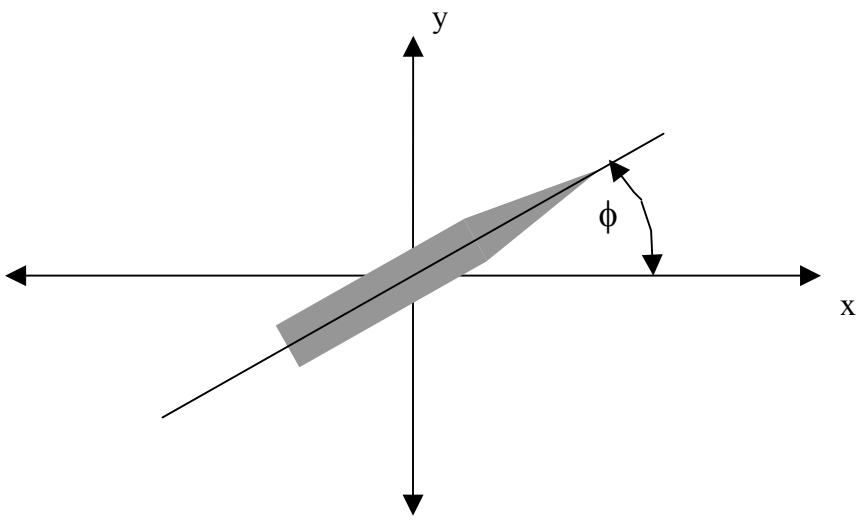


Figure 3.1 Rotation convention used for generating ship geometry

The last two parameters control the location of the ship using an x-offset and a y-offset. A value of zero for both of these parameters positions the center of mass of the ship at the origin.

3.1.3 Program mcrt

The first step when running this program is to specify the type of ray trace, where "1" indicates that the ray trace is for visible light reflected from the sky, and "2" indicates that the ray trace is for infrared emission from all surfaces. The second step is to enter the surface properties of the ship by specifying the reflectivity and the specularity ratio that will be applied to all surfaces of the ship. The next information entered is the location of the optical sensor. The straight-line distance from the center of the ocean surface to the sensor is set to a constant value of 5000 m within the program. Therefore, the only information that is needed is the viewing angle, θ, and orientation angle, ϕ, of the sensor with respect to the center of the ocean, as shown in Figure 3.2.

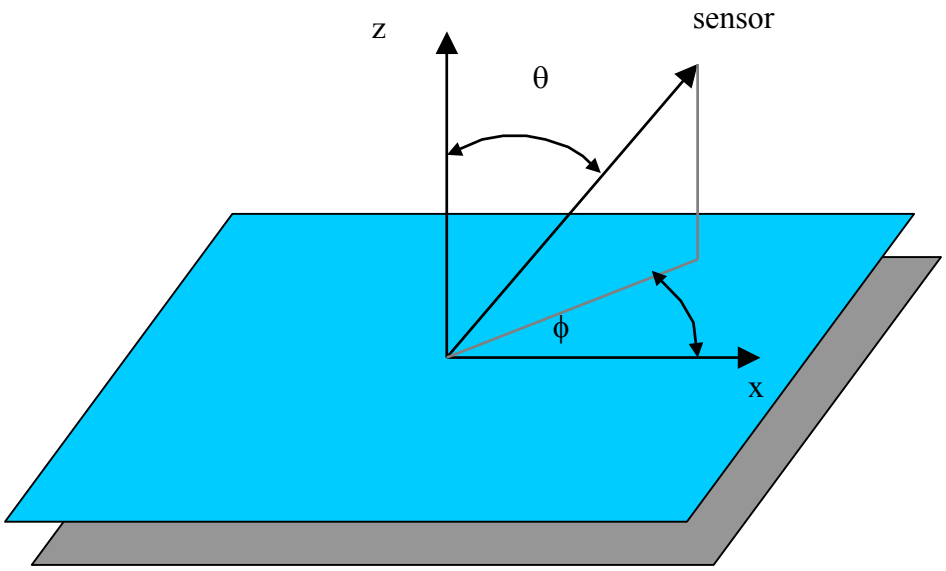


Figure 3.2 Convention for locating the optical sensor

The final input into the program is the number of energy bundles, or rays, to be traced. Since the program was written so that the rays are always emitted in a square array, i.e., as a raster scan, the program only needs the number of rays to be emitted in a horizontal band along the screen.

3.1.4 Program post

The information needed to execute this program varies depending on the type of ray trace used. For both types of ray trace, the number of pixels on the screen must be specified. This is done by entering the number of pixels along one direction of the screen; the program will automatically apply this number to the other direction. For the visible-light ray trace, the location of the sun must be entered so that the distribution of the sky intensity can be determined. The sun location is entered in the same way that the optical sensor location was entered in the previous section. The infrared ray trace requires the surface temperature of the ship and the water, both of which must be specified in kelvins.

3.2 Programmers' manual

This section gives an overview of how each program is organized. Each subsection describes, step-by-step, program operation and the order in which it performs each task. The method for linking the programs together is also described in this section.

3.2.1 Program ocean

The purpose of the program described in this section is to generate the ocean surface used in the overall MCRT model. The ocean surface is created using an array of rectangular facets. Once the ocean is completed this program stores all the information about each facet in two text files, **Ocean_Height** and **Ocean_Properties**. The commented code for this program can be found in Appendix A.

The first part of this program allows the user to specify information that affects the ocean surface being generated. Using this information the standard deviation, skewness, and kurtosis of both PDF's, crosswind and along-wind, are determined. These functions are then numerically integrated to give a function similar to the one shown in Figure 2.3. A series of waves is generated in each direction using the corresponding integrated PDF. Three nested loops are used to create each of these wave patterns. The outside loop is over the number of waves, the middle loop is over the number of segments per wave, and the inner loop is over the number of facets per segment. The number of segments per wave is held constant for both directions and for all conditions, but the number of facets per segment randomly varies between two and four to give variety to the size of the waves. Each facet along the wave pattern is of equal length to all other facets and is assigned a tilt angle based on the integrated PDF. The process for determining these angles is discussed in detail in Section 2.1.2. Once an angle has been assigned to a facet, the vertical and horizontal changes in location can be calculated using

$$\Delta H = L\cos\theta \tag{3.1}$$

$$\Delta V = L\sin\theta \tag{3.2}$$

where ΔH is the change in horizontal location, ΔV is the change in vertical location, L is the length of the facet, and θ is the angle assigned to the facet. The beginning of each facet starts at the end of the previous facet. A close-up view of a wave being assembled is shown in Figure 3.3.

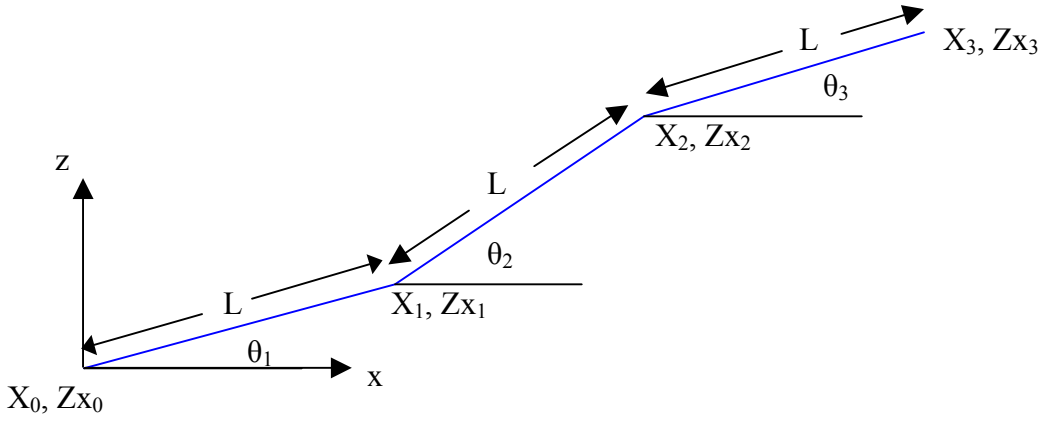


Figure 3.3 Two-dimensional wave assembly in the along-wind direction

The vertical location of every point that makes up the ocean wave patterns is recorded in a text file called **Ocean_Height**. After both wave patterns are recorded, they are brought together to make a three-dimensional surface, as illustrated in Figure 2.4. Figure 3.4 shows a typical facet of the ocean surface, where X_k is less than X_m, and Y_1 is less than Y_n. The normal vector for a facet is determined from a cross product of $\vec{V}_x$ and $\vec{V}_y$.

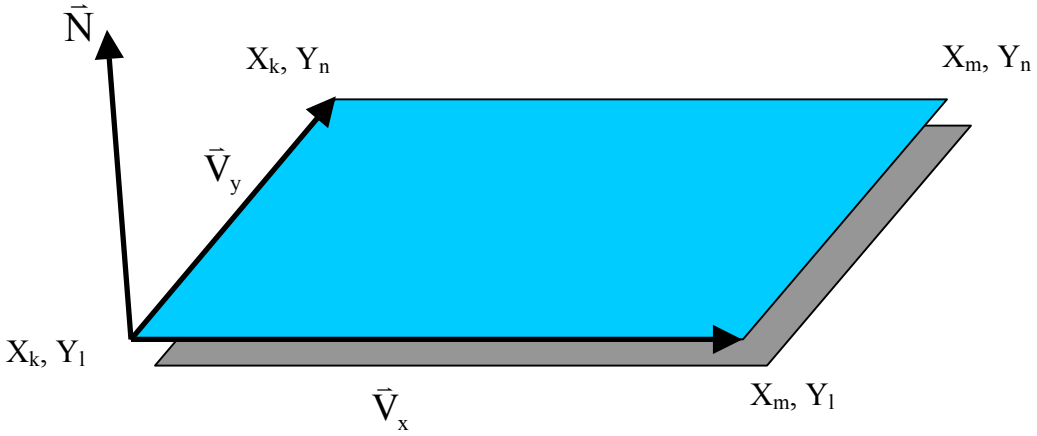


Figure 3.4 Typical rectangular facet with its normal vector

Given the location and direction of the normal vector, an equation for the plane containing the facet can be determined using

$$Ax + By + Cz = D \tag{3.3}$$

where A is the x-component of the normal vector, B is the y-component of the normal vector, C is the z-component of the normal vector, D is the distance between the plane and the origin of the coordinate system and x, y, z are the coordinates for the origin of the normal vector. The quantity D is the only unknown and can therefore be calculated using Equation 3.3. The quantities A, B, C, and D are then stored in a text file called **Ocean_properties**, along with the number of facets and the maximum and minimum values of the ocean surface heights. The information stored in this text file is critical for running the Monte Carlo ray trace described in Section 3.2.3.

3.2.2 Program ship

The program for establishing the ship geometry was written to allow the ship to be placed anywhere on the ocean surface and oriented any direction. The geometry is stored in a text file as a grouping of triangular facets. The program **mcrt** uses this text file to locate the ship during operation. The commented code for this program can be found in Appendix B.

The first part of the ship generated is the main deck. The hull is then formed by extending the main deck down at an 80-deg angle on both sides of the deck, as shown in Figure 3.5. The topsides are extended downward sufficiently far to ensure that its lower edge is completely below the ocean surface. All the points that make up the section above the main deck of the ship are then added to the existing geometry, as shown in Figure 3.6.

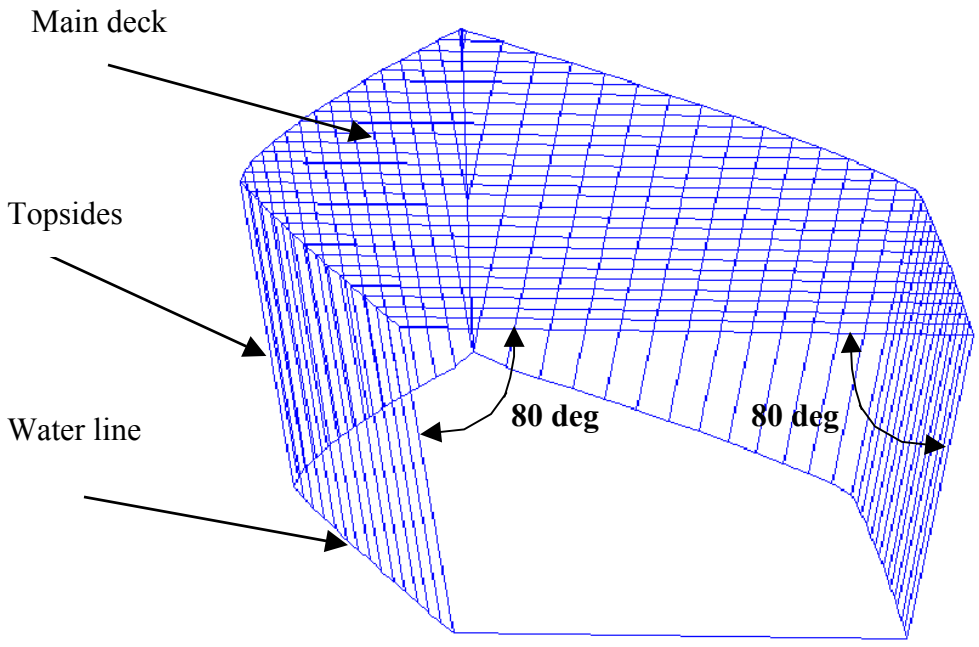


Figure 3.5 Main deck and topsides of the destroyer

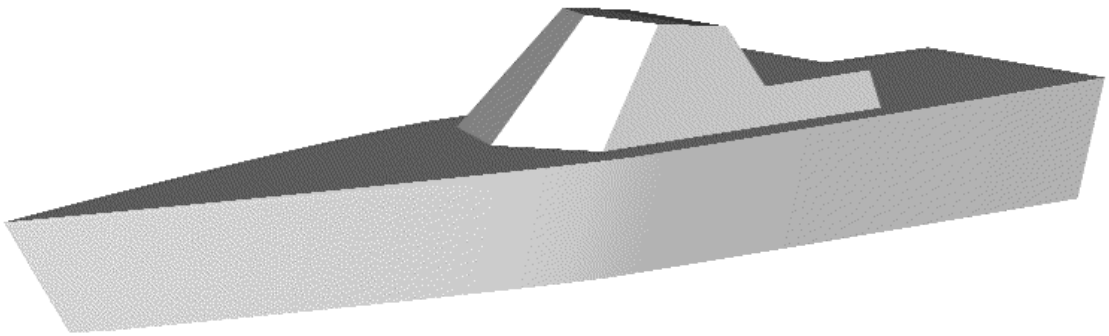


Figure 3.6 Rendered image of the completed ship

At this point the user is prompted to enter any desired rotation or offset. The rotation modifies all x and y coordinates using the transformation

$$x_{NEW} = x_{OLD}\cos\phi - y_{OLD}\sin\phi \tag{3.4}$$

$$y_{NEW} = y_{OLD}\cos\phi + x_{OLD}\sin\phi \tag{3.5}$$

where ϕ is the desired angle of rotation. The translation modifies all x and y coordinates using the transformation

$$x_{NEW} = x_{OLD} + x_{OFFSET} \tag{3.6}$$

$$y_{NEW} = y_{OLD} + y_{OFFSET} \tag{3.7}$$

Now that the coordinates to all the points describing the ship are known, triangular facets can be used to mesh the visible surface of the ship. The equations needed to represent the facets must next be determined. This is done using a method similar to the one used for the ocean surface. Figure 3.7 shows a typical facet on the ship's surface.

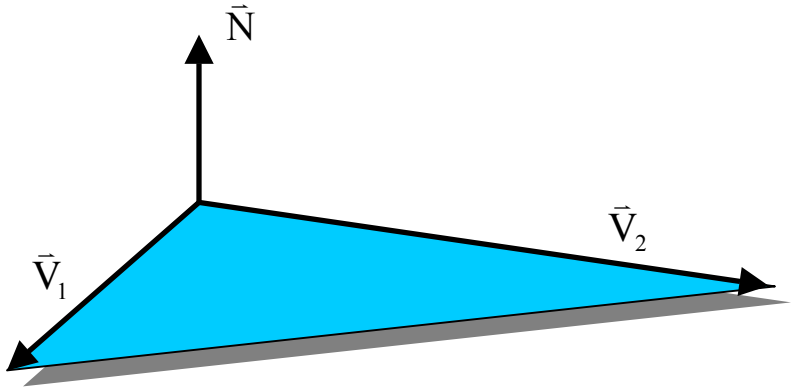


Figure 3.7 Typical triangular ship facet with its normal vector

The location of the normal vector is not important for this type of facet. The only criterion that must be met is that the normal vector point away from the ship. The

coefficients A, B, C are the components of the normal vector as described in Section 3.2.1 and D is found using Equation 3.3. These values, along with the vectors $\vec{V}_1$ and $\vec{V}_2$, are stored in the text file called **Ship_Facets**.

3.2.3 Program mcrt

This is the main program around which the other programs are designed. A specified number of energy bundles is emitted and traced to the collection of facets stored in the three text files discussed earlier. After each energy bundle is traced, all relevant information about that energy bundle, shown in Table 3.2, is stored in a new text file used by the final program, **post**. The commented code for this program can be found in Appendix C.

Table 3.2 Information stored after a ray trace

Visible light ray trace	Infrared ray trace
Point of intersection with the screen	Point of intersection with the screen
Direction of the ray after the last reflection	Direction of the ray after the last reflection
Power of the ray after absorption	Location where the ray was absorbed
	Power of the ray after absorption

3.2.3.1 Emitting energy bundles

The first step for the **mcrt** program is to read in all the information created and stored by the previous two programs, **ocean** and **ship**. Once this is complete, all of the user-specified information must be entered. This information is then used to fix the location of the optical sensor and the center of the virtual screen. During the ray trace these locations are used to determine the starting point and direction of each ray. Figure 3.8 illustrates how the rays are distributed on the screen. The increments in horizontal and vertical angles between the rays are all equal. This even distribution is accomplished by using rotational transformations on the ray marking the center of the screen, $\vec{C}$ (shown red in Figure 3.8).

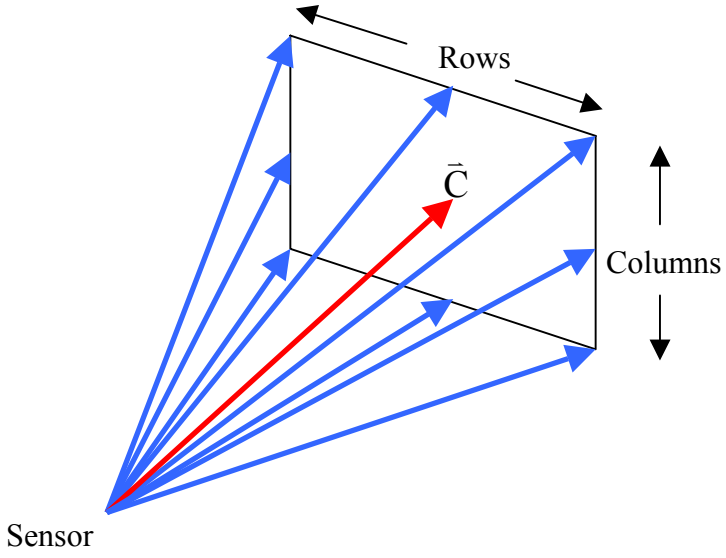


Figure 3.8 Ray distribution and location on the screen

To move the direction of emission between the different rows, the center ray must rotate about the vector

$$\vec{V} = \vec{C} \times \vec{Z}_{AXIS} \tag{3.8}$$

Once the ray is rotated vertically to the correct row, the ray can then be rotated horizontally to the desired column. This requires an axis of rotation that varies with each row. The axis of rotation for each row is computed from

$$\vec{H}_i = \vec{C}_i \times \vec{V} \tag{3.9}$$

where $\vec{C}_i$ is the center ray for row *i*, and $\vec{H}_i$ is the horizontal axis of rotation for row *i*. The transformation matrix required to rotate about any axis is

$$[R_{uvw}] = \begin{bmatrix} c\theta + u^2(1-c\theta) & -ws\theta + uv(1-c\theta) & vs\theta + uw(1-c\theta) \\ ws\theta + uv(1-c\theta) & c\theta + v^2(1-c\theta) & -us\theta + vw(1-c\theta) \\ -vs\theta + uw(1-c\theta) & us\theta + vw(1-c\theta) & c\theta + w^2(1-c\theta) \end{bmatrix} \quad (3.10)$$

where c and s represent cosine and sine respectively, θ is the angle of rotation, and (u, v, w) is the vector notation for the axis of rotation in global coordinates [http://www.celos.psu.edu/kinematics/Kin_tutor06.html].

3.2.3.2 Locating the point of intersection

Each time a ray is emitted or reflected all facets must be visited to determine which facets, if any, the ray intersects. The facets that constitute the ocean surface are visited first. Since the ocean consists of approximately 40,000 facets, visiting each facet individually is not a reasonable task, considering that millions of rays typically need to be traced. It was therefore deemed necessary to find a way to eliminate a large percentage of these facets without checking them directly. The method used involves temporarily replacing the ocean with a much simpler geometry, as illustrated in Figure 3.9.

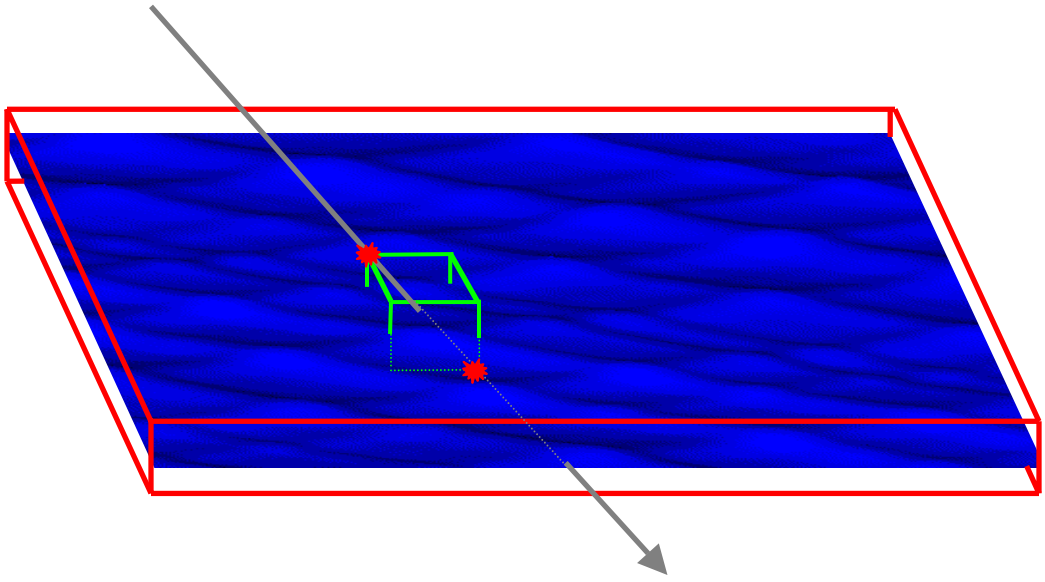


Figure 3.9 Illustration of a scheme for reducing the number of facets

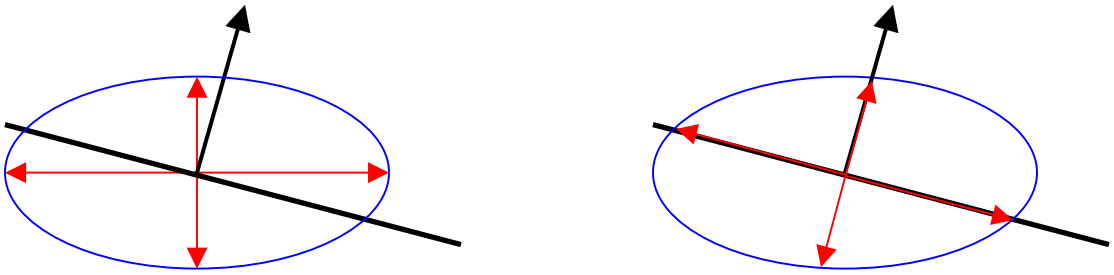
The x and y coordinates of the points of intersection with the red box represent the maximum and minimum possible values for the x and y coordinates of the point of intersection with the ocean. This means that the range of possible facets on the ocean can be reduced to those that are within the green box. Each facet within the green box and each facet on the ship are checked for intersection points. Each time an intersection point is found the distance to the point of emission, T, calculated using Equation 2.16, is compared to the value of T for the previous point of intersection. Once all the facets are checked, the point with the smallest positive value for T will be the true point of intersection. If no points of intersection are found, the direction of the ray, along with the current power and the point of intersection with the screen, is stored in a text file called **Rays**.

3.2.3.3 Reflecting rays

If a point of intersection is found, the ray could either be absorbed or reflected with a loss of power. For the visible-light ray trace the only possibility is reflection with a loss of power. The infrared ray trace, however, permits either reflection of the ray or complete absorption of the ray.

3.2.3.3.1 Visible-light ray trace

Since any intersection of a ray with a facet results in a reflection for the case of visible light, the only decision that must be made is for the direction of the reflected ray. If the point of intersection lies on an ocean facet the reflection must be specular, but if the point of intersection lies on a ship facet the reflection can either be specular or diffuse depending on the value of the specularity ratio. In addition to the direction, the reflected power for both polarizations must be calculated to account for the reflection. It is first necessary to properly align the power vectors with the facet. The original orientation of the two electric field components is vertical and horizontal with respect to the global coordinate system. However, the magnitude of the perpendicular and parallel electric field components with respect to the facet is required to perform the reflection correctly. Figure 3.10 shows the orientation of the electric field components, represented by power vectors, before and after rotation.



a) b)

Figure 3.10 Rotating the power vectors

Each power vector can then be multiplied by the reflectivity, corresponding to its respective polarization component, to obtain the correct remaining power. The program then returns to Section 3.2.3.2 to find the next point of intersection.

3.2.3.3.2 Infrared ray trace

For this type of ray trace the program must first determine if the ray will be reflected or absorbed. This was done by choosing a uniformly distributed random number and comparing it to the total reflectivity of the facet in question. If the value of the random number is less than or equal to the reflectivity, the ray is reflected and the same procedure outlined in Section 3.2.3.3.1 is followed. Otherwise, the ray is absorbed and all relevant information about the ray is stored in the text file **Rays**. This information will include the intersection point with the screen, the location of adsorption, and the final power of the ray.

3.2.4 Program post

The purpose of this post-processing program is to convert all the information generated as the result of a ray trace into distribution factors, regardless of the type of ray trace performed. The commented code for this program can be found in Appendix D.

The screen that all the rays pass through during the ray trace is divided into the number of pixels specified by the user, all equal in size. This will result in some pixels having more rays than others, but the results are only concerned with the average of those rays so the number of rays per facet is not critical. Since the ray traces that are performed are reverse ray traces, the program must now apply the initial conditions to account for variations in the initial power represented by each ray. For the visible light ray trace the initial power of each ray is determined by finding the relative intensity of the sky at the location from which it came. The initial power of a ray used for the infrared ray trace is found by integrating the blackbody distribution function, as described in Section 2.3.5.2. After the power is adjusted, each ray is associated with the pixel that it passed through during the ray trace. This allows the program to calculate the average relative power of the rays in any given pixel. This value is referred to earlier as the signal strength of a pixel. When the entire process is finished, two values for each pixel remain: one for horizontal polarization and one for vertical polarization. These values are stored in text files to be read in by a program, such as MatLab, capable of making three-dimensional plots and color-scaled plots.

3.3 Summary

This chapter covers the use of each program and its organization. The first section in this chapter covers the users' manual, which is responsible for ensuring that the reader can reproduce the results or create new results. The next section in this chapter is a programmers' manual for each of the four programs used. This manual, along with a documented program, allows the programmer to understand the organization of the program for the purpose of adding or changing lines of code.

CHAPTER 4 Results

This chapter presents the results of a series of test cases. The first section presents the variations studied using visible light, while the second section presents the infrared results. These results are then summarized and discussed.

4.1 Visible light ray traces

Before any test cases could be run it was necessary to perform a convergence study to determine the minimum number of energy bundles to be traced. Once this was completed, four different parameters were varied to determine their effect. First, the reflectivity of the ship was varied to determine the optimum value to hide the ship from a sensor not having a polarizing filter. Next, the specularity ratio was varied to determine its optimum value to hide the ship. Once these values were determined, the viewing angle of the optical sensor and the location of the sun were varied to determine the effect this has on the detectability of a ship. Polarizing filters were also used to determine the importance of the polarization of the ocean. Table 4.1 shows the nominal case along with all the variations that are considered in this section.

Table 4.1 Variations of parameters used for visible light ray trace

Parameters	Minimum	Nominal	Maximum
Reflectivity of the ship	0.1	0.15	0.2
Specularity Ratio of the ship	0.1	0.5	0.9
Viewing angle, θ	10, 30	50	70
Azimuth Angle of the sun, ϕ	0	90	180

4.1.1 Convergence study for visible radiation

A convergence study was performed to determine the minimum number of energy bundles that need to be traced. Holding all other factors constant, the number of energy bundles was increased until the image produced converged. Convergence was verified by computing the global change in the relative power of each pixel on the virtual screen using

$$\Delta P = \sqrt{\sum_{j=1}^{j_{max}} \sum_{i=1}^{i_{max}} \left(P_{ij}^{new} - P_{ij}^{old} \right)^2} \qquad (4.1)$$

where P^{new} is the new relative power after additional energy bundles are traced, P^{old} is the previous relative power, i_{max} is the number of pixels in the horizontal direction on the screen, j_{max} is the number of pixels in the vertical direction on the screen, and ΔP is the change in relative power. Eight different cases were run for this convergence study. The number of energy bundles traced for each case is given in Table 4.2.

Table 4.2 Number of energy bundles traced for convergence study

Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8
250,000	500,000	1,000,000	2,000,000	3,000,000	4,000,000	5,000,000	6,000,000

The results of this study are shown in Figure 4.1. After examining Figure 4.1, and taking into consideration the time required to compute each case, a value of 4,000,000 energy bundles was chosen for the visible light ray traces.

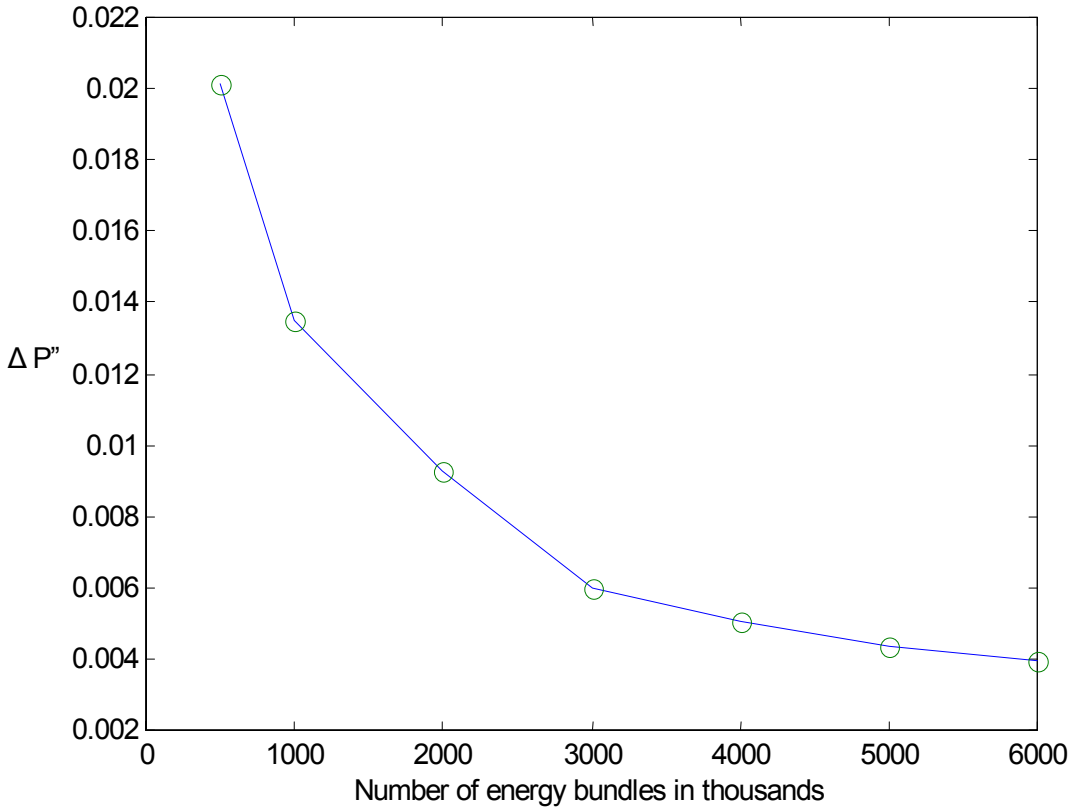


Figure 4.1 Convergence for visible light ray trace

4.1.2 Influence of variations in ship reflectivity in the visible

This first parameter that was varied was the reflectivity. The reflectivity was varied in steps of 0.05 and the best possible value to hide the ship was chosen from those cases studied. Figure 4.2 shows three examples of the images created corresponding to different values of reflectivity.

A visible reflectivity of 0.15 gives the best cover for the ship under the nominal conditions shown in Table 4.1. This result, however, could change drastically if the nominal conditions are changed. The power of the signal coming from the ship compared to the power of the signal from the ocean can best be seen in Figure 4.3, which shows cross-sections of the signal strength with labels indicating the location of the ship. A visible reflectivity of 0.15 produces a ship signal closest to that of the nominal ocean

signal and therefore will likely provide the best camouflage. There is, however, a shadow that is apparent in all these images that does not blend in as well as the rest of the ship.

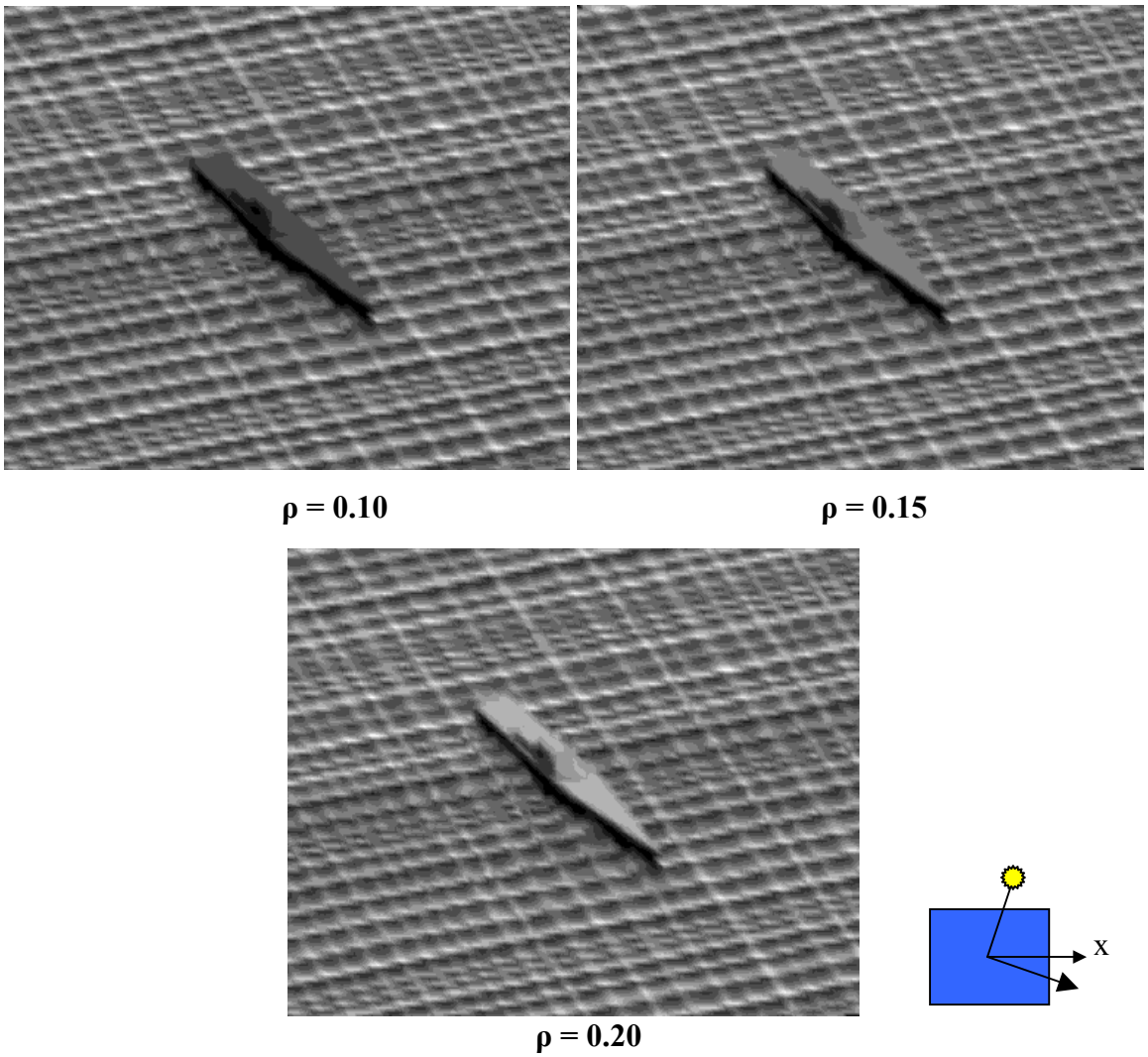


Figure 4.2 Influence on visible image from variations in reflectivity, ρ (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.5)

4.1.3 Influence of variations in ship specularity ratio in the visible

The second parameter varied was the visible specularity ratio, defined by Equation 2.11. This ratio was varied to find a value that would better hide the ship compared to the nominal specularity ratio. Figure 4.4 shows three examples of the images created when the visible specularity ratio was varied while holding the visible

reflectivity constant at ρ = 0.15. It is important to note that these images, along with all other images in this chapter, use a scaling that changes with each image. Very little is gained by changing the specularity ratio, but a slight reduction in the shadows on the side of the ship is obtained when this ratio is reduced. These shadows are still visible after all adjustments to surface properties and so it may be necessary to have separate properties on the nearly vertical panels of the ship to eliminate this problem. Therefore a value of $r^s = 0.1$ has been used in the following analyses.

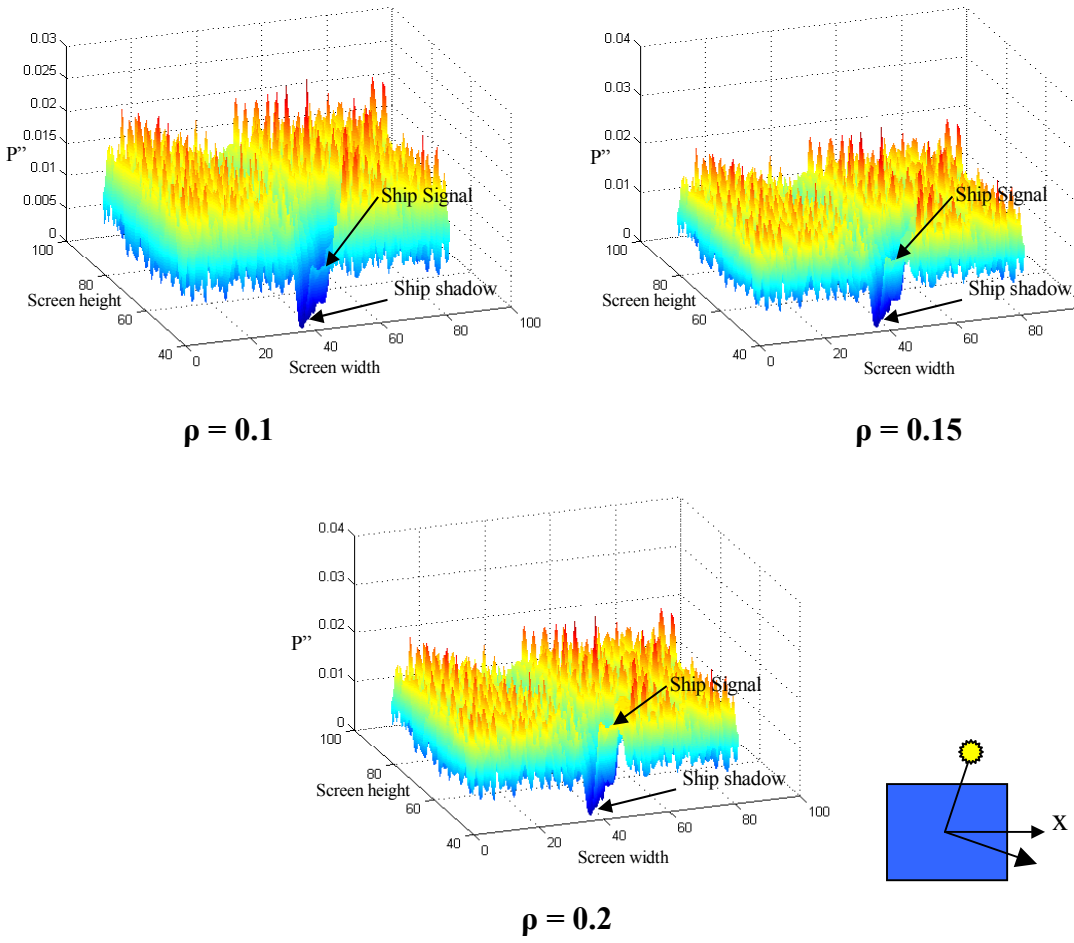
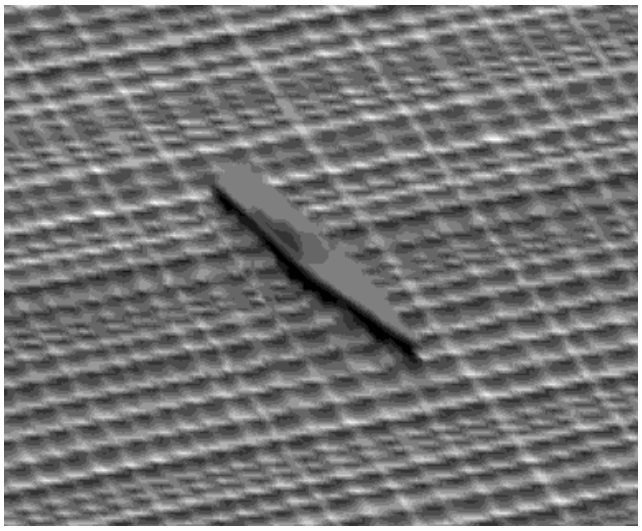
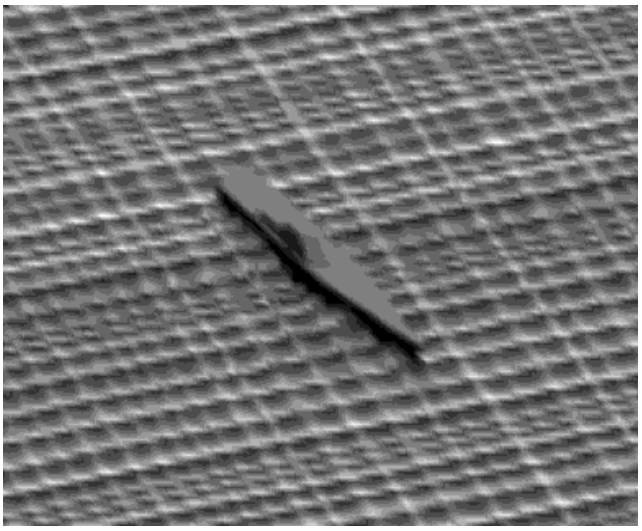


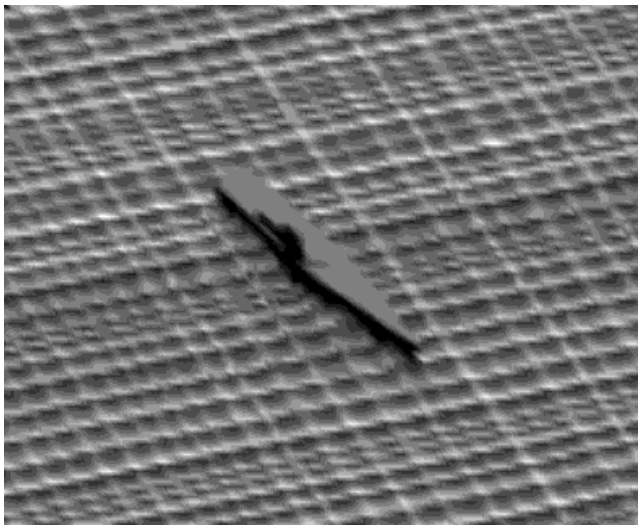
Figure 4.3 Influence on visible contrast from variations in reflectivity (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, $r^s = 0.5$)

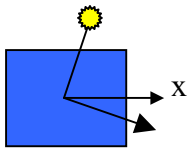


$r^s = 0.1$



$r^s = 0.5$





$r^s = 0.9$

Figure 4.4 Influence on visible image from variations in specularity ratio (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, ρ = 0.15)

4.1.4 Influence of variations in the viewing angle in the visible

Now that the nominal values of reflectivity and specularity ratio for the ship have been chosen, the other parameters can be varied. The viewing angle of the optical sensor

has a very large impact on the signal strength of the ocean and can therefore produce a large effect on the visibility of the ship. Figure 4.5 shows the image created corresponding to viewing angles of 70, 30, and 10 deg, along with the nominal angle of 50 deg. As the viewing angle is increased, the size of the screen must contract to keep all of the rays within the boundaries of the limited ocean surface. The result gives a zooming effect due to a higher concentration of rays in a smaller area, but, in reality, all of the ships are equally far away.

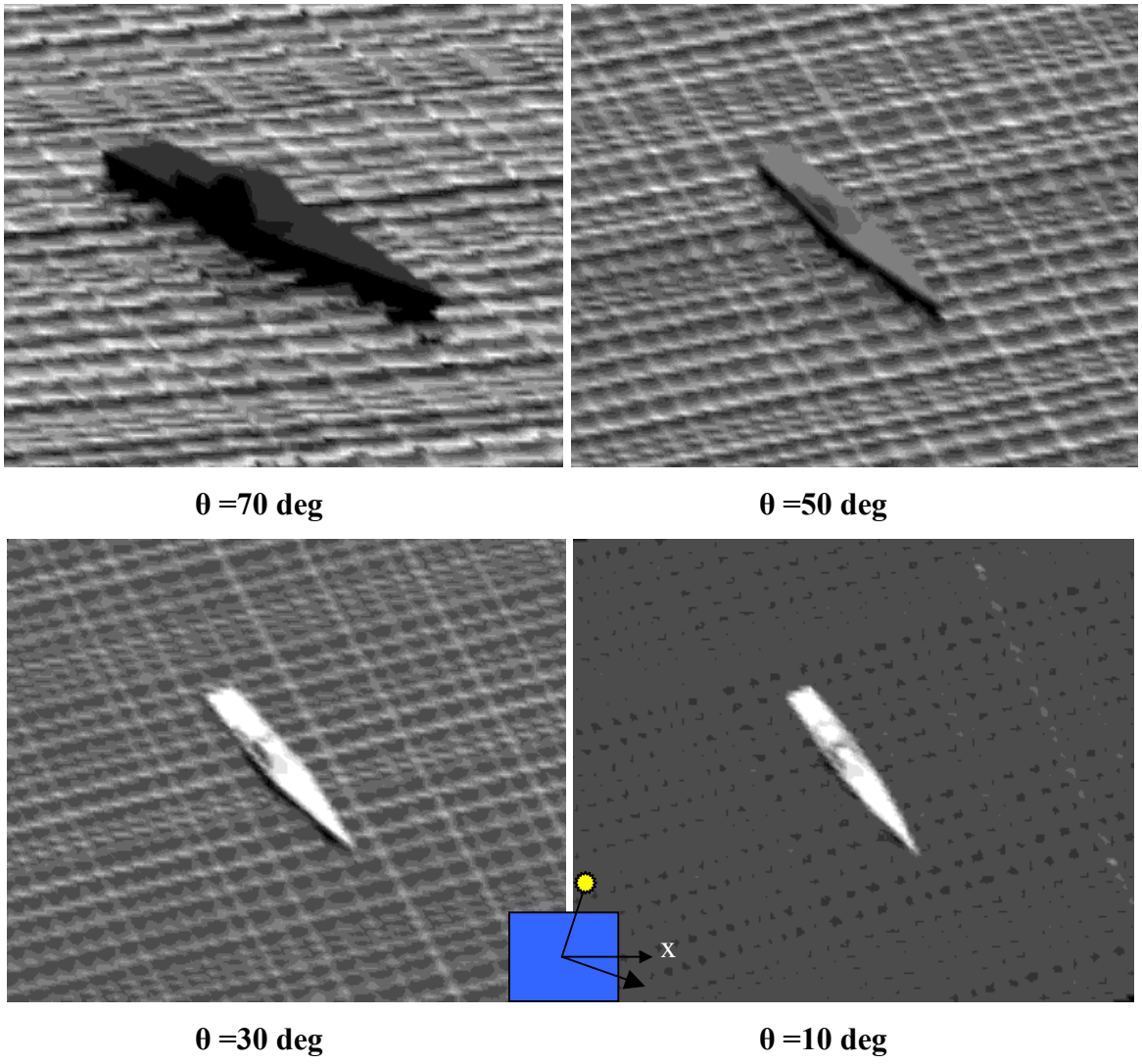


Figure 4.5 Influence on visible image from variations in viewing angle (ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.5, ρ = 0.15)

The contrast between the ship and the ocean is dramatically increased when the viewing angle is varied from the nominal value. This is due to a water reflectivity that varies with the incident angle. The power of the signal from the ship is nearly constant while the power of the signal from the water drops significantly when the zenith angle of the incident light is decreased. The optimum reflectivity for the ship is therefore dependant on the viewing angle of the optical sensor. Figure 4.6 shows more dramatically how different the signal can be by varying this angle.

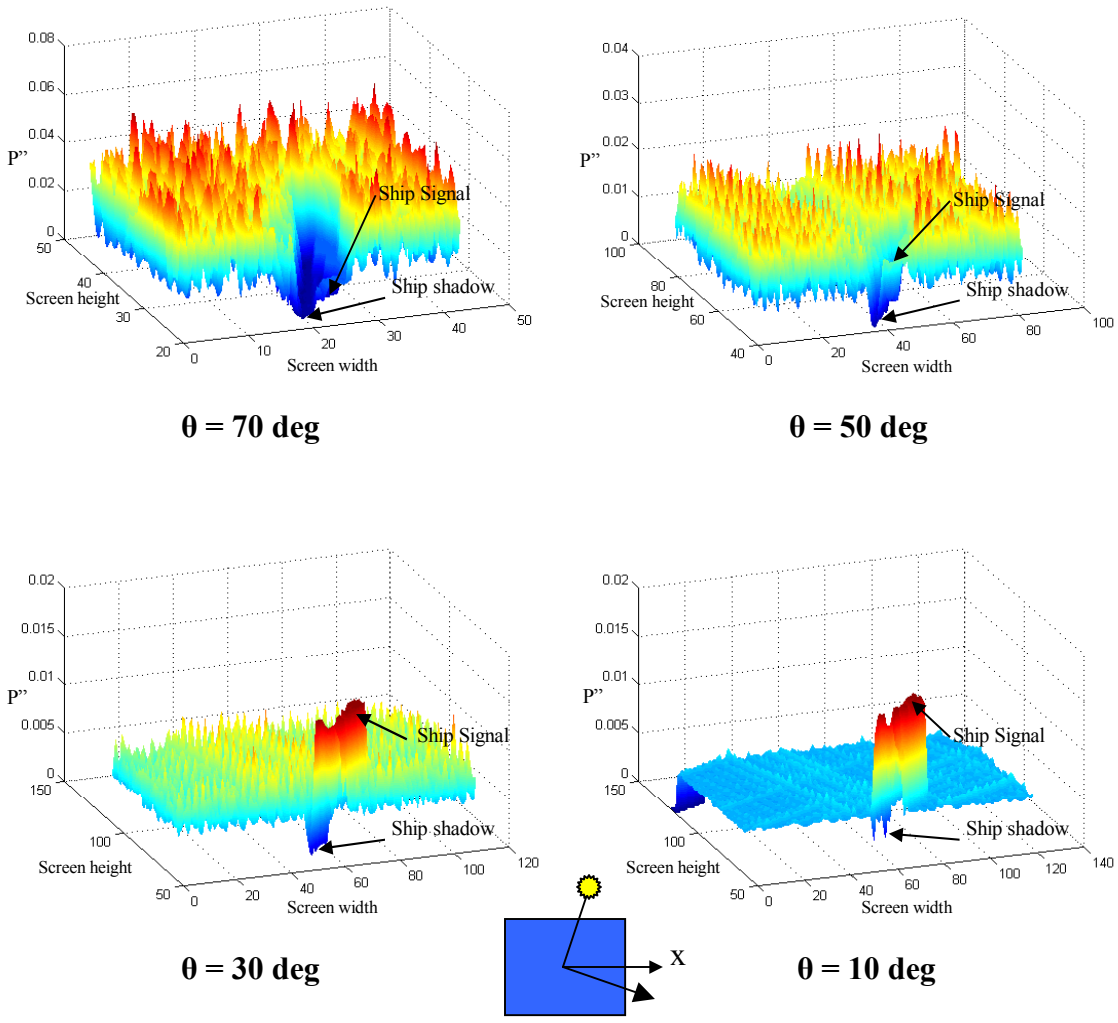
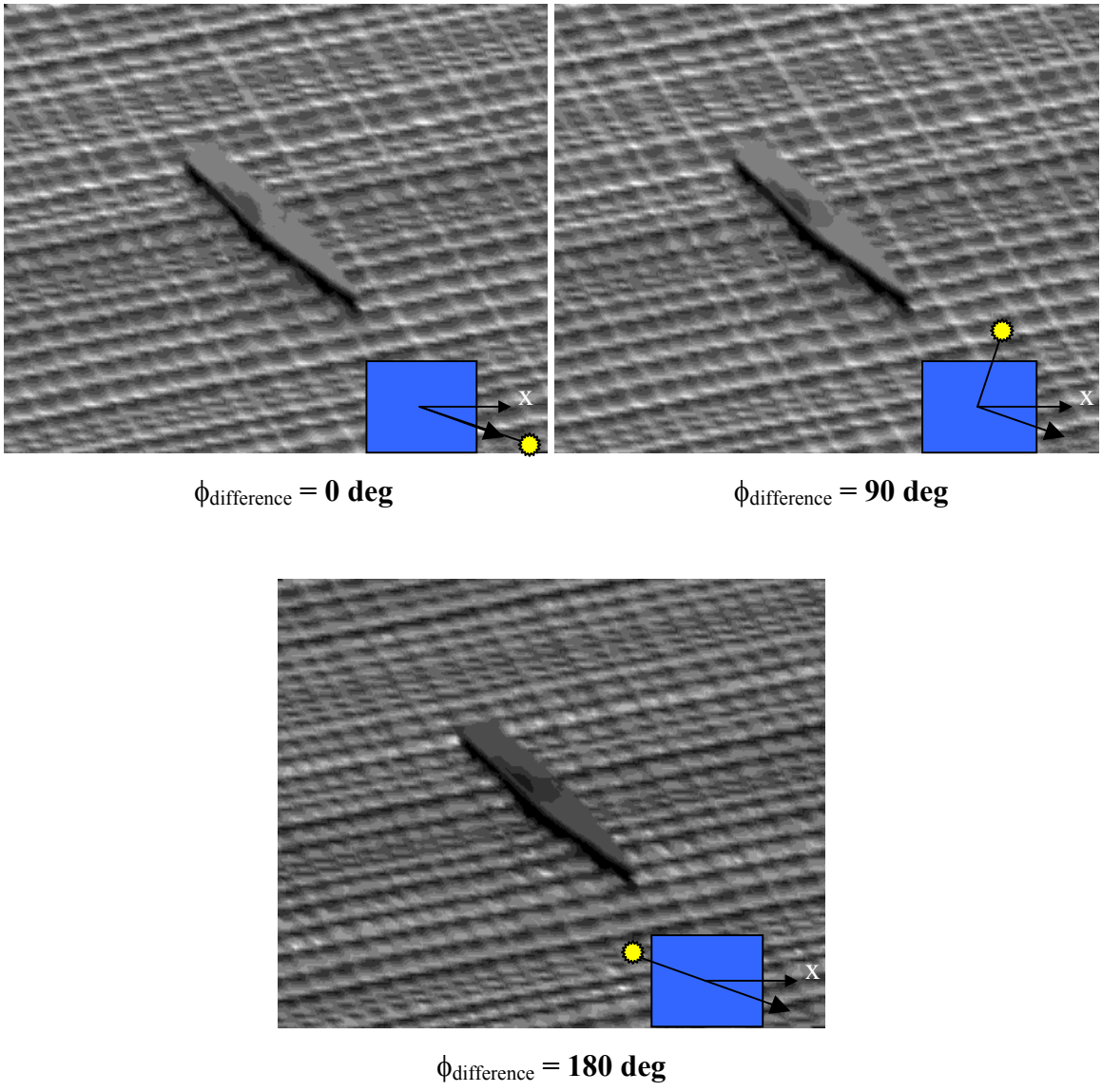


Figure 4.6 Influence on visible contrast from variations in viewing angle (ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.5, ρ = 0.15)

4.1.5 Influence of variations in the location of the sun

The location of the sun was varied in both the zenith and azimuth directions. The sun was first rotated about the vertical axis with the sensor as the reference point and the zenith angle held constant at the nominal value of $\theta = 50$ deg. The impact on the signal was small, but noticeable. Figure 4.7 shows the nominal case of a 90-deg difference along with 0-deg and 180-deg differences.



$\phi_{difference}$ **= 0 deg**

$\phi_{difference}$ **= 90 deg**

$\phi_{difference}$ **= 180 deg**

Figure 4.7 Influence on visible image due to variations in the solar azimuth angle ($\theta_{view} = 50$ deg, $\phi_{view} = -30$ deg, $\theta_{sun} = 50$ deg, $r^s = 0.5$, $\rho = 0.15$)

The two extreme cases are shown in Figure 4.8 in another format.

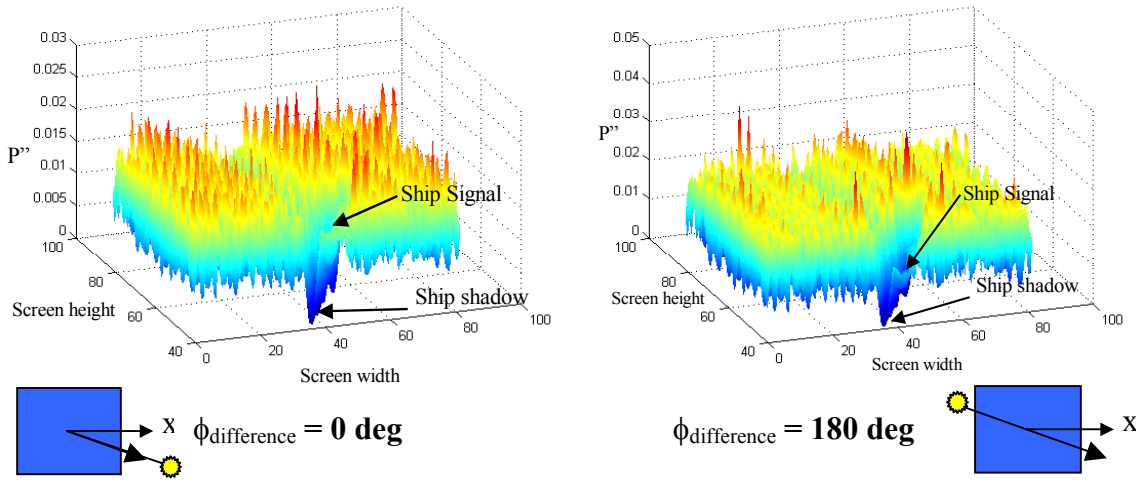
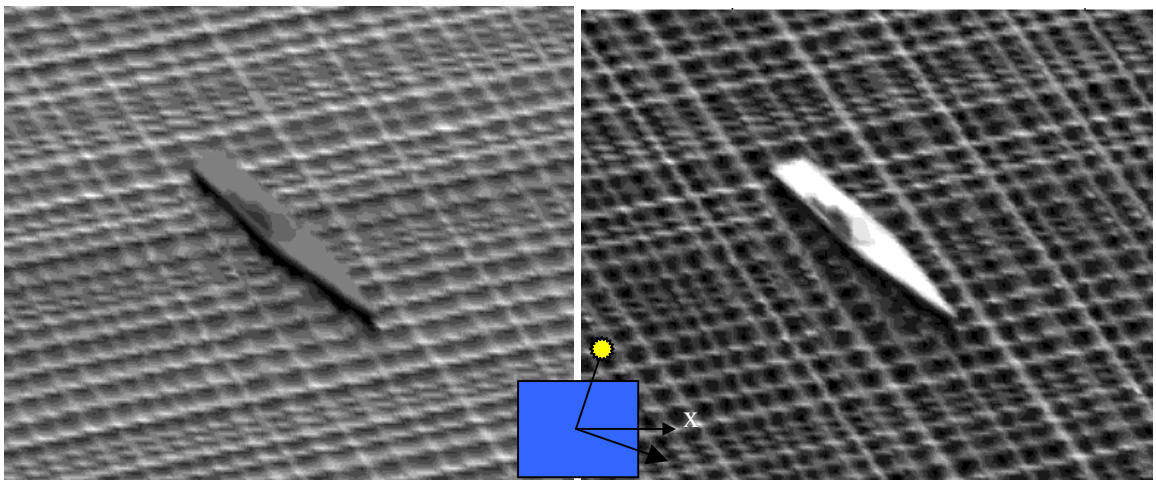


Figure 4.8 Influence on visible contrast due to variations in the solar azimuth angle (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, r^s= 0.5, ρ = 0.15)

The zenith angle of the sun was also varied, but the impact on the signal was even less than that of changing its azimuth angle. This is due to having only a small percentage of the energy bundles originating at the solar aureole, thus creating results similar to that of a uniform sky.

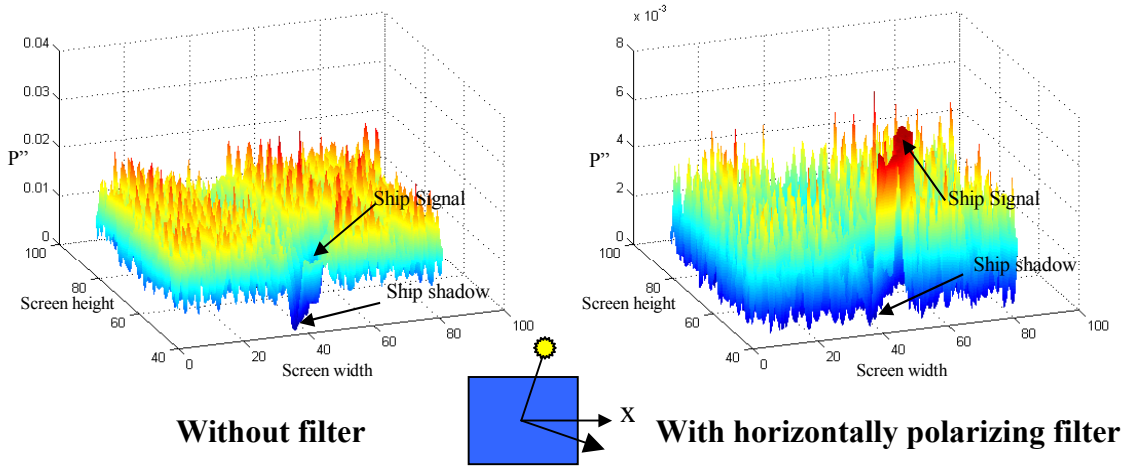
4.1.6 Influence of Polarization filters in the visible

A polarization filter can be used to increase the contrast between the ship signal and the ocean signal. A ship that would normally be relatively difficult to detect can become much more visible because of an increase in contrast between the ocean and the ship. Objects that do not polarize light, such as the ship, always have a reduction in signal strength of %50 when using a polarization filter. The ocean surface, however, polarizes light and has a reduction in signal strength that is dependent on the degree of polarization. Figure 4.9 shows the nominal case without a polarization filter on the left, and with a horizontal polarization filter on the right. The ship is clearly much easier to detect with a horizontally polarizing filter since the ocean signal strength is reduced by a larger amount than that of the ship. The difference in signal strength can also be seen in Figure 4.10.



Without filter **With horizontally polarizing filter**

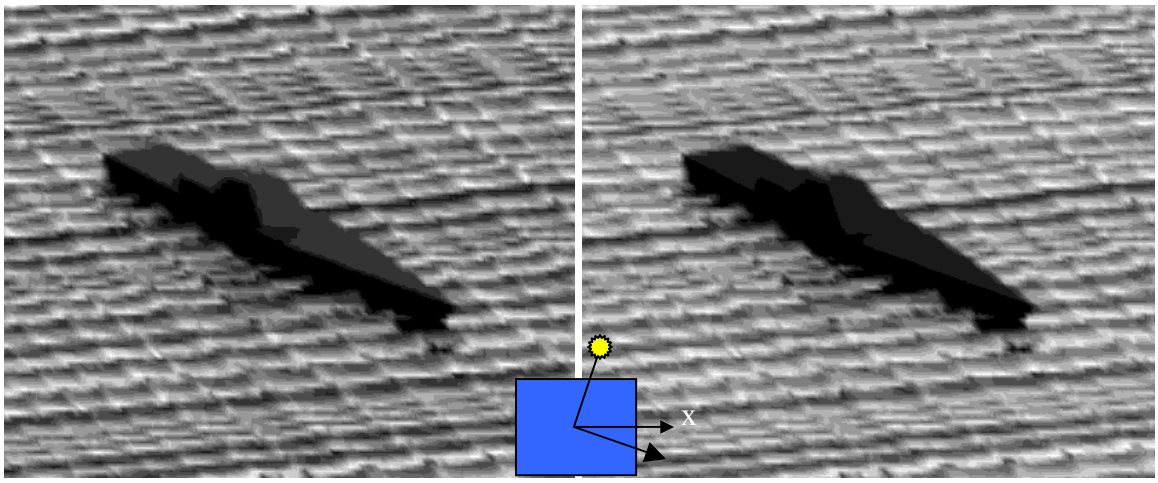
Figure 4.9 Influence on visible image from using a horizontally polarizing filter (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.1, ρ = 0.15)



Without filter **With horizontally polarizing filter**

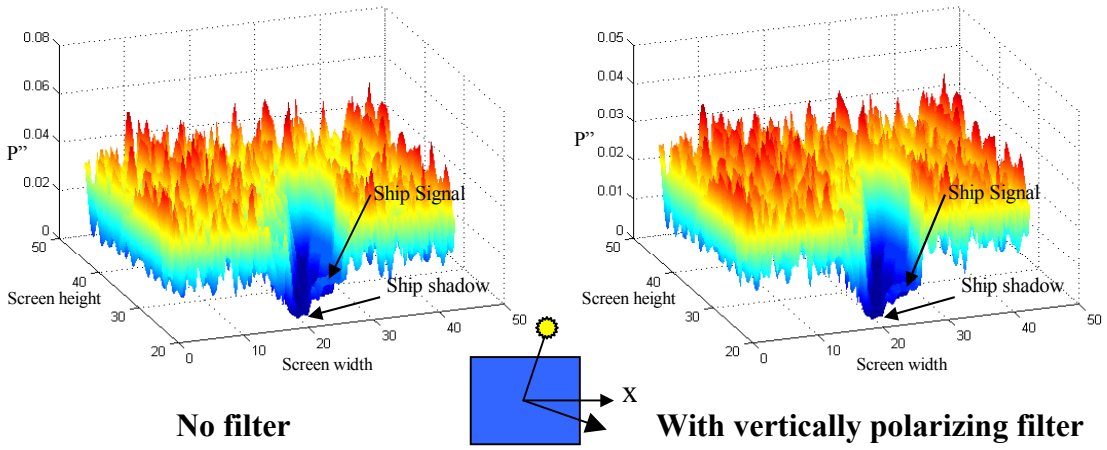
Figure 4.10 Influence on visible contrast from using a horizontally polarizing filter (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.1, ρ = 0.15)

If the signal produced by the ship is weaker than that produced by the water, use of a vertically polarizing filter makes the ship more visible. Figure 4.11 shows the ship viewed at a zenith angle of 70 deg with and without a vertically polarizing filter. The enhanced contrast between the ship and the water is best shown in Figure 4.12.



Without filter **With vertically polarizing filter**

Figure 4.11 Influence on visible image from using a vertically polarizing filter (θ_{view} = 70 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.1, ρ = 0.15)



No filter **With vertically polarizing filter**

Figure 4.12 Influence on visible contrast from using a vertically polarizing filter (θ_{view} = 70 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.1, ρ = 0.15)

4.1.7 Ship with water surface properties

In addition to looking at diffuse-specular surface properties for the ship, the case where the water and the ship have the same surface properties was considered. This was done to determine the maximum reduction in contrast that could be accomplished from using engineered surface properties. Figure 4.13 shows the image created at angles of 70,

30, and 10 deg along with the nominal case of 50 deg when the ship is given the same surface properties as the seawater.

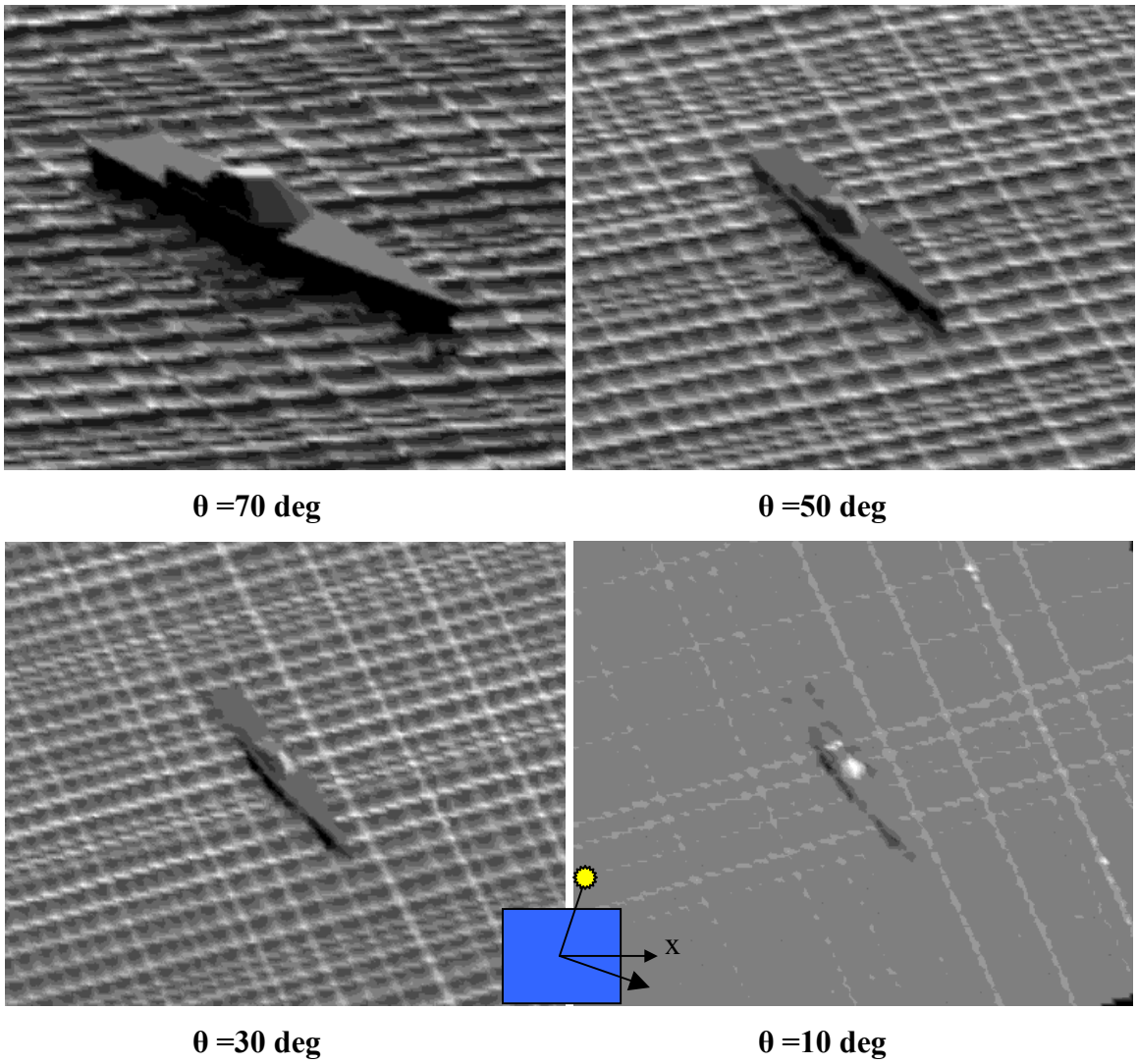
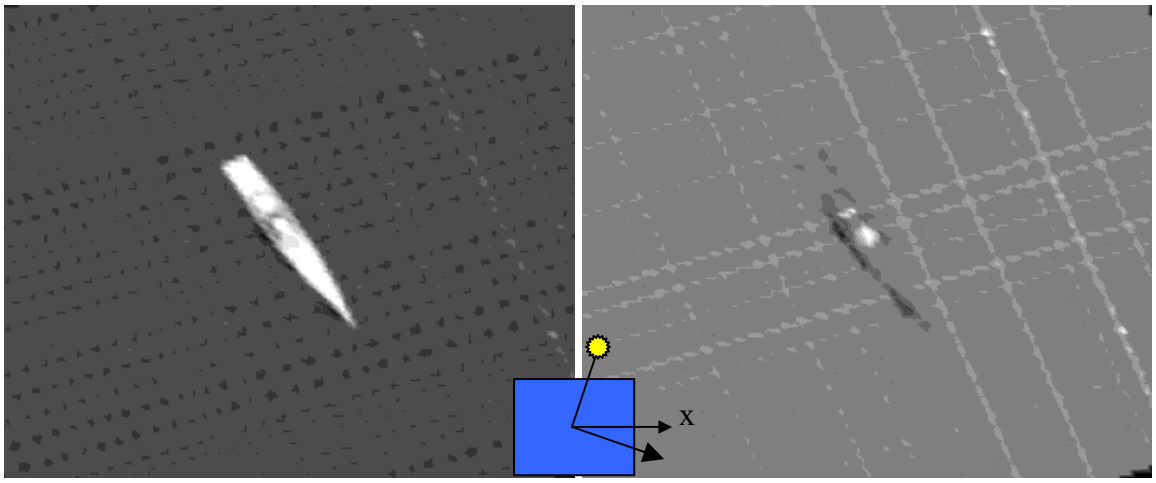


Figure 4.13 Influence on visible image from variations in viewing angle when the ship has the same optical properties as the water (ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg)

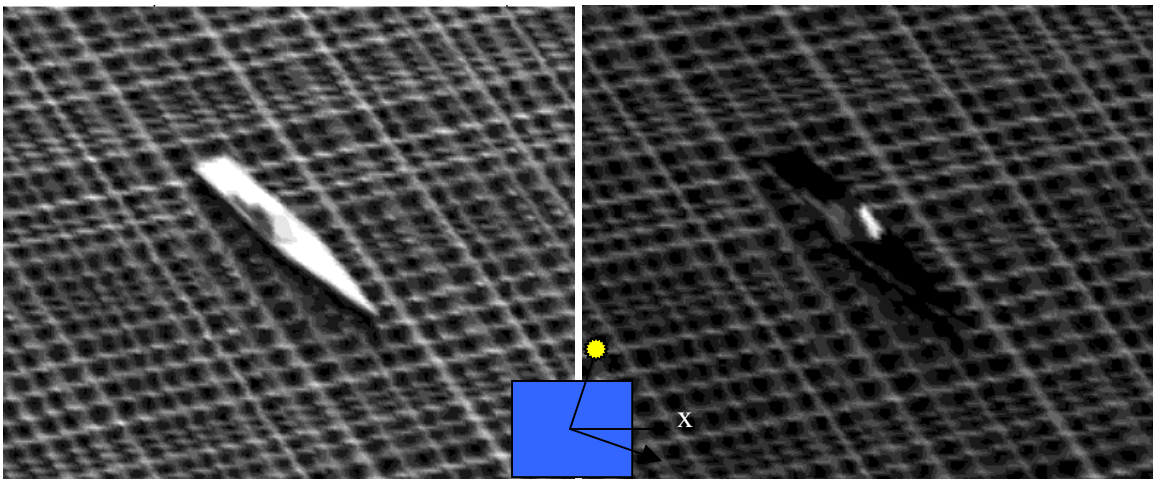
The contrast with the ocean background was greatly reduced, especially at angles much different than the nominal. The worst case that was shown for diffuse-specular was at a zenith angle of θ =10 deg. A comparison of the two surface properties at this angle is shown in Figure 4.14.



Diffuse-specular **Seawater properties**

Figure 4.14 Comparison of the visible contrast when the ship has diffuse-specular and seawater surface properties (θ_{view} = 10 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.1, ρ = 0.15)

Since the ship is now polarizing light in the same way as the ocean, the change in contrast using a polarization filter is significantly reduced, as shown in figure 4.15.



Diffuse-specular **Seawater properties**

Figure 4.15 The difference in contrast between a diffuse-specular ship and a ship with seawater surface properties while using a horizontally polarizing filter (θ_{view} = 50 deg, ϕ_{view} = -30 deg, θ_{sun} = 50 deg, ϕ_{sun} = 60 deg, r^s= 0.1, ρ = 0.15)

4.2 Infrared ray traces

The results for the infrared ray traces are organized in much the same way as are those for the visible ray traces described in the previous section. Since the main goal of this effort is to create a useful tool, not to complete a full study, the test cases discussed in this chapter assume that the temperature of the ocean and the ship are both 290 K and are not varied. The temperature of the sky is assumed to be 220 K, an established value for a clear night. A convergence study was performed in the same manner as for visible-light ray traces. Once this was completed, three different parameters were varied to determine their influence on ship visibility. First, the reflectivity of the ship was varied to determine the optimum value to reduce contrast without a polarization filter. Next, the specularity ratio was varied to determine its optimum value to reduce contrast. The viewing angle of the optical sensor was then varied to determine the effect this has on the contrast between the ship and the sea. Polarization filters were also used to determine the importance of polarization. Table 4.3 shows the nominal case along with the parameter variations considered in this section.

Table 4.3 Variations in parameters for the infrared ray trace

Parameters	Minimum	Nominal	Maximum
Emissivity	0.8	0.85	0.9
Specularity Ratio	0.1	0.5	0.9
Zenith Angle	10, 30	50	70

4.2.1 Convergence study for infrared radiation

A convergence study was carried out as discussed in Section 4.1.1. The results of this study are shown in Figure 4.13.

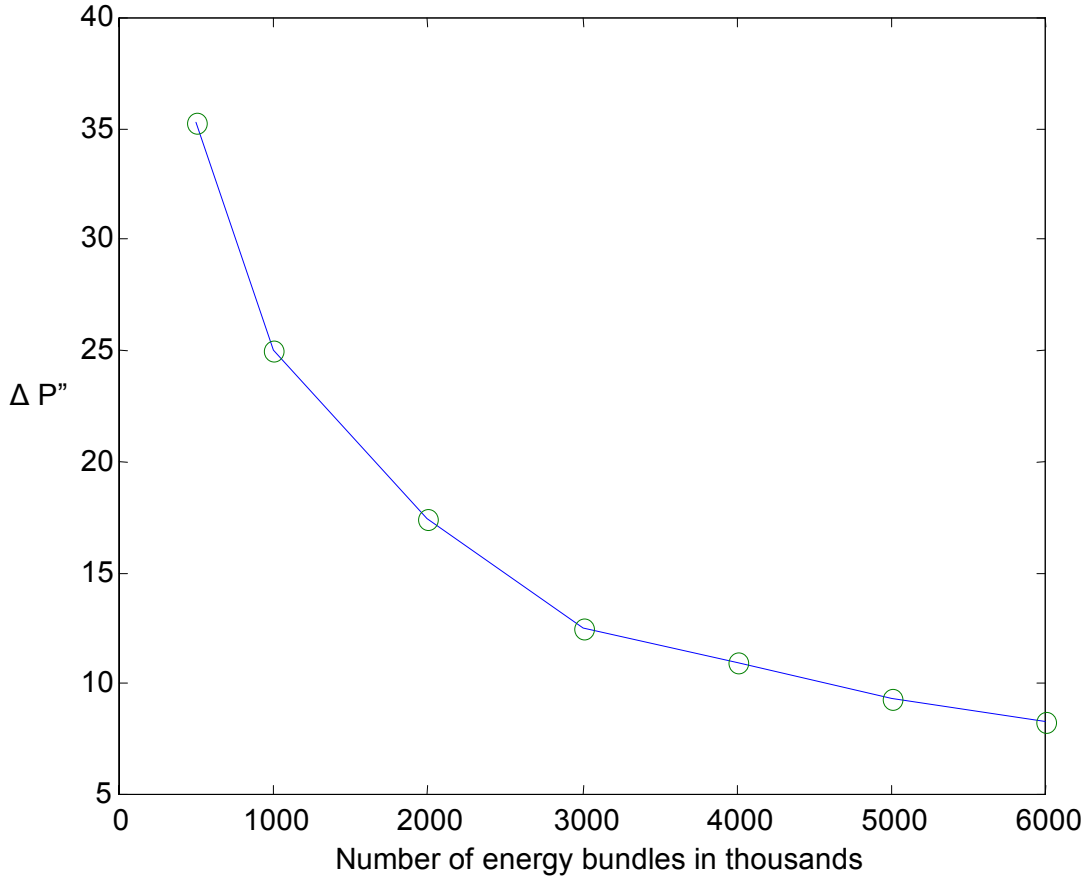
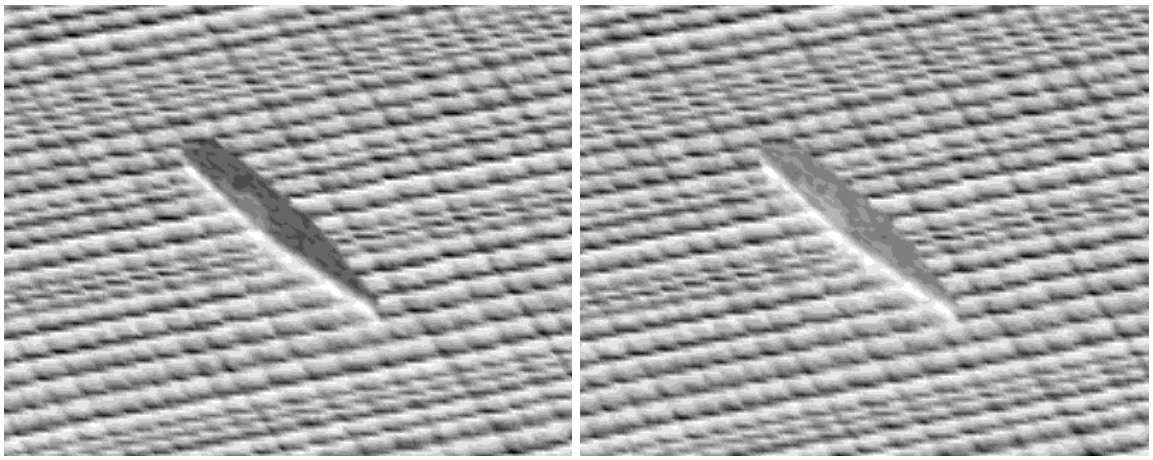


Figure 4.16 Convergence for the infrared ray trace

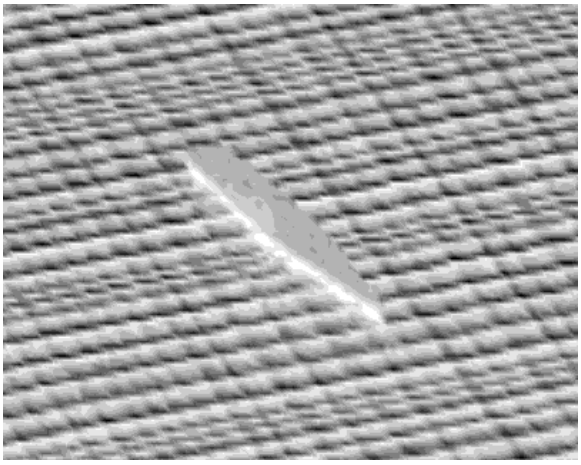
After examining this figure and taking into consideration the time required to compute each case, a value of 4,000,000 energy bundles was chosen for the rest of the infrared ray traces.

4.2.2 Influence of variations in ship reflectivity

The first parameter that was varied was the emissivity of the ship. The emissivity was varied in steps of 0.05 and the best possible value to hide the ship was chosen from those cases studied. Figure 4.17 shows three examples of the images created when the emissivity of the ship is varied.



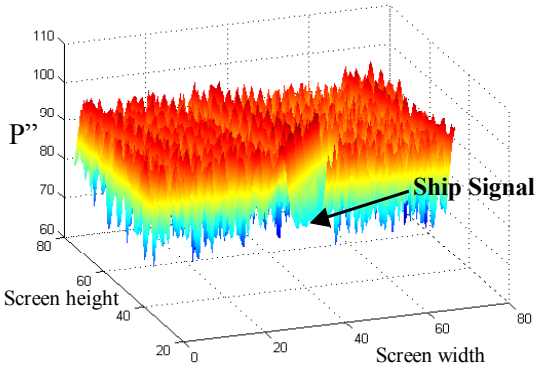
ε = 0.80 **ε = 0.85**



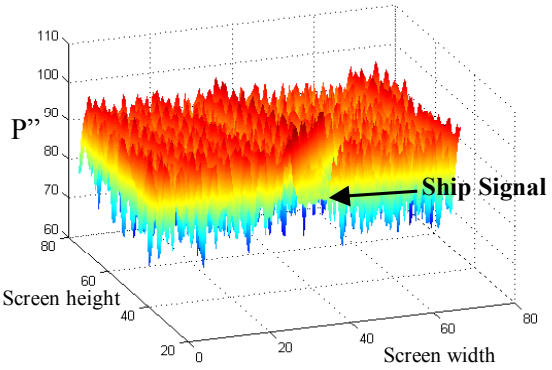
ε = 0.9 0

Figure 4.17 Influence on the infrared image from variations in emissivity, ε, of the ship ($\varepsilon = 0.85$, $r^s = 0.5$, $\theta = 50$ deg, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

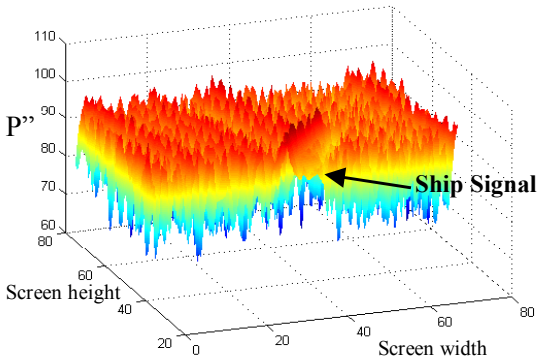
Among the values tested, a ship emissivity of 0.85 provides the minimum contrast between the ship and the sea under the nominal conditions given in Table 4.3 for a ship temperature of 290 K and a sea surface temperature of 290 K. Figure 4.18 shows cross-sections of the signal strength with labels showing the location of the ship.



ε = 0.80



ε = 0.85



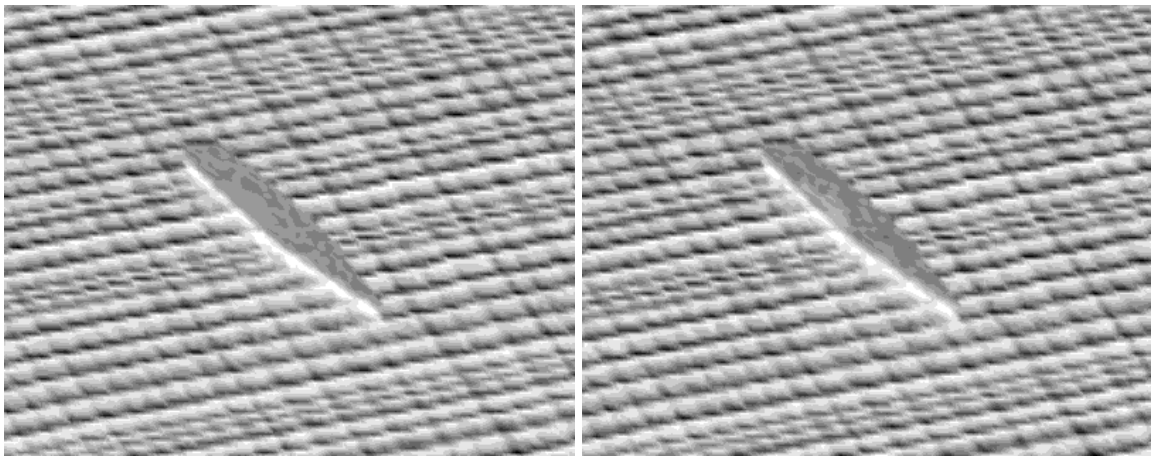
ε = 0.90

Figure 4.18 Influence on the infrared contrast from variations in emissivity, ε, of the ship (ε = 0.85, r^s = 0.5, θ = 50 deg, T_{sea} = 290 K, T_{ship} = 290 K)

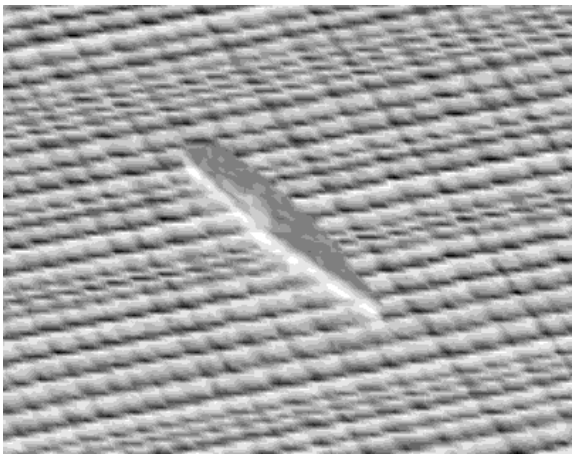
An emissivity of 0.85 shows minimum contrast for the temperatures used, and is therefore chosen as the nominal value.

4.2.3 Influence of variations in ship specularity ratio in the infrared

The second parameter varied was the specularity ratio. This ratio was varied to find a value that would better hide the ship compared to the nominal. Figure 4.19 shows three examples of the images created when the specularity ratio is varied.



$r^s = 0.1$ **$r^s = 0.5$**



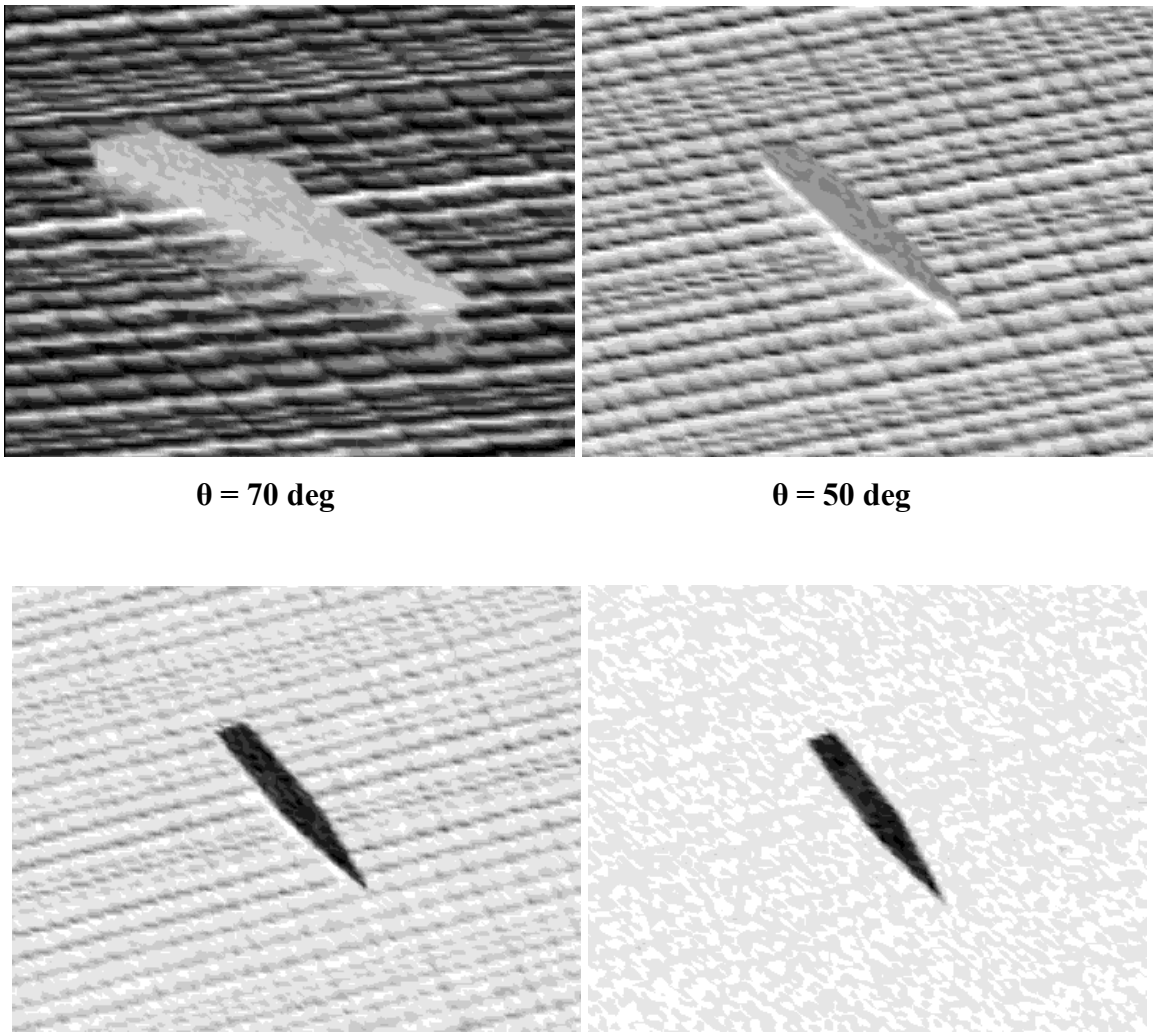
$r^s = .9$

Figure 4.19 Influence on the infrared image from variations in specularity ratio, r^s, of the ship ($\varepsilon = 0.85$, $\theta = 50$ deg, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

Only a slight reduction in the contrast with the side of the ship is obtained when the specularity ratio is reduced.

4.2.4 Variations in the viewing angle

The viewing angle of the optical sensor has a large impact on the infrared signal strength of the ocean just as it did for visible light. Figure 4.20 shows the image created at angles of 70, 30, and 10 deg along with the nominal case of 50 deg.



θ = 70 deg **θ = 50 deg**

θ = 30 deg **θ = 10 deg**

Figure 4.20 Influence on the infrared image due to variations in viewing angle ($\varepsilon = 0.85$, $r^s = 0.1$, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

The contrast between the ship and the ocean is dramatically increased when the viewing angle is changed from the nominal value. Figure 4.21 shows the degree of contrast of the ship with the ocean.

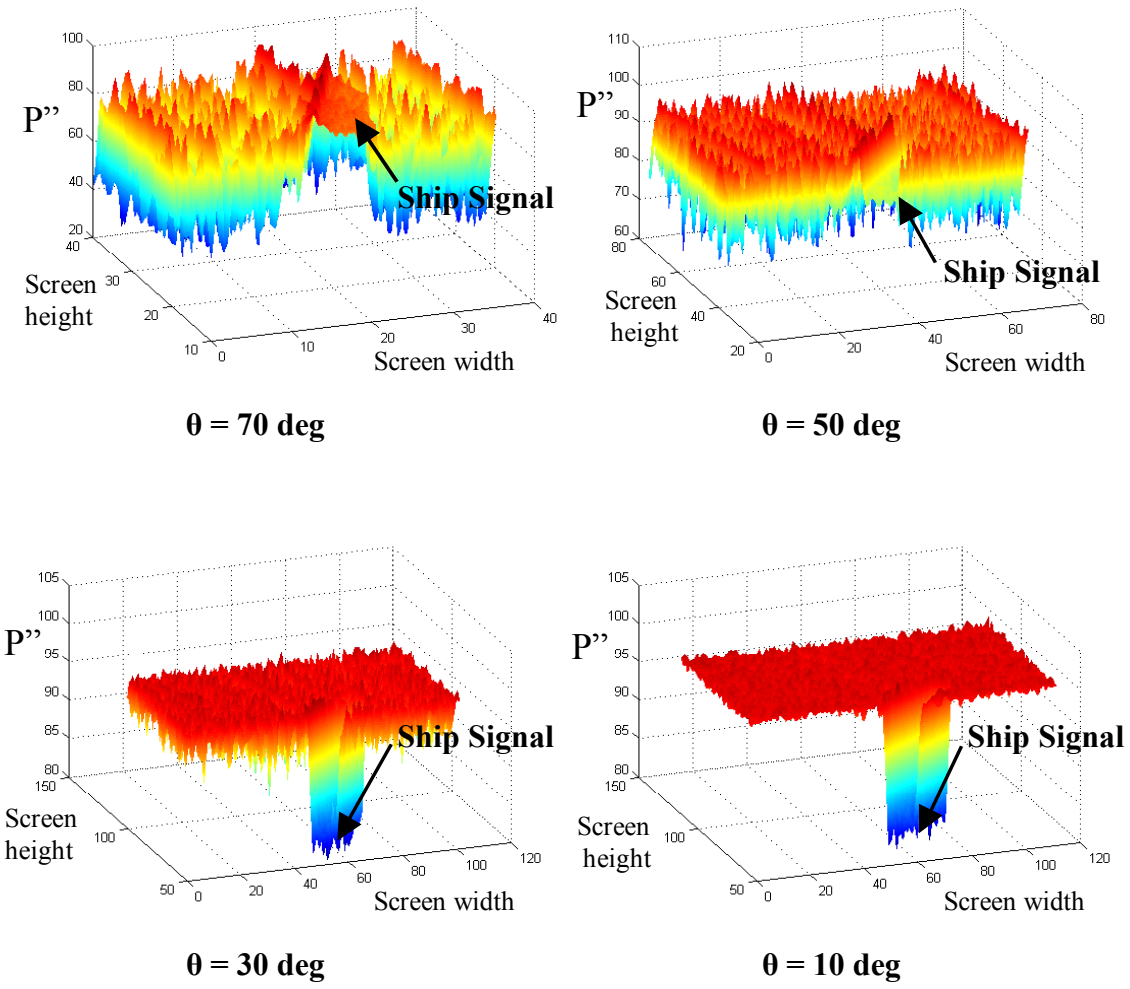
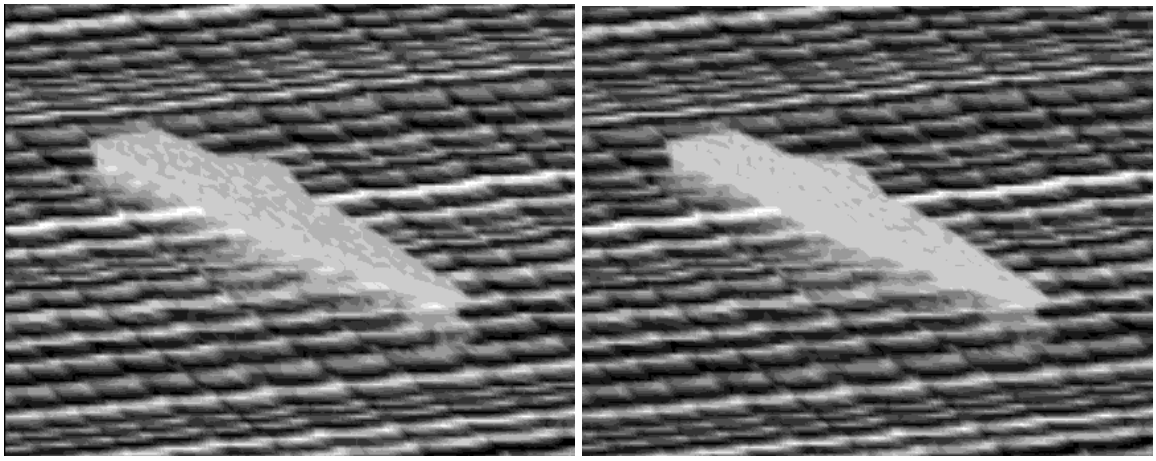


Figure 4.21 Influence on the infrared contrast due to variations in viewing angle ($\varepsilon = 0.85$, $r^s = 0.1$, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

4.2.5 Influence of polarization filters in the infrared

A polarization filter can also be used to increase the contrast in the infrared. Figure 4.22 shows a viewing angle of $\theta = 70$ deg with and without a polarization filter.



Without filter **With vertically polarizing filter**

Figure 4.22 Influence on infrared image from using a vertically polarizing filter ($\varepsilon = 0.85$, $r^s = 0.1$, $\theta = 50$ deg, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

The ship viewed with the filter is slightly easier to detect than the ship viewed with no filter, but the difference is not nearly as dramatic as it is for visible light reflection. Figure 4.23 shows a pixel-by-pixel difference image of the two images shown in Figure 4.22 to more clearly show the difference in contrast. The power of the signal without the filter was reduced by %50 to give both ships approximately the same signal.

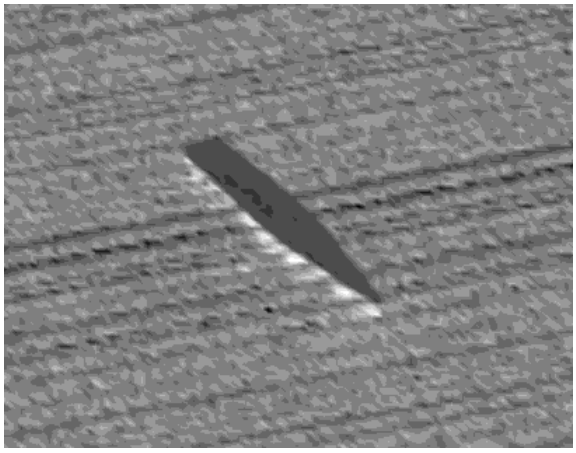


Figure 4.23 The change in infrared contrast produced when using a vertical polarization filter ($\varepsilon = 0.85$, $r^s = 0.1$, $\theta = 50$ deg, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

4.1.7 Ship with water surface properties

In addition to looking at diffuse-specular surface properties for the ship, the case where the water and the ship have the same surface properties was considered. This was done to determine the reduction in contrast that could be accomplished from using engineered surface properties. Figure 4.24 shows the image created at angles of 70, 30, and 10 deg along with the nominal case of 50 deg when the ship is given the same surface properties as the seawater.

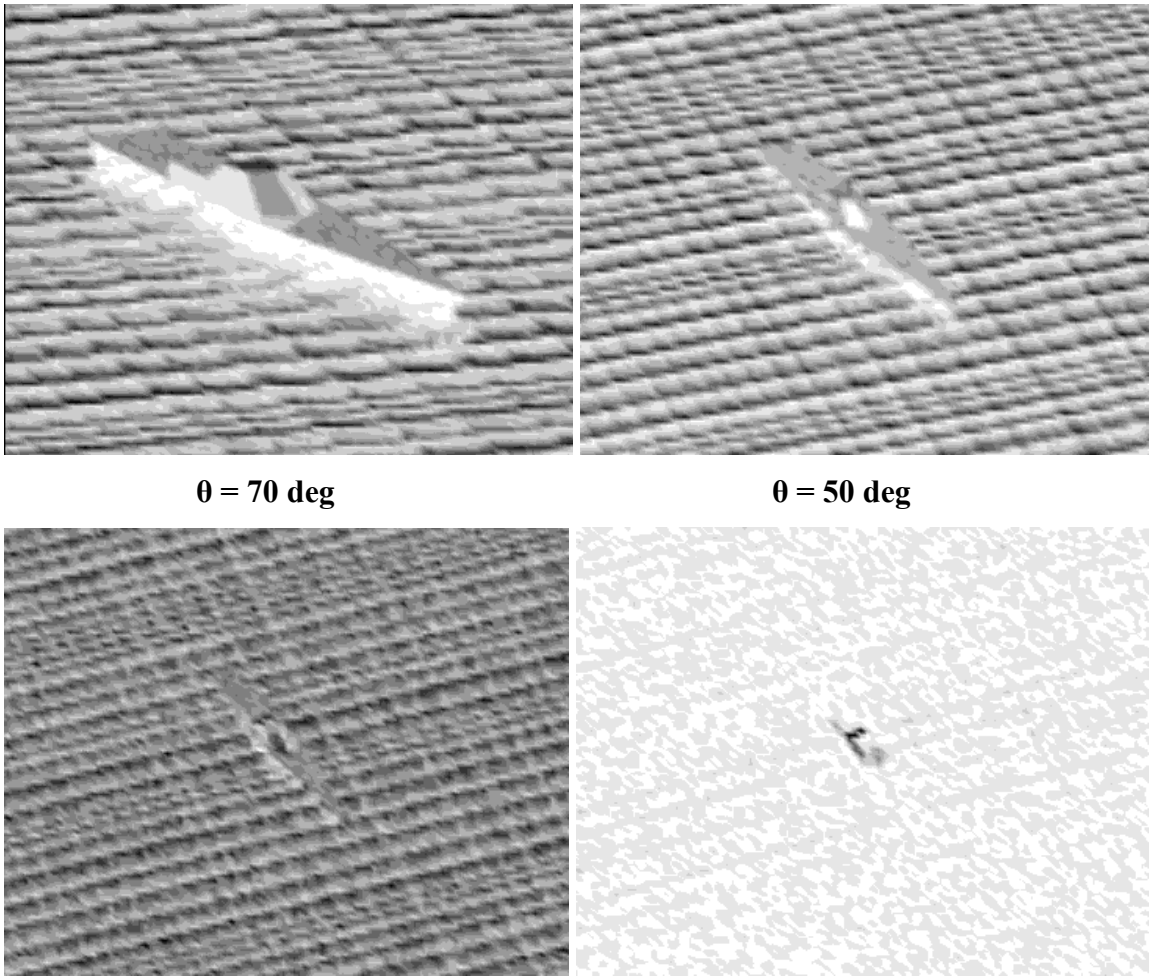
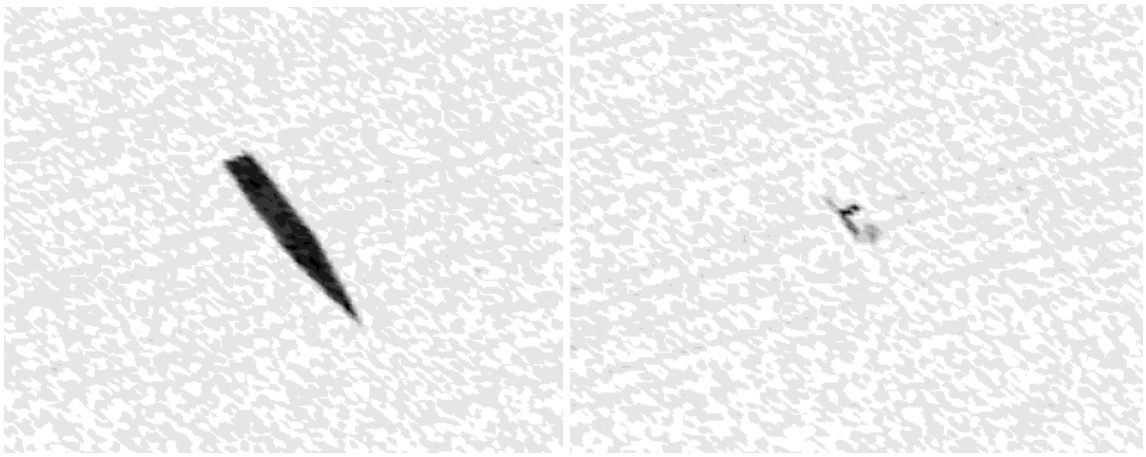


Figure 4.24 Influence on the infrared image due to variations in viewing angle when the ship has the same optical properties as the water ($\varepsilon = 0.85$, $r^s = 0.1$, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

The contrast with the ocean background was greatly reduced, especially at angles much different than the nominal. The worst case that was shown for a diffuse-specular ship was at a zenith angle of $\theta = 10$ deg. A comparison of the two surface properties at this angle is shown in Figure 4.25. Since the ship is now polarizing light in the same way as the ocean, the change in contrast using a polarization filter is significantly reduced, as shown in Figure 4.26.



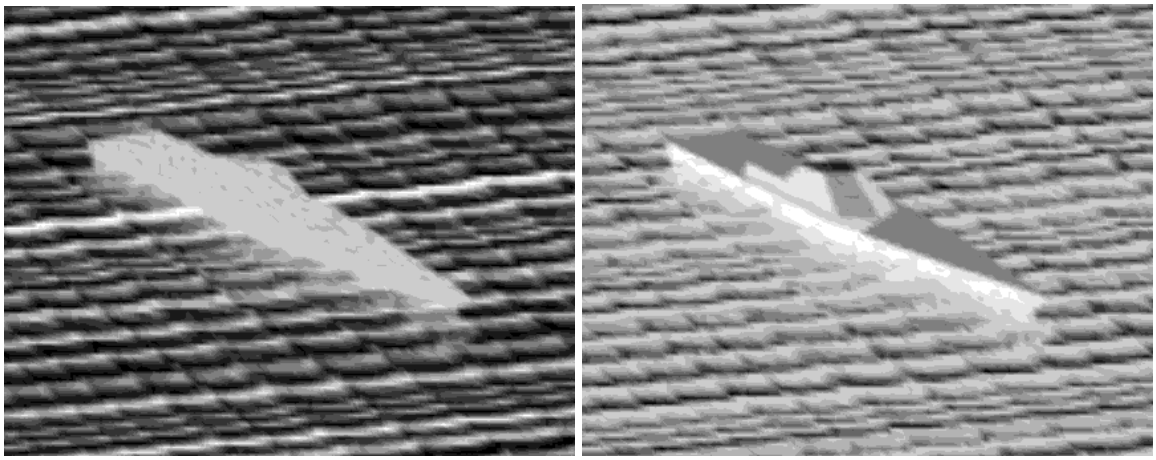
Diffuse-specular **Seawater properties**

Figure 4.25 Comparison of infrared images for a ship having diffuse-specular properties and a ship having seawater surface properties ($\varepsilon = 0.85$, $r^s = 0.1$, $\theta = 10$ deg, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

4.3 Summary

This chapter provides the results of both visible and infrared radiation. A convergence study was performed for both wavelength ranges to determine the minimum number of energy bundles that needed to be traced. Once this was completed, each parameter of interest was varied to determine the effects on the visibility of the ship. The emissivity and the specularity ratio of the ship were varied to determine the optimum values needed to hide the ship when the detector is not fitted with a polarization filter. Once these values were determined, the viewing angle of the optical sensor was varied to determine the effect this has on the visibility of the ship. Polarization filters were then

used to determine the importance of the polarizing properties of the ship and the surrounding seaway. The visible-light ray trace also considered the effects of the location of the sun.



Diffuse-specular **Seawater properties**

Figure 4.26 Comparison of infrared images for a ship having diffuse-specular properties and a ship having seawater surface properties, while using a vertically polarizing filter ($\varepsilon = 0.85$, $r^s = 0.1$, $\theta = 50$ deg, $T_{sea} = 290$ K, $T_{ship} = 290$ K)

CHAPTER 5 Conclusions and recommendations

5.1 Conclusions

We draw the following conclusions based on the results presented in this thesis.

1. The surface properties of naval vessels must be more complex than a simple diffuse-specular, nonpolarizing surface to avoid detection.

2. The viewing angle of the optical sensor can have a large impact on the visibility of a vessel at sea. It is therefore critical to have surface properties that vary with viewing angle in a manner similar to the way the ocean does.

3. Polarization plays a large roll in the visibility of the ship. Polarization filters can be used to create contrast between the ocean and the ship, even when contrast is minimal without a polarization filter. It is very important that the surface properties of the ship polarize light similar to the way the ocean polarizes light.

4. It may be necessary to have separate properties for surfaces on the ship that are nearly vertical. For this particular geometry there was a very noticeable shadow in most of the images that could be reduced by making the sides more reflective than the top of the ship.

5.2 Recommendations for further work

1. The design tool presented in this thesis should be validated using an existing ship, where the surface properties are well known, to determine its accuracy.

2. Sharp angles between panels on the ship have been shown to create contrast from one panel to the next. A study could be performed to determine if more ideal geometry should be used for warships.

3. Add the capability to give the nearly vertical panels of the ship different properties than the nearly horizontal panels of the ship. This would allow the user to determine if the shadows can be reduced without making the ship more visible at other viewing angles.

4. Add the capability to input complex surface properties, such as fully bi-directional reflectivity without having to compile the program each time a change is made. Having the program read in the bi-directional reflectivity from a text file may be a good solution to this problem.

REFERENCES

Baudouel D. *An Efficient Ray-Polygon Intersection*. Graphics Gems, ed. A. S. Glassner, Academic Press, New York 1990.

Churnside J.H. and Shaw J.A. *Scanning-laser glint measurements of sea-surface slope statistics.* Applied Optics. **Vol**. 36, No. 18, pp. 4202-4213. 1997.

Cooper A.W., Lentz W.J., and Walker P.L. *Infrared Polarization ship images and contrast in the MAPTIP experiment.* SPIE. **Vol**. 2828, pp. 85-93. 1996.

Cooper A.W., Lentz W.J., Walker P.L., and Chan P.M. *Infrared Polarization measurements of ship signatures and background contrast.* SPIE. **Vol**. 2223, pp. 300-307. 1994.

Cox C. and Munk W. *Statistics of the sea surface derived from sun glitter.* Journal of Marine Research. **Vol**. 13, No. 2, pp. 198-227. 1954.

Cox C. and Munk W. *Some problems in optical oceanography.* Journal of Marine Research. **Vol**. 14, No. 1, pp. 63-78. 1955.

Eloranta E.W., Parent J.A., Twitty J.T., and Weinman J.A. *Aerosol size distributions: remote determination from air-borne measurements of the solar aureole.* Applied Optics. **Vol**. 15, No. 4, pp. 980-989. 1976.

Gao W., Grant R.H., and Heisler G.M. *Photosynthetically-active radiation: sky radiance distributions under clear and overcast conditions.* Agricultural and Forest Meteorology. **Vol**. 82, pp. 267-292. 1996.

Green A.E.S. and McPeters R.D. *Photographic aureole measurements and the validity of aerosol single scattering.* Applied Optics. **Vol**. 15, No. 10, pp. 2457-2463. 1976.

Hale G.M., and Querry M.R. *Optical constants of water in the 200-nm to 200-μm wavelength region.* Applied Optics. **Vol**. 12, No. 3, pp. 555-563. 1973.

Hobson D.E. and Williams D. *Infrared spectral reflectance of seawater.* Applied Optics. **Vol**. 10, No. 10, pp. 2372-2373. 1971.

Mahan J.R. *Radiation Heat Transfer: A Statistical Approach.* 2002. John Wiley & Sons, Inc. New York. 2002.

Sidran M. *Broadband reflectance and emissivity of specular and rough water surfaces.* Applied Optics. **Vol**. 20, No. 18, pp. 3176-3183. 1981.

Sparrow E.M. and Cess R.D. *Radiation Heat Transfer.* Wadsworth Publishing Company, Inc., Brooks/Cole Publishing Company, Belmont CA, 1966.

Turk J. A. *Acceleration Techniques for the Radiation Analysis of General Computational Fluid Dynamics Solutions Using Reverse Monte Carlo Ray Tracing.* Ph. D. Dissertation, Mechanical Engineering Department. Virginia Polytechnic Institute and State University. Blacksburg, Virginia, U.S.A., 1994.

Internet Web Cites

http://www.celos.psu.edu/kinematics/Kin_tutor06.html

http://www.naval-technology.com/projects/dd21/

http://www.physicsclassroom.com/Class/light/U12L1e.html#refln

Appendix A

```
PROGRAM OCEAN
      IMPLICIT NONE
      DOUBLE PRECISION::DX,DY,WAVESEG,DI,IS,&
            SAS,SCS,AR,ANGLE,RANDOM,DZX,DZY,ZMAX,ZMIN,&
            DT,G,ZM,TW,UZ,RI,PI,C3,C4,DN,C2,TEMP,MMAG,DXX,DYY
      INTEGER::I,J,WAVE,IMIN,IMAX,JMAX,WAVEMAX,INDEX1,&
            INDEX2,K,L,NX,NY,XF,YF,NFX,NFY,LEFT,BOTTOM
      DOUBLE PRECISION,DIMENSION(3)::V1,V2,M
      DOUBLE PRECISION,DIMENSION(20001)::N,SUM1,SUM2,P1,P2,N2
      DOUBLE PRECISION,DIMENSION(:),ALLOCATABLE::X0,Z0X,Y0,Z0Y,
      X,Y,ZX,ZY
      DOUBLE PRECISION,DIMENSION(:,:),ALLOCATABLE::A,B,C,D,Z

      INTERFACE
            FUNCTION CROSS(A,B)
                  DOUBLE PRECISION, DIMENSION(3)::CROSS
                  DOUBLE PRECISION, DIMENSION(3), INTENT(IN)::A,B
            END FUNCTION CROSS
      END INTERFACE

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! DX = Length of facets in the x-direction
! DY = Length of facets in the y-direction
! WAVESEG = # of sections per wave
! DI = Range of random numbers per section
! IS = starting value for the range DI
! SAS = Along-wind standard deviation squared
! SCS = Crosswind standard deviation squared
! AR = adjusted random number used to choose angle for facets
! ANGLE = angle of the facet
! RANDOM = evenly distributed random number
! DZX = change in elevation of a facet in the x-direction
! DZY = change in elevation of a facet in the y-direction
! ZMAX = maximum height of the ocean
! ZMIN = minimum height of the ocean
! DT = air-sea temperature difference
! G = gravity, 9.81
! ZM = height of the glint sensor that took along-wind measurements
! TW = Temperature of the water
! UZ = Wind speed
! RI = parameter
! PI = 3.14
! C3 = expansion coefficient for PDF
! C4 = expansion coefficient for PDF
! DN = increment for generating PDF curve
! C2 = constant
```

```
! TEMP = temporary variable
! MMAG = magnitude of M
! DXX = change in x of a facet in the x-direction
! DYY = change in y of a facet in the y-direction
! I = incrementing variable
! J = incrementing variable
! WAVE = incrementing variable
! IMIN = initial starting point for random number range
! IMAX = final ending point for random number range
! JMAX = number of facets per section
! WAVEMAX = number of waves
! INDEX1 = counting variable
! INDEX2 = counting variable
! K = counting variable
! L = incrementing variable
! NX = number of points in the x-direction
! NY = number of points in the x-direction
! XF = number of points in the x-direction
! YF = number of points in the x-direction
! NFX = number of points in the x-direction
! NFY = number of points in the x-direction
! LEFT = location of the left side of the ocean surface
! BOTTOM = location of the bottom side of the ocean (top view)
! V1 = first tangent vector of facet
! V2 = second tangent vector of facet
! M = normal vector
! N = independent variable for PDF
! SUM1 = sum variable for integration
! SUM2 = sum variable for integration
! P1 = along-wind PDF
! P2 = crosswind PDF
! N2 = independent variable for PDF
! X0 = location of left side of facet for along-wind
! Z0X = height of left side of facet for along-wind
! Y0 = location of left side of facet for crosswind
! Z0Y = height of left side of facet for crosswind
! X = location of nodes in the x-direction
! Y = location of nodes in the y-direction
! ZX = height of nodes in the x-direction
! ZY = height of nodes in the y-direction
! A = x-component of the normal vector
! B = y-component of the normal vector
! C = z-component of the normal vector
! D = distance from the plane to the origin of the coordinate system
! Z = matrix of all ocean heights
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!PROMPT TO ENTER PARAMETERS
      PRINT *,"ENTER THE AIR-SEA TEMPERATURE DIFFERENCE"
      READ *,DT
      PRINT *,"ENTER THE WIND SPEED"
      READ *,UZ
      PRINT *,"ENTER THE NUMBER OF FACETS IN EACH DIRECTION"
      READ *,NFX
      NFY=NFX
      PRINT *,"ENTER THE SEGMENT SIZE"
      READ *,DX
```

```
      DY=DX
!DEFINE BOUNDRIES OF THE OCEAN
      LEFT=-.5*NFX*DX
      BOTTOM=-.5*NFY*DY
      ALLOCATE(A(NFX,NFY),B(NFX,NFY),C(NFX,NFY),D(NFX,NFY),ZX(NFX*2),
            ZY(NFY*2),X0(NFX+1),Y0(NFY+1),Z0X(NFX+1),Z0Y(NFY+1),
            X(NFX*2),Y(NFY*2),Z(NFX+1,NFY+1))
      PRINT *,'GENERATING OCEAN WAVES....'
      G=9.81
      ZM=10
      TW=15
      RI=G*DT*ZM/(TW*UZ*UZ)
      PI=DACOS(DBLE(-1))
      C2=.00316
!FIND STANDARD DEVIATION FOR ALONG-WIND STATISTICS
      IF (RI>=.27) THEN
            SAS=UZ*.65*C2
      ELSE
            SAS=C2*UZ*(1.42-2.8*RI)
      END IF
      C3=-.2349+4.812*SAS
      C4=(3.463-7.643*RI)/10
      DN=.001
      OPEN(UNIT=50,FILE="PDF.TXT",ACTION="WRITE",POSITION="REWIND")
!DETERMINE THE ALONG-WIND PDF
      DO I=1,20001
            N(I)=((I-1)*DN-DBLE(10))
            P1(I)=(((C3/6)*(N(I)**3-N(I)*3))+((C4/24.)*(N(I)**4-
            6.*N(I)**2+3.))+1.)*(1./DSQRT(2.*PI))*DEXP(-N(I)**2/2.)
      END DO
!INTEGRATE THE ALONG-WIND PDF
      SUM1(1)=0
      DO I=2,20001
            SUM1(I)=SUM1(I-1)+DN*(P1(I-1)+P1(I))/2.
      END DO
!CALCULATE THE CROSSWIND PDF
      SCS=.003+1.92E-3*Uz
!DETERMINE THE CROSSWIND PDF
      DO I=1,20001
            N2(I)=((I-1)*DN-DBLE(10))
            P2(I)=(1./DSQRT(2.*PI))*DEXP(-N2(I)**2/2.)
      WRITE(50,'(5F16.10)')N(I)*SQRT(SAS)*180/PI,P1(I)*PI/180/SQRT(SAS)
      ,N2(I)*SQRT(SCS)*180/PI,P2(I)*PI/180/SQRT(SCS),SUM1(I)
      END DO
!INTEGRATE THE CROSSWIND PDF
      SUM2(1)=0
      DO I=2,20001
            SUM2(I)=SUM2(I-1)+DN*(P2(I-1)+P2(I))/2.
      END DO
      NX=NFX+1
      NY=NFY+1
!PARAMETERS TO CONTROL NUMBER AND FREQUENCY OF WAVES
      WAVEMAX=NFX/8
      IMIN=5
      IMAX=15
      WAVESEG=4
      DI=(IMAX-IMIN)/(WAVESEG)
```

```
      INDEX1=1
      ZX(INDEX1)=0
      X(1)=LEFT
!GENERATE WAVES IN THE X-DIRECTION
      DO WAVE=1,WAVEMAX
            CALL RANDOM_NUMBER(RANDOM)
            JMAX=2+3*RANDOM
            DO I=1,WAVESEG
                  IS=IMIN+(I-1)*DI
                  DO J=1,JMAX
                        INDEX1=INDEX1+1
                        CALL RANDOM_NUMBER(RANDOM)
                        AR=(DI*RANDOM/10)+.1*IS
                        IF (AR>=1) THEN
                              AR=AR-1
                        END IF
                        K=1
                        DO WHILE (SUM1(K)<=AR)
                              K=K+1
                        END DO
                        ANGLE=N(K)*DSQRT(SAS)
                        ZX(INDEX1)=ZX(INDEX1-1)+DSIN(ANGLE)*DX
                        X(INDEX1)=X(INDEX1-1)+DX*DCOS(ANGLE)
                  END DO
            END DO
      END DO
      X0=X(1:NX)
      INDEX2=1
      ZY(INDEX2)=0
      Y(1)=BOTTOM
!GENERATE WAVES IN THE Y-DIRECTION
      DO WAVE=1,WAVEMAX
            CALL RANDOM_NUMBER(RANDOM)
            JMAX=2+3*RANDOM
            DO I=1,WAVESEG
                  IS=IMIN+(I-1)*DI
                  DO J=1,JMAX
                        INDEX2=INDEX2+1
                        CALL RANDOM_NUMBER(RANDOM)
                        AR=(DI*RANDOM/10)+.1*IS
                        IF (AR>=1) THEN
                              AR=AR-1
                        END IF
                        K=1
                        DO WHILE (SUM2(K)<=AR)
                              K=K+1
                        END DO
                        ANGLE=N(K)*DSQRT(SCS)
                        DO L=1,INDEX2
                              ZY(INDEX2)=ZY(INDEX2-1)+DSIN(ANGLE)*DY
                              Y(INDEX2)=Y(INDEX2-1)+DY*DCOS(ANGLE)
                        END DO
                  END DO
            END DO
      END DO
      Z0X=ZX(1:NX)
      Z0Y=ZY(1:NY)
```

```
      Y0=Y(1:NY)
      OPEN(UNIT=10,FILE="OCEAN_HEIGHT.TXT",ACTION="WRITE",POSITION="REW
      IND")
!WRITE COORDINATES FOR WAVES IN X-DIRECTION
      PRINT*,NX,NY
      DO I=1,NX
            WRITE(10,'(2F16.10)') X0(I), Z0X(I)
      END DO
!WRITE COORDINATES FOR WAVES IN Y-DIRECTION
      DO J=1,NY
            WRITE(10,'(2F16.10)') Y0(J), Z0Y(J)
      END DO
      CLOSE(UNIT=10)
      PRINT *,'STORING FACET PROPERTIES....'
!FIND EQUATIONS FOR EACH FACET
      TEMP=0
      DO YF=1,NFY
          DO XF=1,NFX
              DZX=Z0X(XF+1)-Z0X(XF)
              DZY=Z0Y(YF+1)-Z0Y(YF)
              DXX=X(XF+1)-X(XF)
              DYY=Y(YF+1)-Y(YF)
              V1=(/DXX,TEMP,DZX/)
              V2=(/TEMP,DYY,DZY/)
              M=CROSS(V1,V2)
              MMAG=DSQRT(DOT_PRODUCT(M,M))
              M=M/MMAG
              A(XF,YF)=M(1)
              B(XF,YF)=M(2)
              C(XF,YF)=M(3)
              D(XF,YF)=-A(XF,YF)*X0(XF)-B(XF,YF)*Y0(YF)-C(XF,YF)*
              (Z0X(XF)+Z0Y(YF))
           END DO
      END DO
!FIND MAX AND MIN OCEAN HEIGHT
      DO I=1,NFX
            DO J=1,NFY
                  Z(I,J)=(Z0X(I)+Z0Y(J))
            END DO
      END DO
      ZMAX=MAXVAL(Z)
      ZMIN=MINVAL(Z)
!WRITE MAX AND MIN HEIGHT AND NUMBER OF FACETS IN EACH DIRECTION
      OPEN(UNIT=20,FILE="OCEAN_PROPERTIES.TXT",ACTION="WRITE",POSITION=
      "REWIND")
      WRITE(20,'(8F16.10)') ZMAX,ZMIN
      WRITE(20,'(2I16)') NFX,NFY
!WRITE CONSTANTS THAT MAKE UP THE EQUATION FOR EACH FACETS
      DO I=1,NFX
            DO J=1,NFY
                  WRITE(20,'(4F16.10)') A(I,J),B(I,J),C(I,J),D(I,J)
            END DO
      END DO
      CLOSE(UNIT=20)
!USED FOR RENDERED DRAWINGS
      OPEN(UNIT=30,FILE="ACAD_OCEAN.TXT",ACTION="WRITE",POSITION=
      "REWIND")
```

```
      WRITE(30,'(A6)')"3DMESH"
      WRITE(30,'(I3)') NFX
      WRITE(30,'(I3)') NFY
      DO I=1,NFX
            DO J=1,NFY
                  WRITE(30,'(F10.2,",",F10.2,",",F10.4)') X0(I),Y0(J),
            Z(I,J)
            END DO
      END DO
END PROGRAM OCEAN

      FUNCTION CROSS(A,B)
            IMPLICIT NONE
            DOUBLE PRECISION, DIMENSION(3)::CROSS
            DOUBLE PRECISION, DIMENSION(3), INTENT(IN)::A,B
            CROSS=(/A(2)*B(3)-A(3)*B(2),A(3)*B(1)-A(1)*B(3),A(1)*B(2)-
            A(2)*B(1)/)
      END FUNCTION CROSS
```

Appendix B

```
PROGRAM SHIP
      IMPLICIT NONE
      DOUBLE
      PRECISION::PI,MID,YMAX,XMIN,XMAX,DZX,DYX,DXX,DZY,DYY,DXY,MMAG,A,B
      ,C,D,ZMAX, X0,Y0,Z0,YMIN,DELTAX,PHI,XOFF,YOFF
      INTEGER::I,J,DY,YMINI,LI,YLM,Y1,Y2,COUNT1,XF,YF,COUNT2
      DOUBLE PRECISION,DIMENSION(:,:), ALLOCATABLE::X,Y,Z,XN,YN
      DOUBLE PRECISION,DIMENSION(4,7)::XT,YT,ZT,XTN,YTN
      DOUBLE PRECISION,DIMENSION(3,6)::AT,BT,CT,DT
      DOUBLE PRECISION,DIMENSION(3)::V1,V2,M

      INTERFACE
            FUNCTION CROSS(A,B)
                  DOUBLE PRECISION, DIMENSION(3)::CROSS
                  DOUBLE PRECISION, DIMENSION(3), INTENT(IN)::A,B
            END FUNCTION CROSS
      END INTERFACE

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! PI = 3.14
! MID = WIDTH OF THE SHIP AT ITS MID POINT
! YMAX = LOCATION OF THE FRONT OF THE SHIP
! XMIN = X-COORDINATE FURTHEST TO THE PORT SIDE OF THE SHIP
! XMAX = X-COORDINATE FURTHEST TO THE STARBOARD SIDE OF THE SHIP
! XFMX = max value in a range of facets in the x-direction
! DZX = Z-component of V1
! DYX = Y-component of V1
! DXX = X-component of V1
! DZY = Z-component of V1
! DYY = Y-component of V1
! DXY = X-component of V1
! MMAG = Magnitude of M
! A = X-component of M
! B = Y-component of M
! C = Z-component of M
! D = distance from the plane to the origin of the coordinate system
! ZMAX = maximum height of the ocean
! X0 = X-coordinate of starting point for normal vector
! Y0 = Y-coordinate of starting point for normal vector
! Z0 = Z-coordinate of starting point for normal vector
! YMIN = Location of the back of the ship
! DELTAX = Change in x from the deck to bottom of topside
! PHI = angle of rotation of ship
! XOFF = x offset of ship
! YOFF = y offset of ship
! I = incrementing variable
```

```
! J = incrementing variable
! DY = SPACING OF COORDINATES IN THE Y-DIRECTION
! YMINI = Location of the back of the ship (integer)
! LI = COUNTING VARIABLE
! YLM = COUNTING VARIABLE
! Y1 = incrementing variable
! Y2 = incrementing variable
! COUNT1 = COUNTING VARIABLE
! XF = incrementing variable
! YF = incrementing variable
! COUNT2 = COUNTING VARIABLE
! X = x-coordinates
! Y = y-coordinates
! Z = z-coordinates
! XN = x-coordinates after rotation
! YN = y-coordinates after rotation
! XT = x-coordinates of top part of ship
! YT = y-coordinates of top part of ship
! ZT = z-coordinates of top part of ship
! XTN = x-coordinates of top part of ship after rotation
! YTN = y-coordinates of top part of ship after rotation
! AT = X-component of M
! BT = Y-component of M
! CT = Z-component of M
! DT = distance from the plane to the origin of the coordinate system
! V1 = First tangent vector of facet
! V2 = Second tangent vector of facet
! M = Normal vector
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

COUNT2=0
! PARAMETERS THAT CONTROL THE SIZE OF THE SHIP
      PI=DACOS(DBLE(-1))
      MID=10 !WIDTH OF THE SHIP AT ITS MID POINT
      DY=5 !SPACING OF COORDINATES IN THE Y-DIRECTION
      YMINI=-70 !LOCATION OF THE BACK OF THE SHIP
      YMAX=12*(MID)**(.8) !LOCATION OF THE FRONT OF THE SHIP
      XMIN=-MID !X-COORDINATE FURTHEST TO THE PORT SIDE OF THE SHIP
      XMAX=MID !X-COORDINATE FURTHEST TO THE STARBOARD SIDE OF THE SHIP
      LI=1 !COUNTING VARIABLE
      YLM=YMINI
! FIND THE SECOND MOST FORWARD Y-COORDINATE
      DO WHILE (YLM<=YMAX-DY)
            LI=LI+1
            YLM=YLM+DY
      END DO
      LI=LI+1
      ALLOCATE(X(4,LI),Y(4,LI),Z(4,LI),XN(4,LI),YN(4,LI))

! FIND X,Y,Z COORDINATES OF POINTS ON THE HULL AND DECK
      COUNT1=0
      DO Y1=YMINI,0,DY
! COORDINATES FOR THE BACK HALF OF THE SHIP, STARBOARD SIDE
         COUNT1=COUNT1+1
         Y(2,COUNT1)=Y1 !COORDINATES FOR THE DECK
         X(2,COUNT1)=MID-(DBLE(Y1)/(-50))**(1.25)
! COORDINATES FOR THE DECK
```

```
          Z(2,COUNT1)=0 !COORDINATES FOR THE DECK
          Y(1,COUNT1)=Y1 !COORDINATES FOR THE BOTTOM OF THE HULL
          DELTAX=10*DTAN(10*PI/180)
! CHANGE IN X FROM THE DECK TO THE BOTTOM OF THE HULL DIRECTLY BELOW
          IF (DELTAX>X(2,COUNT1)) THEN
! COORDINATES FOR THE BOTTOM OF THE HULL
            X(1,COUNT1)=0
            Z(1,COUNT1)=Z(1,COUNT1)-X(1,COUNT1)*DTAN(80*PI/180)
          ELSE
            X(1,COUNT1)=X(2,COUNT1)-DELTAX
            Z(1,COUNT1)=Z(2,COUNT1)-10
          END IF

      END DO

      DO Y2=DY,YLM,DY
! COORDINATES FOR THE FRONT HALF OF THE SHIP, STARBOARD SIDE
          COUNT1=COUNT1+1
          Y(2,COUNT1)=Y2!COORDINATES FOR THE DECK
          X(2,COUNT1)=MID-(DBLE(Y2)/12)**(1.25)
! COORDINATES FOR THE DECK
          Z(2,COUNT1)=0!COORDINATES FOR THE DECK
          Y(1,COUNT1)=Y2!COORDINATES FOR THE BOTTOM OF THE HULL
          DELTAX=10*DTAN(10*PI/180)
          IF (DELTAX>X(2,COUNT1)) THEN
! COORDINATES FOR THE BOTTOM OF THE HULL
            X(1,COUNT1)=0
            Z(1,COUNT1)=Z(2,COUNT1)-X(2,COUNT1)*DTAN(80*PI/180)
          ELSE
            X(1,COUNT1)=X(2,COUNT1)-DELTAX
            Z(1,COUNT1)=Z(2,COUNT1)-10
          END IF
      END DO
      COUNT1=COUNT1+1
      X(2,COUNT1)=0
      Y(2,COUNT1)=YMAX
      Z(2,COUNT1)=0
      X(1,COUNT1)=0
      Y(1,COUNT1)=Y(2,COUNT1)
      Z(1,COUNT1)=0
! COORDINATES FOR THE PORT SIDE OF THE SHIP
      X(3,:)=-X(2,:) !DECK
      Y(3,:)=Y(2,:) !DECK
      Z(3,:)=Z(2,:) !DECK
      X(4,:)=-X(1,:) !HULL
      Y(4,:)=Y(1,:) !HULL
      Z(4,:)=Z(1,:) !HULL
      Z=Z+5 ! MOVE THE DECK ABOVE WATER LEVEL
! FIND X,Y,Z COORDINATES OF POINTS ON THE TOP SECTION
! (1,*) INDICATES BOTTOM STARBOARD
! (2,*) INDICATES TOP STARBOARD
! (3,*) INDICATES TOP PORT
! (4,*) INDICATES BOTTOM PORT
      ZT(2,1)=5
      ZT(2,2)=3+ZT(2,1)
      ZT(2,3)=3+ZT(2,1)
      ZT(2,4)=8+ZT(2,1)
```

```
      ZT(2,5)=8.5+ZT(2,1)
      ZT(2,6)=9+ZT(2,1)
      XT(2,1)=7
      XT(2,2)=XT(2,1)-(ZT(2,2)-ZT(2,1))*DTAN(15*PI/180)
      XT(2,3)=XT(2,2)
      XT(2,4)=XT(2,1)-(ZT(2,4)-ZT(2,1))*DTAN(15*PI/180)
      XT(2,5)=XT(2,1)-(ZT(2,5)-ZT(2,1))*DTAN(15*PI/180)
      XT(2,6)=XT(2,1)-(ZT(2,6)-ZT(2,1))*DTAN(15*PI/180)
      XT(2,7)=0
      ZT(2,7)=ZT(2,6)+.8*XT(2,5)/9.5
      YT(2,1)=-40
      YT(2,2)=YT(2,1)+(ZT(2,2)-ZT(2,1))*DTAN(15*PI/180)
      YT(2,3)=YT(2,2)+15
      YT(2,4)=YT(2,3)+(ZT(2,4)-ZT(2,3))*DTAN(40*PI/180)
      YT(2,5)=YT(2,4)+4.75
      YT(2,6)=YT(2,5)+4.75
      YT(2,7)=YT(2,6)+.8*XT(2,5)
      XT(1,:)=XT(2,1)
      XT(1,7)=0
      ZT(1,:)=ZT(2,1)
      YT(1,1)=YT(2,1)+.4
      YT(1,2:5)=YT(2,2:5)
      YT(1,6)=YT(2,1)+40
      YT(1,7)=YT(1,6)+.8*XT(2,1)
      XT(3,:)=-XT(2,:)
      XT(4,:)=-XT(1,:)
      YT(3,:)=YT(2,:)
      YT(4,:)=YT(1,:)
      ZT(3,:)=ZT(2,:)
      ZT(4,:)=ZT(1,:)

! ORIENT THE SHIP
! PROMPT USER TO ENTER ROTATION ANGLE AND TRANSLATION OFFSET
      PRINT *,"ENTER THE ANGLE THAT DESCRIBES THE ORIENTATION OF THE
      SHIP (DEGREES)"
      READ *,PHI
      PHI=(PHI-90)*PI/180
      PRINT *,"ENTER THE X-OFFSET"
      READ *,XOFF
      PRINT *,"ENTER THE Y-OFFSET"
      READ *,YOFF
!ROTATE THE SHIP
      XN=X*DCOS(PHI)-Y*DSIN(PHI)
      YN=X*DSIN(PHI)+Y*DCOS(PHI)
      XTN=XT*DCOS(PHI)-YT*DSIN(PHI)
      YTN=XT*DSIN(PHI)+YT*DCOS(PHI)
!TRANSLATE THE SHIP
      X=XN+XOFF
      Y=YN+YOFF
      XT=XTN+XOFF
      YT=YTN+YOFF
!CREATE THE MESHES REQUIRED FOR ACAD TO BUILD THE SHIP
      OPEN(UNIT=10,FILE="ACAD_SHIP.TXT",ACTION="WRITE",
      POSITION="REWIND")
!BASE
      WRITE(10,'(A6)')"3DMESH"
      WRITE(10,'(I1)') 4
```

```
      WRITE(10,'(I2)') COUNT1
      DO I=1,4
          DO J=1,COUNT1
              WRITE(10,'(F8.4,",",F8.4,",",F8.4)') X(I,J),Y(I,J),Z(I,J)
          END DO
      END DO
!BACK
      WRITE(10,'(A6)')"3DMESH"
      WRITE(10,'(I1)') 2
      WRITE(10,'(I1)') 2
      DO I=1,2
          WRITE(10,'(F8.4,",",F8.4,",",F8.4)') X(I,1),Y(I,1),Z(I,1)
      END DO
      DO I=4,3,-1
          WRITE(10,'(F8.4,",",F8.4,",",F8.4)') X(I,1),Y(I,1),Z(I,1)
      END DO
      ZMAX=MAXVAL(ZT)
      YMIN=DBLE(YMINI)
!TOP
      WRITE(10,'(A6)')"3DMESH"
      WRITE(10,'(I1)') 4
      WRITE(10,'(I1)') 7
      DO I=1,4
            DO J=1,7
                  WRITE(10,'(F8.4,",",F8.4,",",F8.4)') XT(I,J),YT(I,J),
                  ZT(I,J)
          END DO
      END DO
      OPEN(UNIT=20,FILE="SHIP_FACETS.TXT",ACTION="WRITE",
      POSITION="REWIND")
      WRITE(20,'(1X,I4)') 6*3*2+(LI-1)*3*2-3
!FIND EQUATIONS OF FACETS FOR RAY TRACE
!TOP
      DO XF=1,3
            DO YF=1,6
                  DO J=0,1
                        I=2*J-1
                        X0=XT(XF+J,YF+J)
                        Y0=YT(XF+J,YF+J)
                        Z0=ZT(XF+J,YF+J)
                        DZX=ZT(XF+1,YF)-Z0
                        DZY=ZT(XF,YF+1)-Z0
                        DXX=XT(XF+1,YF)-X0
                        DYX=YT(XF+1,YF)-Y0
                        DXY=XT(XF,YF+1)-X0
                        DYY=YT(XF,YF+1)-Y0
                        V1=(/DXX,DYX,DZX/)
                        V2=(/DXY,DYY,DZY/)
                        M=I*CROSS(V1,V2)
                        MMAG=DSQRT(DOT_PRODUCT(M,M))
                        IF (MMAG>1E-9) THEN
                              M=M/MMAG
                              A=M(1)
                              B=M(2)
                              C=M(3)
                              D=-A*X0-B*Y0-C*Z0
                              WRITE(20,'(1X,8F16.8)') A,B,C,D,X0,Y0,Z0
```

```
                              WRITE(20,'(1X,3F16.8)') V1,V2
                              COUNT2=COUNT2+1
                        END IF
                  END DO
            END DO
      END DO
!BASE
      DO XF=1,3
            DO YF=1,LI-1
                  DO J=0,1
                        I=2*J-1
                        X0=X(XF+J,YF+J)
                        Y0=Y(XF+J,YF+J)
                        Z0=Z(XF+J,YF+J)
                        DZX=Z(XF+1,YF)-Z0
                        DZY=Z(XF,YF+1)-Z0
                        DXX=X(XF+1,YF)-X0
                        DYX=Y(XF+1,YF)-Y0
                        DXY=X(XF,YF+1)-X0
                        DYY=Y(XF,YF+1)-Y0
                        V1=(/DXX,DYX,DZX/)
                        V2=(/DXY,DYY,DZY/)
                        M=I*CROSS(V1,V2)
                        MMAG=DSQRT(DOT_PRODUCT(M,M))
                        IF (ABS(MMAG)>1E-9) THEN
                              M=M/MMAG
                              A=M(1)
                              B=M(2)
                              C=M(3)
                              D=-A*X0-B*Y0-C*Z0
                              WRITE(20,'(1X,7F16.8)') A,B,C,D,X0,Y0,Z0
                              WRITE(20,'(1X,3F16.8)') V1,V2
                              COUNT2=COUNT2+1
                        END IF
                  END DO
            END DO
      END DO
!BACK
      DO J=0,1
            I=2*J-1
            X0=X(1+2*J,1)
            Y0=Y(1+2*J,1)
            Z0=Z(1+2*J,1)
            DZX=Z(4,1)-Z0
            DZY=Z(2,1)-Z0
            DXX=X(4,1)-X0
            DYX=Y(4,1)-Y0
            DXY=X(2,1)-X0
            DYY=Y(2,1)-Y0
            V1=(/DXX,DYX,DZX/)
            V2=(/DXY,DYY,DZY/)
            M=I*CROSS(V1,V2)
            MMAG=DSQRT(DOT_PRODUCT(M,M))
            IF (ABS(MMAG)>1E-9) THEN
                  M=M/MMAG
                  A=M(1)
                  B=M(2)
```

```
                  C=M(3)
                  D=-A*X0-B*Y0-C*Z0
                  WRITE(20,'(1X,7F16.8)') A,B,C,D,X0,Y0,Z0
                  WRITE(20,'(1X,3F16.8)') V1,V2
                  COUNT2=COUNT2+1
            END IF
      END DO

END PROGRAM SHIP

      FUNCTION CROSS(A,B)
            IMPLICIT NONE
            DOUBLE PRECISION, DIMENSION(3)::CROSS
            DOUBLE PRECISION, DIMENSION(3), INTENT(IN)::A,B
            CROSS=(/A(2)*B(3)-A(3)*B(2),A(3)*B(1)-A(1)*B(3),A(1)*B(2)-
            A(2)*B(1)/)
      END FUNCTION CROSS
```

Appendix C

```
PROGRAM MCRT
      IMPLICIT NONE
      DOUBLE PRECISION::RANDOM,XE,YE,ZE,DIST,NW,PPOWER,SPOWER,XS,
      YS,ZS,LE,ME,NE,VPMAG,VSMAG,T,T1,T2,TI,XI,YI,ZI,XP,YP,ZP,VRMAG,
      COSTHETA,THETA,PI,ZMAX,ZMIN,XRANGE1,XRANGE2,YRANGE1,YRANGE2,
      THETAT,RP,RS,U0,U1,U2,W0,W1,W2,ALPHA,BETA,RTYPE,NMAG,THETAD,PHID,
      VBSMAG,VCSMAG,THETAS,DTHETA,D_SCREEN,T_SCREEN,THETAX,THETAY,X2,
      Y2,Z2,RSHIP,SPECULAR,THETAVIEW,PHIVIEW,ODIST,XMAX,YMAX,MAXXV,
      MAXYV,MINXV,MINYV,XS1,YS1,ZS1,TEMP2

      INTEGER::I,J,RAY,ROW,CHECK,RNY,RNX,FREERAY,XF,YF,VECT,MINXF,
      MAXXF,MINYF,MAXYF,NFX,NFY,RAYSX,RAYSY,TRAYS,NSF,SFN,PLANE,TRACE,T
      COUNT,ROWT,TTYPE

      DOUBLE PRECISION,DIMENSION(:),ALLOCATABLE::XSCREEN,YSCREEN,
      ZSCREEN,THETAR,PHIR,HPOWER,VPOWER,FACET,AS,BS,CS,DS,X0S,Y0S,Z0S,X
      OUT,YOUT,SURFACE,X0,Y0,Z0X,Z0Y,XSCREEN2,YSCREEN2

      INTEGER, DIMENSION(:),ALLOCATABLE::REFLECTION,TEMP
      DOUBLE PRECISION,DIMENSION(:,:),ALLOCATABLE::A,B,C,D,Z,V1,V2
      DOUBLE PRECISION,DIMENSION(1000,3)::VP,VS
      DOUBLE PRECISION,DIMENSION(3)::VE,ZAXIS,N,VI,VR,TN1,TN2,VBS,VCS,
      S_CENTER,R_AXIS,R_AXIS1,VCSR

      INTEGER, PARAMETER::DP=SELECTED_REAL_KIND(14)
      COMPLEX(DP)::MW,COSTHETAT

      INTERFACE
            FUNCTION CROSS(A,B)
                  DOUBLE PRECISION, DIMENSION(3)::CROSS
                  DOUBLE PRECISION, DIMENSION(3), INTENT(IN)::A,B
            END FUNCTION CROSS
      END INTERFACE

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! RANDOM = uniformly distributed random number
! XE = x-coordinate for emission point
! YE = y-coordinate for emission point
! ZE = z-coordinate for emission point
! DIST = distance to screen from sensor
! PPOWER = magnitude of power parallel to a facet
! SPOWER = magnitude of power perpendicular to a facet
! XS = x-coordinate of intersection point with screen
! YS = y-coordinate of intersection point with screen
! ZS = z-coordinate of intersection point with screen
! LE = x-direction cosine
```

```
! ME = y-direction cosine
! NE = z-direction cosine
! VPMAG = magnitude of vector VP
! VSMAG = magnitude of vector VS
! T = length of ray
! T1 = length of ray to Zmax
! T2 = length of ray to Zmin
! TI = length of ray to an intersection point
! XI = x-coordinate of an intersection point
! YI = y-coordinate of an intersection point
! ZI = z-coordinate of an intersection point
! XP = x-coordinate of the intersection point
! YP = y-coordinate of the intersection point
! ZP = z-coordinate of the intersection point
! VRMAG = Magnitude of vector VR
! COSTHETA = cos(theta)
! THETA = angle of incidence
! PI = 3.14
! ZMAX = max height of ocean
! ZMIN = min height of ocean
! XRANGE1 = x-value of intersection with ocean top
! XRANGE2 = x-value of intersection with ocean bottom
! YRANGE1 = y-value of intersection with ocean top
! YRANGE2 = y-value of intersection with ocean bottom
! THETAT = imaginary transmission angle
! RP = parallel reflectivity
! RS = perpendicular reflectivity
! U0 = parameter for determining dominant plane
! U1 = parameter for determining dominant plane
! U2 = parameter for determining dominant plane
! W0 = parameter for determining dominant plane
! W1 = parameter for determining dominant plane
! W2 = parameter for determining dominant plane
! ALPHA = parameter for determining if intersection point is on facet
! BETA = parameter for determining if intersection point is on facet
! RTYPE = indicates type of reflection
! NMAG = magnitude of normal vector
! THETAD = azimuth angle of diffusely reflected ray
! PHID = zenith angle of diffusely reflected ray
! VBSMAG = Magnitude of VBS vector
! VCSMAG = Magnitude of VCS vector
! THETAS = ANGLE BETWEEN VCS AND VBS
! DTHETA = difference in angle between each ray
! D_SCREEN = Distance from screen center to origin of coordinate system
! T_SCREEN = length of ray to reach screen center
! THETAX = change in x-direction relative to the center of the screen
! THETAY = change in y-direction relative to the center of the screen
! X2 = x-Direction cosine before horizontal rotation
! Y2 = x-Direction cosine before horizontal rotation
! Z2 = x-Direction cosine before horizontal rotation
! RSHIP = reflectivity of the ship
! SPECULAR = specularity ratio
! THETAVIEW = azimuth angle of sensor
! PHIVIEW = zenith angle of sensor
! ODIST = distance from sensor to coordinate system origin
! MAXXV = maximum possible x-value of intersection with ocean
! MAXYV = maximum possible y-value of intersection with ocean
```

```
! MINXV = minimum possible x-value of intersection with ocean
! MINYV = minimum possible y-value of intersection with ocean
! XS1 = x-coordinate of intersection point with screen for first ray
! YS1 = y-coordinate of intersection point with screen for first ray
! ZS1 = z-coordinate of intersection point with screen for first ray
! TEMP2 = temporary variable
! I = incrementing variable
! J = incrementing variable
! RAY = counts # of rays
! ROW = Counts # of rows of rays
! CHECK = checks to determine location of reflection if any
! RNY = incrementing variable for rays in the x-direction
! RNX = incrementing variable for rays in the y-direction
! FREERAY = specifies if a ray has no intersection points
! XF = facet # in the x-direction
! YF = facet # in the y-direction
! VECT = keeps record of all vectors that trace the current ray
! MINXF = minimum possible x-facet for intersection with ocean
! MAXXF = maximum possible x-facet for intersection with ocean
! MINYF = minimum possible y-facet for intersection with ocean
! MAXYF = maximum possible y-facet for intersection with ocean
! NFX = # of facets in the x-direction
! NFY = # of facets in the y-direction
! RAYSX = number of rays in the x-direction
! RAYSY = number of rays in the y-direction
! TRAYS = total # of rays
! NSF = # of ship facets
! SFN = incrementing variable for ship facets
! PLANE = specifies the dominant plane
! TTYPE = indicates type of ray trace, infrared or visible
! XSCREEN = x-coordinates of intersection points with screen
! YSCREEN = y-coordinates of intersection points with screen
! ZSCREEN = z-coordinates of intersection points with screen
! THETAR = azimuth angle of ray reflected to sky
! PHIR = zenith angle of ray reflected to sky
! HPOWER = power of horizontally polarized light
! VPOWER = power of vertically polarized light
! AS = x-component of the normal vector of ship facet
! BS = x-component of the normal vector of ship facet
! CS = x-component of the normal vector of ship facet
! DS = Distance from ship facet to origin of coordinate system
! X0S = X0 for ship facets
! Y0S = Y0 for ship facets
! Z0S = Z0 for ship facets
! XOUT = x-coordinate of screen location after adjustments
! YOUT = y-coordinate of screen location after adjustments
! SURFACE = indicates type of surface
! X0 = location of left side of facet for along-wind
! Y0 = location of left side of facet for crosswind
! Z0X = height of left side of facet for along-wind
! Z0Y = height of left side of facet for crosswind
! XSCREEN2 = temporary variable used while adjusting screen location
! YSCREEN2 = temporary variable used while adjusting screen location
! REFLECTION = # of reflections
! TEMP = temporary variable
! A = x-component of the normal vector of ocean facet
! B = x-component of the normal vector of ocean facet
```

```
! C = x-component of the normal vector of ocean facet
! D = Distance from ocean facet to origin of coordinate system
! Z = matrix of all ocean heights
! V1 = first tangent vector of ocean facet
! V2 = second tangent vector of ocean facet
! VP = Vector in direction of parallel electric field
! VS = Vector in direction of perpendicular electric field
! VE = Vector in direction of emission
! ZAXIS = z-axis of global coordinate system
! N = Normal vector
! VI = DIRECTION OF INCIDENT RAY
! VR = DIRECTION OF REFLECTED RAY
! TN1 = first tangent vector of ship facet
! TN2 = second tangent vector of ship facet
! VBS = Vector locating bottom of screen from emission point
! VCS = Vector locating center of screen from emission point
! S_CENTER = Vector locating center of screen
! R_AXIS = First axis of rotation
! R_AXIS1 = Second axis of rotation
! VCSR = VECTOR LOCATING MIDPOINT OF SCREEN AT GIVEN VERTICAL LOCATION
! MW = complex index of refraction
! COSTHETAT = cos(thetat)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!READ IN INFORMATION ON OCEAN GEOMETRY
      PRINT *, 'READING OCEAN GEOMETRY.....'
      OPEN(UNIT=20,FILE="OCEAN_PROPERTIES.TXT",ACTION="READ",POSITION="
      REWIND")
      READ(20,'(2F16.10)') ZMAX,ZMIN
      READ(20,'(2I16)') NFX,NFY
      ALLOCATE(A(NFX,NFY),B(NFX,NFY),C(NFX,NFY),D(NFX,NFY),X0(NFX+1),&
            Y0(NFY+1),Z0X(NFX+1),Z0Y(NFY+1))
      DO I=1,NFX
            DO J=1,NFY
                  READ(20,'(4F16.10)') A(I,J),B(I,J),C(I,J),D(I,J)
            END DO
      END DO
!READ IN INFORMATION ON OCEAN HEIGHTS
      OPEN(UNIT=10,FILE="OCEAN_HEIGHT.TXT",ACTION="READ",POSITION="REWI
      ND")
      PRINT*,NFX,NFY
      DO I=1,NFX+1
            READ(10,'(2F16.10)') X0(I), Z0X(I)
      END DO

      DO J=1,NFY+1
            READ(10,'(2F16.10)') Y0(J), Z0Y(J)
      END DO
!READ IN INFORMATION ABOUT THE SHIP
      OPEN(UNIT=30,FILE="SHIP_FACETS.TXT",ACTION="READ",POSITION="REWIN
      D")
      PRINT *,'READING SHIP GEOMETRY.....'
      READ(30,'(1X,I4)') NSF
      PRINT*,NSF
      ALLOCATE(FACET(NSF),AS(NSF),BS(NSF),CS(NSF),DS(NSF),X0S(NSF),Y0S(
      NSF),Z0S(NSF),V1(NSF,3),V2(NSF,3))
      DO SFN=1,NSF
```

```
            READ(30,'(1X,7F16.8)')
AS(SFN),BS(SFN),CS(SFN),DS(SFN),X0S(SFN),&
                  Y0S(SFN),Z0S(SFN)
            READ(30,'(1X,3F16.8)') V1(SFN,:),V2(SFN,:)
      END DO
      PI=DACOS(DBLE(-1))
!ENTER THE TYPE OF RAY TRACE
      PRINT *,"ENTER THE TYPE OF RAY TRACE:"
      PRINT *,"ENTER (1) FOR REFLECTION"
      PRINT *,"ENTER (2) FOR EMISION"
      READ *,TTYPE
!ENTER IN INFORMATION ABOUT THE SHIP
      PRINT *,"ENTER THE REFLECTIVITY OF THE SHIP"
      READ *,RSHIP
      PRINT *,"ENTER THE SPECULARITY OF THE SHIP"
      READ *,SPECULAR
!ENTER THE # OF RAYS TO BE EMMITED AND VIEWING ANGLE
      PRINT *,"ENTER THE ZENITH ANGLE (RELATIVE TO Z-AXIS)"
      READ *,THETAVIEW
      THETAVIEW=THETAVIEW*PI/180
      PRINT *,"ENTER THE AZIMUTH ANGLE RELATIVE TO THE X-AXIS"
      READ *,PHIVIEW
      PHIVIEW=PHIVIEW*PI/180
      PRINT *,"ENTER THE # OF RAYS IN ONE DIRECTION"
      READ *,RAYSX
      RAYSY=RAYSX
      TRAYS=RAYSX*RAYSY !TOTAL NUMBER OF RAYS
      TRACE=RAYSX/6
      TCOUNT=0
      ALLOCATE(XSCREEN(TRAYS),YSCREEN(TRAYS),ZSCREEN(TRAYS),
      THETAR(TRAYS),PHIR(TRAYS),HPOWER(TRAYS),VPOWER(TRAYS),
      REFLECTION(TRAYS),TEMP(1),XOUT(TRAYS),YOUT(TRAYS),SURFACE(TRAYS),
      XSCREEN2(TRAYS),YSCREEN2(TRAYS),HPOWER2(TRAYS),VPOWER2(TRAYS))
      REFLECTION=0
!EMISION ORIGIN
      ODIST=15000 !STRAIGHT-LINE DISTANCE FROM THE CENTER OF THE OCEAN
      XE=ODIST*DCOS(PHIVIEW)*DCOS(THETAVIEW)
      YE=ODIST*DSIN(PHIVIEW)*DCOS(THETAVIEW)
      ZE=ODIST*DSIN(THETAVIEW)
      VBS=(/-XE+175*DCOS(PHIVIEW),-YE+175*DSIN(PHIVIEW),0-ZE/)
!VECTOR TO THE BOTTOM OF THE SCREEN
      VBSMAG=DSQRT(DOT_PRODUCT(VBS,VBS)) !MAGNITUDE OF VBS VECTOR
      VCS=(/0-XE,0-YE,0-ZE/)!VECTOR TO CENTER OF SCREEN
      VCSMAG=DSQRT(DOT_PRODUCT(VCS,VCS)) !MAGNITUDE OF VBS VECTOR
      VBS=VBS/VBSMAG !CONVERT TO UNIT VECTOR
      VCS=VCS/VCSMAG !CONVERT TO UNIT VECTOR
      THETAS=DACOS(DOT_PRODUCT(VBS,VCS)) !ANGLE BETWEEN VCS AND VBS
      DTHETA=2*THETAS/(RAYSX-1) !CHANGE IN ANGLE BETWEEN RAYS
!SCREEN LOCATION (5000M FROM EMISSION ORIGIN)
      DIST=5000
!CENTER OF SCREEN
      S_CENTER=(/XE+DIST*VCS(1),YE+DIST*VCS(2),ZE+DIST*VCS(3)/)
!VALUE "D" FOR THE EQUATION OF THE SCREEN
      D_SCREEN=-DOT_PRODUCT(VCS,S_CENTER)
!WATER PROPERTIES
      IF (TTYPE==1) THEN !COMPLEX INDEX OF REFRACTION FOR 550nm
            MW=(1.339,1.86E-9)
```

```
      ELSE !COMPLEX INDEX OF REFRACTION FOR 1100nm
            MW=(1.153,.0968)
      END IF


      RAY=0
      CHECK=1
      ROW=0
! CREATE AXIS OF ROTATION FOR CHANGING THE DIRECTION OF EMISSION
! VERTICALLY ON THE SCREEN
      R_AXIS1=(/DCOS(PHIVIEW+90*PI/180),DSIN(PHIVIEW+90*PI/180),DBLE(0)
      /)
      TEMP2=DSQRT(DOT_PRODUCT(R_AXIS1,R_AXIS1))
      R_AXIS1=R_AXIS1/TEMP2 !CONVERT TO UNIT VECTOR
      OPEN(UNIT=50,FILE="ACAD_RAYS.TXT",ACTION="WRITE",POSITION="REWIND
      ")
      DO RNY=1,RAYSY
!FIND THE VERTICAL ANGLE TO EMMIT, RELATIVE TO THE CENTER OF THE SCREEN
            THETAY=-(RNY-1)*DTHETA+THETAS
!ROTATE TO THE CORRECT VERTICAL LOCATION ON THE SCREEN
            X2=VCS(1)*(DCOS(THETAY)+R_AXIS1(1)**2*(1-DCOS(THETAY)))
            +VCS(2)*(-R_AXIS1(3)*DSIN(THETAY)+R_AXIS1(1)*R_AXIS1(2)*
            (1-DCOS(THETAY)))+VCS(3)*(R_AXIS1(2)*DSIN(THETAY)+
            R_AXIS1(1)*R_AXIS1(3)*(1-DCOS(THETAY)))

            Y2=VCS(1)*(R_AXIS1(3)*DSIN(THETAY)+R_AXIS1(1)*R_AXIS1(2)*
            (1-DCOS(THETAY)))+VCS(2)*(DCOS(THETAY)+R_AXIS1(2)**2*
            (1-DCOS(THETAY)))+VCS(3)*(-R_AXIS1(1)*DSIN(THETAY)+
            R_AXIS1(2)*R_AXIS1(3)*(1-DCOS(THETAY)))

            Z2=VCS(1)*(-R_AXIS1(2)*DSIN(THETAY)+R_AXIS1(1)*R_AXIS1(3)*
            (1-
            DCOS(THETAY)))+VCS(2)*(R_AXIS1(1)*DSIN(THETAY)+R_AXIS1(2)*
            R_AXIS1(3)*(1-DCOS(THETAY)))+VCS(3)*(DCOS(THETAY)+
            R_AXIS1(3)**2*(1-DCOS(THETAY)))
!VECTOR LOCATING MIDPOINT OF THE SCREEN AT A GIVEN VERTICAL LOCATION
            VCSR=(/X2,Y2,Z2/)
!CREATE AXIS OF ROTATION FOR CHANGING THE DIRECTION OF EMISSION
! HORIZONTALLY ON THE SCREEN
            R_AXIS=CROSS(VCSR,R_AXIS1)
            TEMP2=DSQRT(DOT_PRODUCT(R_AXIS,R_AXIS))
            R_AXIS=R_AXIS/TEMP2 !CONVERT TO UNIT VECTOR
            PRINT *,RAYSY-RNY
            DO RNX=1,RAYSX
                  RAY=RAY+1
!FIND HORIZONTAL ANGLE TO EMIT, RELATIVE TO THE CENTER OF THE SCREEN
                  THETAX=(RNX-1)*DTHETA-THETAS
                  TCOUNT=TCOUNT+1
                  IF (TCOUNT==TRACE.AND.ROWT==TRACE) THEN
                        WRITE(50,'(A4)') "LINE"
                  END IF
!SET THE INITIAL POWER OF EACH RAY FOR EACH POLARIZAION
                  PPOWER=.5
                  SPOWER=.5

!FIND DIRECTION COSINES
                  LE=X2*(DCOS(THETAX)+R_AXIS(1)**2*(1-DCOS(THETAX)))+&
```

```
                  Y2*(-R_AXIS(3)*DSIN(THETAX)+R_AXIS(1)*R_AXIS(2)*
                  (1-DCOS(THETAX)))+Z2*(R_AXIS(2)*DSIN(THETAX)+
                  R_AXIS(1)* R_AXIS(3)*(1-DCOS(THETAX)))
                  ME=X2*(R_AXIS(3)*DSIN(THETAX)+R_AXIS(1)*R_AXIS(2)*
                  (1-DCOS(THETAX)))+Y2*(DCOS(THETAX)+R_AXIS(2)**2*
                  (1-DCOS(THETAX)))+Z2*(-R_AXIS(1)*DSIN(THETAX)
                  +R_AXIS(2)*R_AXIS(3)*(1-DCOS(THETAX)))

                  NE=X2*(-R_AXIS(2)*DSIN(THETAX)+R_AXIS(1)*R_AXIS(3)
                  *(1-DCOS(THETAX)))+Y2*(R_AXIS(1)*DSIN(THETAX)
                  +R_AXIS(2)*R_AXIS(3)*(1-DCOS(THETAX)))+Z2*
                  (DCOS(THETAX)+R_AXIS(3)**2*(1-DCOS(THETAX)))

                  TEMP2=SQRT(LE**2+ME**2+NE**2)
                  LE=LE/TEMP2
                  ME=ME/TEMP2
                  NE=NE/TEMP2
!FIND THE DISTANCE FROM THE POINT OF EMISSION TO THE SCREEN
                  T_SCREEN=(VCS(1)*XE+VCS(2)*YE+VCS(3)*ZE+D_SCREEN)/&
                  (-VCS(1)*LE-VCS(2)*ME-VCS(3)*NE)
!FIND THE LOCATION WHERE THE RAY PASSES THROUGH THE SCREEN
                  XSCREEN(RAY)=XE+LE*T_SCREEN
                  YSCREEN(RAY)=YE+ME*T_SCREEN
                  ZSCREEN(RAY)=ZE+NE*T_SCREEN
                  XS=XSCREEN(RAY)
                  YS=YSCREEN(RAY)
                  ZS=ZSCREEN(RAY)
                  VE=(/LE,ME,NE/) !INITIAL DIRECTION OF THE RAY
                  ZAXIS=(/0,0,1/)
                  IF (TCOUNT==TRACE.AND.ROWT==TRACE) THEN
                        WRITE(50,'(F10.3,",",F10.3,",",F10.3)')
                        8800*LE+XS,8800*ME+YS,8800*NE+ZS
                  END IF
!FIND THE INITIAL DIRECTION OF THE HORIZONTAL POWER VECTOR
                  VP(1,:)=CROSS(VE,ZAXIS)
                  VPMAG=DSQRT(VP(1,1)**2+VP(1,2)**2+VP(1,3)**2)
                  VP(1,:)=VP(1,:)/VPMAG
!FIND THE INITIAL DIRECTION OF THE HORIZONTAL POWER VECTOR
                  VS(1,:)=CROSS(VP(1,:),VE)
                  VSMAG=DSQRT(VS(1,1)**2+VS(1,2)**2+VS(1,3)**2)
                  VS(1,:)=VS(1,:)/VSMAG
                  FREERAY=0
                  DO WHILE (FREERAY==0)
                        CHECK=0
                        T=99999999
!FIND RANGE OF VALUES FOR THE INTERSECTION POINT WITH THE OCEAN SURFACE
                        T1=(ZMAX-ZS)/NE
!DISTANCE TO ZMAX (HIGHEST POINT OF OCEAN SURFACE)
                        IF (T1>1E-10) THEN
                              XRANGE1=XS+LE*T1
                              YRANGE1=YS+ME*T1
                        ELSE
                              XRANGE1=XS
                              YRANGE1=YS
                        END IF
                        T2=(ZMIN-ZS)/NE
!DISTANCE TO ZMIN (LOWEST POINT OF OCEAN SURFACE)
```

```
                        IF (T2>1E-10) THEN
                              XRANGE2=XS+LE*T2
                              YRANGE2=YS+ME*T2
                        ELSE
                              XRANGE2=XS
                              YRANGE2=YS
                        END IF
!DETERMINE MAX AND MIN VALUES OF RANGE
                        IF (XRANGE1>XRANGE2) THEN
                              MAXXV=XRANGE1
                              MINXV=XRANGE2
                        ELSE
                              MAXXV=XRANGE2
                              MINXV=XRANGE1
                        END IF
                        IF (YRANGE1>YRANGE2) THEN
                              MAXYV=YRANGE1
                              MINYV=YRANGE2
                        ELSE
                              MAXYV=YRANGE2
                              MINYV=YRANGE1
                        END IF
!GET THE FACET RANGE FOR THE VALUES JUST CALCULATED
                        XF=2
                        DO WHILE ((X0(XF)<MINXV).AND.(XF.NE.NFX+1))
                              XF=XF+1
                        END DO
                        MINXF=XF-1
                        XF=2
                        DO WHILE ((X0(XF)<MAXXV).AND.(XF.NE.NFX+1))
                              XF=XF+1
                        END DO
                        MAXXF=XF-1
                        YF=2
                        DO WHILE ((Y0(YF)<MINYV).AND.(YF.NE.NFY+1))
                              YF=YF+1
                        END DO
                        MINYF=YF-1
                        YF=2
                        DO WHILE ((Y0(YF)<MAXYV).AND.(YF.NE.NFY+1))
                              YF=YF+1
                        END DO
                        MAXYF=YF-1
!FIND FIRST FORWARD CANDIDATE
                        DO XF=MINXF,MAXXF
                              DO YF=MINYF,MAXYF
!FIND DISTANCE TO INTERSECTION POINT WITH FACET (XF,YF)
                                    TI=(A(XF,YF)*XS+B(XF,YF)*YS+
                                    C(XF,YF)*ZS+D(XF,YF))/(-
                                    A(XF,YF)*LE-B(XF,YF)*ME-
                                    C(XF,YF)*NE)
!CHECK TO SEE IF INTERSECTION POINT IS ON THE FACET
                                    IF (TI>1E-10.AND.TI<T) THEN
                                          XI=XS+LE*TI
!X-COORDINATE OF INTERSECTION POINT
                                          IF (XI>=X0(XF).AND.
                                          XI<=X0(XF+1)) THEN
```

```
                                                YI=YS+ME*TI
!Y-COORDINATE OF INTERSECTION POINT
                                                IF (YI>=Y0(YF).AND.
                                                YI<=Y0(YF+1)) THEN
!DEFINE INTERSECTION POINT AND NORMAL VECTOR
                                                      XP=XI
                                                      YP=YI
                                                      ZP=ZS+NE*TI
                                                      N=(/A(XF,YF),B(XF
                                                      ,YF), C(XF,YF)/)
                                                      T=TI
                                                      CHECK=1
                                                END IF
                                          END IF
                                     END IF
                                END DO
                           END DO
!CHECK SHIP FOR INTERSECTIONS
                           DO SFN=1,NSF
!FIND THE DISTANCE TO THE POINT OF INTERSECTION WITH THE FACET
                                TI=(AS(SFN)*XS+BS(SFN)*YS+CS(SFN)
                                *ZS+DS(SFN))/(-AS(SFN)*LE-BS(SFN)*ME-
                                CS(SFN)*NE)
                                IF (TI>1E-10.AND.TI<T) THEN
!THE COORDINATES OF THE POINT OF INTERSECTION ARE...
                                     XI=XS+LE*TI
                                     YI=YS+ME*TI
                                     ZI=ZS+NE*TI
!DETERMINE THE DOMINANT PLANE

                                     TEMP=MAXLOC((/DABS(AS(SFN)),DABS(BS
                                     (SFN)),DABS(CS(SFN))/))
                                     PLANE=TEMP(1)
                                     IF (PLANE==1) THEN
!DOMINANT PLANE IS YZ
                                          U0=YI-Y0S(SFN)
                                          U1=V1(SFN,2)
                                          U2=V2(SFN,2)
                                          W0=ZI-Z0S(SFN)
                                          W1=V1(SFN,3)
                                          W2=V2(SFN,3)
                                     ELSE IF (PLANE==2) THEN
!DOMINANT PLANE IS XZ
                                          U0=ZI-Z0S(SFN)
                                          U1=V1(SFN,3)
                                          U2=V2(SFN,3)
                                          W0=XI-X0S(SFN)
                                          W1=V1(SFN,1)
                                          W2=V2(SFN,1)
                                     ELSE
!DOMINANT PLANE IS XY
                                          U0=XI-X0S(SFN)
                                          U1=V1(SFN,1)
                                          U2=V2(SFN,1)
                                          W0=YI-Y0S(SFN)
                                          W1=V1(SFN,2)
                                          W2=V2(SFN,2)
```

```
                                    END IF
!CALCULATE ALPHA AND BETA FOR THE FACET
                                    BETA=(W1*U0-U1*W0)/(W1*U2-U1*W2)
                                    IF (W1==0) THEN
                                          ALPHA=(U0-BETA*U2)/U1
                                    ELSE
                                          ALPHA=(W0-BETA*W2)/W1
                                    END IF
!DETERMINE IF THE INTERSECTION POINT LIES ON THE FACET
                                    IF (ALPHA>=1E-9.AND.BETA>=1E-9
                                    .AND.ALPHA+BETA<=1) THEN
!THE COORDINATES OF THE POINT OF INTERSECTION ARE...
                                          XP=XI
                                          YP=YI
                                          ZP=ZI
                                          N=(/AS(SFN),BS(SFN),CS(SFN)/)
!NORMAL VECTOR
                                          NMAG=DSQRT(DOT_PRODUCT(N,N))
                                          N=N/NMAG
!CONVERT TO UNIT VECTOR
                                          TN1=V1(SFN,:)/DSQRT
                                          (DOT_PRODUCT (V1(SFN,:),
                                          V1(SFN,:)))
!TANGENT VECTOR
                                          TN2=CROSS(N,TN1)
!TANGENT VECTOR
                                          T=TI
                                          CHECK=2
                                    END IF
                              END IF
                        END DO
!CHECK=0 MEANS THE RAY INTERSECTS WITH THE SKY
!CHECK=1 MEANS THE RAY INTERSECTS WITH THE OCEAN
!CHECK=2 MEANS THE RAY INTERSECTS WITH THE SHIP
                        IF (CHECK==0) THEN
                              IF (REFLECTION(RAY)==0) THEN
                                    SURFACE(RAY)=0
                              ELSE
                                    SURFACE(RAY)=1
                              END IF
                        ELSE IF (CHECK==1) THEN
                              SURFACE(RAY)=2
                        ELSE IF (CHECK==2) THEN
                              SURFACE(RAY)=3
                        END IF


!DETERMINE REFLECTION
                        IF (CHECK>0) THEN
                              VI=(/LE,ME,NE/)
!DIRECTION OF INCIDENT RAY
                              VR=(VI-2*DOT_PRODUCT(VI,N)*N)
!DIRECTION OF REFLECTED RAY
                              VRMAG=DSQRT(DOT_PRODUCT(VR,VR))
                              VR=VR/VRMAG !CONVERT TO UNIT VECTOR
                              XS=XP
                              YS=YP
```

```
                              ZS=ZP
                              IF (TCOUNT==TRACE.AND.ROWT==TRACE) THEN
                                    WRITE(50,'(F10.3,",",F10.3,",",F10.
                                    3)') XS,YS,ZS
                              END IF
                              REFLECTION(RAY)=REFLECTION(RAY)+1
!ROTATE THE POWER VECTORS TO LINE UP WITH THE FACET
                              VECT=2*REFLECTION(RAY)
                              VP(VECT,:)=CROSS(VI,N)
!DIRECTION OF POWER VECTOR 'P' AFTER ROTATION
                              VPMAG=DSQRT(VP(VECT,1)**2+VP(VECT,2)**2+
                              VP(VECT,3)**2)
                              VP(VECT,:)=VP(VECT,:)/VPMAG
                              VS(VECT,:)=CROSS(VP(VECT,:),VI)
!DIRECTION OF POWER VECTOR 'S' AFTER ROTATION
                              VSMAG=DSQRT(VS(VECT,1)**2+VS(VECT,2)**2+
                              VS(VECT,3)**2)
                              VS(VECT,:)=VS(VECT,:)/VSMAG
!FIND THE POWER OF EACH VECTOR AFTER ROTATION
                              PPOWER=DSQRT((PPOWER*DOT_PRODUCT(VP(VECT-
                              1,:),VP(VECT,:)))**2+(SPOWER*
                              DOT_PRODUCT(VS(VECT-1,:),VP(VECT,:)))**2)
                              SPOWER=DSQRT((SPOWER*
                              DOT_PRODUCT(VS(VECT-1,:),VS(VECT,:)))**2
                              +(PPOWER* DOT_PRODUCT(VP(VECT-1,:)
                              ,VS(VECT,:)))**2)
!DETERMINE THE REFLECTIVITY
                              IF (CHECK==1) THEN
                                    COSTHETA=DABS(DOT_PRODUCT(VI,N))
                                    COSTHETAT=SQRT
                                    (1-((DSIN(THETA))/MW)**2)
                                    RP=ABS((MW*COSTHETA-COSTHETAT)/
                                    (MW*COSTHETA+COSTHETAT))**2
                                    RS=ABS((COSTHETA-MW*COSTHETAT)/
                                    (COSTHETA+MW*COSTHETAT))**2
                              ELSE IF (CHECK==2) THEN
                                    RP=RSHIP*.5
                                    RS=RSHIP*.5
                                    CALL RANDOM_NUMBER(RTYPE)
                                    IF (RTYPE>SPECULAR) THEN
!REFLECTION IS DIFFUSE
                                          CALL RANDOM_NUMBER(RANDOM)
                                          PHID=2*PI*RANDOM
                                          CALL RANDOM_NUMBER(RANDOM)
                                          THETAD=DASIN(SQRT(RANDOM))
                                          VR=N*DCOS(THETAD)+TN1*
                                          DSIN(THETAD)*DCOS(PHID)+
                                          TN2*DSIN(THETAD)*DSIN(PHID)
!DIRECTION OF DIFFUSE REFLECTION
                                    END IF
                              END IF
                              IF (TTYPE==2) THEN
!FOR AN INFRARED RAY TRACE
!CHECK IF RAY IS ABSORBED OR REFLECTED
                                    CALL RANDOM_NUMBER(RTYPE)
                                    IF (RTYPE>RP+RS) THEN
!RAY IS ABSORBED
```

```
                                        PPOWER=PPOWER*(.5-RP)
                                        SPOWER=SPOWER*(.5-RS)
                                        CHECK=0
                                        ELSE !RAY IS REFLECTED
                                        PPOWER=PPOWER*RP
                                        SPOWER=SPOWER*RS
                                        LE=VR(1)
                                        ME=VR(2)
                                        NE=VR(3)
                                        VP(VECT+1,:)=VP(VECT,:)
                                        VS(VECT+1,:)=CROSS
                                        (VR,VP(VECT+1,:))
                                        TEMP2=(PPOWER+
                                        SPOWER+.000000001)
                                        PPOWER=PPOWER/TEMP2
                                        SPOWER=SPOWER/TEMP2
                                  END IF
                             ELSE !FOR A VISIBLE LIGHT RAY TRACE
                                  PPOWER=PPOWER*RP
                                  SPOWER=SPOWER*RS
                                  LE=VR(1)
                                  ME=VR(2)
                                  NE=VR(3)
                                  VP(VECT+1,:)=VP(VECT,:)
                                  VS(VECT+1,:)=CROSS(VR,VP(VECT+1,:))
                             END IF
                        END IF

                        IF (CHECK==0) THEN !RAY INTERSECTS WITH THE SKY
                             FREERAY=1
!FIND AND RECORD THE DIRECTION OF THE RAY
                             THETAR(RAY)=ATAN(DSQRT(LE**2+ME**2)/NE)
                             PHIR(RAY)=ATAN2(ME,LE)
                             IF (TCOUNT==TRACE.AND.ROWT==TRACE) THEN
                                  WRITE(50,'(F10.3,",",F10.3,",",F10.
                                  3/)') XS+100*LE,YS+100*ME,ZS+100*NE
                             END IF
                             IF (TCOUNT==TRACE) THEN
                                  TCOUNT=0
                             END IF
!ROTATE THE SCREEN SO THAT THE LOCAL X-AXIS IS IN THE SAME DIRECTION AS
THE GLOBAL X-AXIS
                             XSCREEN2(RAY)=XSCREEN(RAY)*DSIN(PHIVIEW)-
                             YSCREEN(RAY)*DCOS(PHIVIEW)

                             YSCREEN2(RAY)=XSCREEN(RAY)*DCOS(PHIVIEW)+
                             YSCREEN(RAY)*DSIN(PHIVIEW)
                             IF (RAY==1) THEN
                                  XS1=XSCREEN2(RAY)
                                  YS1=YSCREEN2(RAY)
                                  ZS1=ZSCREEN(RAY)
                             END IF
!TRANSLATE THE SCREEN SO THE BOTTOM LEFT CORNER IS AT (0,0,0)
                             XSCREEN2(RAY)=XSCREEN2(RAY)-XS1
                             YSCREEN2(RAY)=YSCREEN2(RAY)-YS1
                             ZSCREEN(RAY)=ZSCREEN(RAY)-ZS1
!ROTATE THE SCREEN ABOUT THE X-AXIS SO THAT IT LIES ON THE XY PLANE
```

```
                              XOUT(RAY)=XSCREEN2(RAY)
                              YOUT(RAY)=-DSQRT((YSCREEN2(RAY))**2
                              +(ZSCREEN(RAY))**2)
                        END IF
                        IF (NE<0) THEN
!RAY DOES NOT INTERSECT WITH ANY SURFACES
                              THETAR(RAY)=0
                              PHIR(RAY)=0
                              IF (SURFACE(RAY)==0) THEN
                                    PPOWER=0
                                    SPOWER=0
                              END IF
                        END IF
                  END DO!END WHILE
! ROTATE POWER VECTORS SO THEY ARE HORIZONTAL AND VERTICAL RELATIVE TO
! THE GLOBAL COORDINATES

                  HPOWER(RAY)=DSQRT((PPOWER*DOT_PRODUCT(VP(2,:),VP(1,:)
                  ))**2+(SPOWER*DOT_PRODUCT(VS(2,:),VP(1,:)))**2)

                  VPOWER(RAY)=DSQRT((SPOWER*DOT_PRODUCT(VS(2,:),VS(1,:)
                  ))**2+(PPOWER*DOT_PRODUCT(VP(2,:),VS(1,:)))**2)
            END DO
      END DO
      PRINT *,'WRITING.....'
!WRITE ALL INFORMATION ABOUT THE RAY TRACE
      OPEN(UNIT=40,FILE="RAY_INFO.TXT",ACTION="WRITE",POSITION="REWIND"
      )
      WRITE(40,'(I2)') TTYPE
      WRITE(40,'(I16)') TRAYS
      DO RAY=1,TRAYS
            WRITE(40,'(7F16.8)')SURFACE(RAY),THETAR(RAY),PHIR(RAY),
            HPOWER(RAY),VPOWER(RAY),XOUT(RAY),YOUT(RAY)
      END DO
END PROGRAM MCRT

      FUNCTION CROSS(A,B)
            IMPLICIT NONE
            DOUBLE PRECISION, DIMENSION(3)::CROSS
            DOUBLE PRECISION, DIMENSION(3), INTENT(IN)::A,B
            CROSS=(/A(2)*B(3)-A(3)*B(2),A(3)*B(1)-A(1)*B(3),A(1)*B(2)-
            A(2)*B(1)/)
      END FUNCTION CROSS
```

Appendix D

```
PROGRAM POST
      IMPLICIT NONE
      DOUBLE PRECISION::LSY,LSX,DXS,DYS,THETASUN,PHISUN,TS,PI,
      COSDPHI,PSI,SKY,H,K,C,WL,DWL
      INTEGER::RAY,TRAYS,SEG,XSEG,YSEG,I,J,TTYPE
      DOUBLE PRECISION,DIMENSION(:),ALLOCATABLE::
      THETA,PHI,HPOWER,VPOWER,XS,YS,SURFACE,SPOWER,VSUN,VRAY
      DOUBLE PRECISION, DIMENSION(:,:),ALLOCATABLE::
      POWERH,POWERV,POWERT
      INTEGER, DIMENSION(:,:),ALLOCATABLE::RAYCOUNT
      INTEGER, DIMENSION(:),ALLOCATABLE::SURF

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! LSY = Length of the screen in the y-direction
! LSX = Length of the screen in the x-direction
! DXS = Size of a pixel in the x-direction
! DYS = Size of a pixel in the y-direction
! THETASUN = Zenith angle of the sun
! PHISUN = Azimuth angle of the sun
! TS = Temperature of a surface
! PI = 3.14
! COSDPHI = cos(PSI)
! PSI = Angle of a ray relative to the sun
! SKY = Sky power function
! H = Parameter for blackbody radiation distribution function
! K = Parameter for blackbody radiation distribution function
! C = Parameter for blackbody radiation distribution function
! WL = wavelength of interest
! DWL = Amplitude of waveband
! RAY = Incrementing variable
! TRAYS = Total number of rays
! SEG = # of pixels in each direction
! XSEG = Specifies the column of a pixel
! YSEG = Specifies the row of a pixel
! I = Incrementing variable
! J = Incrementing variable
! TTYPE = specifies type of ray trace
! THETA = Final zenith angle of all rays
! PHI = Final Azimuth angle of all rays
! HPOWER = Horizontal polarization power of all rays
! VPOWER = Vertical polarization power of all rays
! XS = x-location on screen
! YS = y-location on screen
! SURFACE = Specifies a surface
! SPOWER = Specifies the power output of each surface
! VSUN = vector locating the sun
```

```
! VRAY = vector representing the ray
! POWERH = Horizontal polarization power of all pixels
! POWERV = Vertical polarization power of all pixels
! POWERT = Sum of both polarizations
! RAYCOUNT = Counting variable
! SURF = surface (integer)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


      PRINT *,'READING.....'
      OPEN(UNIT=10,FILE="RAY_INFO.TXT",ACTION="READ",POSITION="REWIND")
      READ(10,'(I2)') TTYPE
      READ(10,'(I16)') TRAYS
      ALLOCATE(THETA(TRAYS),PHI(TRAYS),HPOWER(TRAYS),VPOWER(TRAYS),&
            XS(TRAYS),YS(TRAYS),SURFACE(TRAYS),SURF(TRAYS),SPOWER(3),&
            VSUN(3),VRAY(3))
      DO RAY=1,TRAYS
      READ(10,'(7F16.8)')SURFACE(RAY),THETA(RAY),PHI(RAY),HPOWER(RAY),&
            VPOWER(RAY),XS(RAY),YS(RAY)
      END DO

      SURF=SURFACE
!PARAMETERS USED FOR TH BLACKBODY DISTRIBUTION FUNCTION
      PI=DACOS(DBLE(-1))
      H=6.626E-34
      K=1.381E-23
      C=2.998E8
!PROMPT USER TO ENTER THE NUMBER OF PIXELS
      PRINT *,"ENTER # OF PIXELS IN EACH DIRECTION"
      READ *,SEG
!DETERMINE THE TYPE OF RAY TRACE THAT WAS PERFORMED
      IF (TTYPE==1) THEN !PROMPT USER TO ENTER THE LOCATION OF THE SUN
            PRINT *,"ENTER THE ZENITH ANGLE OF THE SUN (THETA)"
            READ *,THETASUN
            THETASUN=THETASUN*PI/180
            PRINT *,"ENTER THE ORIENTATION ANGLE OF THE SUN (PHI)"
            READ *,PHISUN
            PHISUN=PHISUN*PI/180
            VSUN=(/DSIN(THETASUN)*DCOS(PHISUN),DSIN(THETASUN)*
            DSIN(PHISUN),DCOS(THETASUN)/)
      ELSE IF (TTYPE==2) THEN
!PROMPT USER TO ENTER THE TEMPERATURE OF ALL SURFACES
            WL=11E-6 !MIDDLE OF WAVELENGTH BAND
            DWL=1E-6 !HALF THE WIDTH OF WAVELENGTH BAND
! INTEGRATE THE BLACKBODY RADIATION DISTRIBUTION FUNCTION OVER THE
! WAVEBAND DEFINED ABOVE
            PRINT *,"ENTER THE TEMPERATURE OF THE SKY"
            READ *,TS
!CALCULATE THE INITIAL POWER OF A RAY EMITTED BY THE SKY
            SPOWER(1)=((2*PI*H*C**2)/((WL+DWL)**5*
            (EXP(H*C/((WL+DWL)*K*TS))-1))+(2*PI*H*C**2)/
            ((WL-DWL)**5*(EXP(H*C/((WL-DWL)*K*TS))-1)))*(2*DWL)
            PRINT *,"ENTER THE TEMPERATURE OF THE SHIP"
            READ *,TS
!CALCULATE THE INITIAL POWER OF A RAY EMITTED BY THE SHIP
            SPOWER(2)=((2*PI*H*C**2)/((WL+DWL)**5*
            (EXP(H*C/((WL+DWL)*K*TS))-1))+(2*PI*H*C**2)/
```

```
            ((WL-DWL)**5*(EXP(H*C/((WL-DWL)*K*TS))-1)))*(2*DWL)
            PRINT *,"ENTER THE TEMPERATURE OF THE OCEAN SURFACE"
            READ *,TS
!CALCULATE THE INITIAL POWER OF A RAY EMITTED BY THE OCEAN
            SPOWER(3)=((2*PI*H*C**2)/((WL+DWL)**5*
            (EXP(H*C/((WL+DWL)*K*TS))-1))+(2*PI*H*C**2)/
            ((WL-DWL)**5*(EXP(H*C/((WL-DWL)*K*TS))-1)))*(2*DWL)
      END IF

      PRINT *,'PROCESSING.....'
!FIND THE DIMENSIONS OF THE SCREEN AND THE SIZE OF EACH PIXEL
      XS=XS-MINVAL(XS)
      YS=YS-MINVAL(YS)
      LSX=MAXVAL(XS)+1E-9 !SCREEN WIDTH
      LSY=MAXVAL(YS)+1E-9 !SCREEN HEIGHT
      DXS=LSX/SEG !PIXEL WIDTH
      DYS=LSY/SEG !PIXEL HEIGHT

      ALLOCATE(RAYCOUNT(SEG,SEG),POWERH(SEG,SEG),POWERV(SEG,SEG),
      POWERT(SEG,SEG))
      RAYCOUNT=0
!SET THE INITIAL POWER OF EACH PIXEL
      POWERH=0
      POWERV=0
      POWERT=0

      DO RAY=1,TRAYS
!DETERMINE THE CORRECT PIXEL THAT THE RAY PASSES THROUGH
            XSEG=FLOOR(XS(RAY)/DXS)+1
            YSEG=FLOOR(YS(RAY)/DYS)+1
            IF (XSEG<=SEG.AND.YSEG<=SEG.AND.SURFACE(RAY)>0) THEN
                  RAYCOUNT(XSEG,YSEG)=RAYCOUNT(XSEG,YSEG)+1
                  IF (TTYPE==1) THEN
!DETERMINE THE INITIAL POWER OF THE RAYS FOR VISIBLE LIGHT

                        VRAY=(/DSIN(THETA(RAY))*DCOS(PHI(RAY)),
                        DSIN(THETA(RAY))*DSIN(PHI(RAY)),DCOS(THETA(RAY)
                        )/)
                        COSDPHI=(DOT_PRODUCT(VSUN,VRAY))
                        PSI=DACOS(DCOS(THETA(RAY))*DCOS(THETASUN)+DSIN(
                        THETA(RAY))*DSIN(THETASUN)*COSDPHI)
                        IF (PSI*180/PI>.27) THEN
!FIND THE POWER OF THE SKY AT ANGLE PSI AWAY FROM THE SUN
                              SKY=.15+.0379*PSI**(-0.7491)
                        ELSE
                              SKY=0
                              RAYCOUNT(XSEG,YSEG)=RAYCOUNT(XSEG,YSEG)-1
                        END IF
                        SPOWER(1)=SKY
                  END IF
!ADD THE FINAL POWER OF THE RAY TO THE CORRECT PIXEL

                  POWERH(XSEG,YSEG)=POWERH(XSEG,YSEG)+HPOWER(RAY)*
                  SPOWER(SURF(RAY))

                  POWERV(XSEG,YSEG)=POWERV(XSEG,YSEG)+VPOWER(RAY)*
                  SPOWER(SURF(RAY))
```

```
            END IF
      END DO
!TOTAL POWER IS THE SUM OF BOTH POLARIZATIONS
      POWERT=POWERV+POWERH

      PRINT *,'WRITING.....'

      OPEN(UNIT=20,FILE="SCREEN_XY.TXT",ACTION="WRITE",POSITION="REWIND
      ")
      OPEN(UNIT=30,FILE="SCREEN_POWERH.TXT",ACTION="WRITE",POSITION="RE
      WIND")
      OPEN(UNIT=40,FILE="SCREEN_POWERV.TXT",ACTION="WRITE",POSITION="RE
      WIND")
      OPEN(UNIT=50,FILE="SCREEN_POWERT.TXT",ACTION="WRITE",POSITION="RE
      WIND")
! WRITE THE POWER OF EACH PIXEL FOR EACH POLARIZATION AND FOR
! UNPOLARIZED
      DO I=1,SEG
            WRITE(20,'(1X, 2F16.6)') (I-1)*DXS,(I-1)*DYS
            WRITE(30,'(1X, 200F16.6)')
            (POWERH(I,J)/(RAYCOUNT(I,J)+.00000001),J=1,SEG)
            WRITE(40,'(1X, 200F16.6)')
            (POWERV(I,J)/(RAYCOUNT(I,J)+.00000001),J=1,SEG)
            WRITE(50,'(1X, 200F16.6)')
            (POWERT(I,J)/(RAYCOUNT(I,J)+.00000001),J=1,SEG)
      END DO

END PROGRAM POST
```

VITA

Joseph Maniscalco was born November 10, 1978 in Queens, NY. He is the youngest of four siblings and the first in his family to pursue engineering as a career. Joseph became interested in engineering at a very young age, making contraptions and inventions from commonly used household items. This ability was later exploited and developed in high school through many design competitions and contests where he received several honors and medals.

Joseph's desire to be an engineer led to an undergraduate career at Virginia Tech. During this time he participated in various projects and internships. One of these projects involved a circuit breaker design where he won third prize in a national competition. Internships at Pratt & Whitney and Cutler-Hammer widened exposure to various aspects of mechanical engineering and generated useful work experience. Involvement with undergraduate research inspired him to learn more by pursuing a graduate degree. During this time he continued his work experience with the Naval Surface Warfare Center in Dahlgren, Virginia. This experience was partially responsible for the inspiration of the research in this thesis.