GEOSYNCHRONOUS BINARY OBJECT DETECTION

THESIS

Patrick B. Cunningham, Captain, USAF

AFIT-ENG-MS-16-M-010

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A**.**
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENG-MS-16-M-010

GEOSYNCHRONOUS BINARY OBJECT DETECTION

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Patrick B. Cunningham, BS

Captain, USAF

March 2016

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENG-MS-16-M-010

GEOSYNCHRONOUS BINARY OBJECT DETECTION

Patrick B. Cunningham, BS

Captain, USAF

Committee Membership:

Dr. Stephen Cain
Chair

Capt. Kevin Vitayaudom
Member

Dr. Mark Oxley
Member

AFIT-ENG-MS-16-M-010

Abstract

This paper will compare competing methods for optically detecting binary objects. This is mostly intended for use in Space Situational Awareness (SSA), though has the potential to be used in other applications. The first method referred to as, "Single Object Detection" is a versatile algorithm which is currently used to detect extraterrestrial objects. However, it does not take into account interference by a nearby object. Therefore a second algorithm is investigated, referred to as "Binary Object Detection", which does. The binary detection algorithm proved to have a comparable or superior Receiver Operating Characteristic (ROC) curve (based upon the area under the curve) in all cases. The algorithm was tested with both simulated and measured data containing single points and binary objects.

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Dr. Stephen Cain, for his guidance and support throughout the course of this thesis effort. The insight and experience was certainly appreciated.

Patrick B. Cunningham

Table of Contents

Page

List of Figures

GEOSYNCHRONOUS BINARY OBJECT DETECTION

I. Introduction

General Issue

Detecting binary objects (two objects that appear to be very close) is critical to SDA (Space Domain Awareness). It is currently possible for a threat nation to deploy a spy satellite in close proximity to one of our assets and use this satellite to hamper our abilities or potentially acquire intelligence from us. A second scenario involves being able to detect a piece of space debris which is in danger of colliding with and potentially destroying, one of our orbital assets. Thirdly, it is possible for an asteroid to escape detection by being too close to a brighter object, which would then serve to mask it. Without the ability to detect these threats, we will have no way of initiating any kind of countermeasure against them. Other, less threatening scenarios in which this would be useful, involve modular space assets, designed to separate in orbit to perform repairs on other orbital assets, or even exoplanet detection.

Problem Statement

Current detection methods are adept at locating a single object in space, as well as two objects that are greatly separated, but have difficulty finding a second object that is nearby. [1] The problem is exacerbated if, as in the two examples listed above, the second object is much dimmer than the first object. One reason for this shortfall is that light generated by the Point Spread Function (PSF) of the brighter object can completely obscure that of the dimmer object if the two are too close together. This is why, the brighter the second object is, or the farther they are apart, the better the current detection

algorithm performs. However, enough possibilities exist that could confound this method, that it is necessary to develop a more specialized approach to deal with the binary object scenario.

Research Objectives/Hypotheses

The current methods simply make a scan of the entire area to detect objects The proposed method, Binary Object Detection (BOD) would work with the current method in that it waits for the current algorithm to detect something and then scans the area around this object for a second object that might not have been detected by the initial scan [2]. In this case the algorithm looks at a pixel and determines how bright an object would be if there were an object in that pixel, then applies a binary test to determine the probability that there an object of that brightness in the pixel, given the known object in its vicinity. In theory, this will have the greatest advantage over the original method when the objects are very close together or the second object is very dim. As the PSFs of the two objects are farther apart, the current method and BOD will likely have more similar results.

Research Focus

For this project, a software only solution was sought. The program was written in Matlab, which is very portable and can be used with multiple platforms. This way the algorithm could be implemented on existing hardware without the need to purchase additional equipment. Currently the program can operate via an imported .mat file but the code can be altered for additional file types if necessary.

Methodology

As stated above, the algorithm was implemented in Matlab. Once the algorithm itself was devised, an artificial pair of binary objects was created, via simulation code, and was processed by both the current algorithm and BOD. This was repeated for several different distances and brightness levels. This produced a series of Receiver Operating Characteristic (ROC) curves, which were compared against each other. Next, actual data was collected by shining Light Emitting Diodes (LEDs) through pinholes and photographing the pinholes. The two algorithms were then fed this data and the resulting ROC curves were compared.

Assumptions/Limitations

There are several assumptions that are necessary for this approach to work. The noise is assumed to be Poisson, meaning that the objects are reflecting incoherent light, most likely from the sun and the electronic noise from the camera is also assumed to be negligible. The optical system is also assumed to be space invariant. Additionally there are proximity limitations associated with the algorithm. The point spread function of the optical system is also assumed to be measured or known a priori. This is true for both the baseline algorithm and BOD. Because of this assumption, the integration time of the sensor must be short enough so that space objects do not streak significantly in the observed images.

Implications

If the camera used is a photon counting detector of some sort and the light is incoherent, the dominant form of noise will be Poisson [3, p. 485]. If the objects exceed

the “nearby” proximity then the code will not detect them because it will not look outside a certain distance, however the current method that it is paired with will likely detect both objects in this case. The assumption that the two point sources are not touching is more critical. If the two point sources are touching, it will be unable to differentiate them from one source that is two pixels long. If the system is not space invariant than the way the code calculates the Optical Transfer Function (OTF) may be inaccurate.

Outline of Chapters

This document is organized as follows. Chapter II will discuss prior work in this topic area as well as explaining part of the mathematical derivation of the algorithm. Chapter III will explain how the code functions as well as describing how the real world data was collected. Chapter IV will cover the results of both methods as well as comparing the current method to the BOD. Lastly, Chapter V will discuss conclusions that can be drawn from this research, as well as list some recommendations for continued investigations and implementation.

II. Literature Review

Chapter Overview

First, this chapter will examine the current method used to detect unknown orbital objects. It will then take a look at some other object detection methods and explain why they are not relevant to our current area of interest. Next, previous work in the area of multi-hypothesis testing will be discussed, as this is critical to BOD. Lastly, BOD will be contrasted with this prior work.

Current Method

Currently, Geosynchronous Earth Orbit (GEO) objects are detected via an algorithm which assumes that the background noise is Gaussian, not Poisson. [2] This method is implemented through the well known Source Extractor or SExtractor program. This method, as stated in Chapter I, is good at locating solitary objects, but has shortfalls in cases where two objects have overlapping PSFs. This can happen if the two objects are very close or if there is a great deal of atmospheric turbulence (which will cause the PSFs to spread out [4, p. 76]). Additionally, the dimmer the secondary object is, the less likely SOD is to pick it up. Recently a new version of SOD was proposed by Gessel [1]. This method is referred to throughout this research as Single Object Detection (SOD).

SOD makes the assumption that objects in GEO do not move appreciably during the exposure time. This assumption holds if the telescope is staring at a section of the sky and not tracking at the sidereal rate. The exposure time must still be short enough so that small orbital perturbations are not detectable. Once the data is collected, the next step is image processing.

To process this data, SOD begins by examining a neighborhood around the candidate pixel, referred to as the processing window. Within this neighborhood the average background is computed via a median filter operation. The background is then subtracted from the data in the window.

Next, a binary hypothesis test can be used to accomplish object detection. To do this the probability that there is one object, $P(D|H_1)$ will be compared with the probability that cell is empty, $P(D|H_0)$. The probability of the data, *D*, given either hypothesis is expressed as Eq. (2.1),

$$P(d(x,y)|H) = \prod_x \prod_y \frac{(I(x,y))^{d(x,y)} e^{-I(x,y)}}{d(x,y)!}, \tag{2.1}$$

where $I(x,y)$ is the expected value of the data. For the single object detector the substitutions would simply be; $E(d(x,y)|H_1) = \alpha_n h(x,y) + b_n$, and $E(d(x,y)|H_0) = b_n$, where *h(x,y)* is the impulse response of the system, α_n is the brightness of the object and b_n is the average value of the background.

In general, the Log Likelihood Ratio Test (LRT) , Λ, is compared against a threshold, t. If $\Lambda > t$ the object is said to be absent and if $\Lambda < t$ the object is said to be present. In order to acquire a ROC curve, a range of t values is necessary [4, p. 92].

$$\Lambda = \frac{ln(P(D|H_1))}{ln(P(D|H_0))} \tag{2.2}$$

Once an object has been found to match all of these criteria it is compared against Satellite Catalogue as a reference. The Satellite Catalogue is a compilation of all detected orbital bodies for use in object tracking and detection of orbital objects

(https://www.space-track.org/auth/login). If no record of the object exists it is listed as a new object. Since the known objects are ignored, any object that is within the PSF of a known object is also ignored, and becomes hidden along with the known object. Additionally, if two new objects are within a few PSFs of one another the detector may only register one object instead of two because it neglects to calculate any contributions the first object may have toward the second object.

BOD effectively makes a second pass around each known object looking for a second object. The additional pass may add computation time, but should be able to discover many objects SOD would have missed. BOD will be discussed in more depth in Chapter III. The advantages of the new approach will be demonstrated by comparing a ROC curve using both BOD and SOD.

A different method that is used to detect and catalogue Near Earth Objects (NEOs) is being implemented by a program called PAN-STARRS (PAN-chromatic Survey Telescope and Rapid Response System) [5, p. 2]. The PAN-STARRS telescope monitors the sky by taking long exposure (30 second) images. This image is averaged and combined with images from as many as four telescopes to create a Master Sky image. Whenever a new image is taken of the same section of sky, it is compared against the previous master image. A difference detection algorithm compares the two and documents anomalies. These anomalies are catalogued and added to a separate database for review. If they are found to not already be in the catalogue, they are added as new objects. This method proved to have a 40% increased detection rate over its predecessors. This is shown by looking at the 0.1 probability of false alarm mark in Figure 1, which consists of a comparison between the Poisson and Gaussian models.

In 2012, a new algorithm was proposed for PAN-STARRS, which if implemented could increase its detection rate by as much as 700% (see Figure 1) [5, p. 54]. The proposal involved treating the data as Poisson instead of Gaussian. However, while it performs better than the previous algorithms, it still encounters similar problems for detecting two objects in close proximity to one another, especially in scenarios where there is a large difference in brightness between the two objects [5, p. 52]. PAN-STARRS ability to detect objects in close proximity is dependent upon the quality of the Master Sky image. The proposed BOD method does not depend upon this because it generates its own version of a Master Sky image as part of the algorithm.

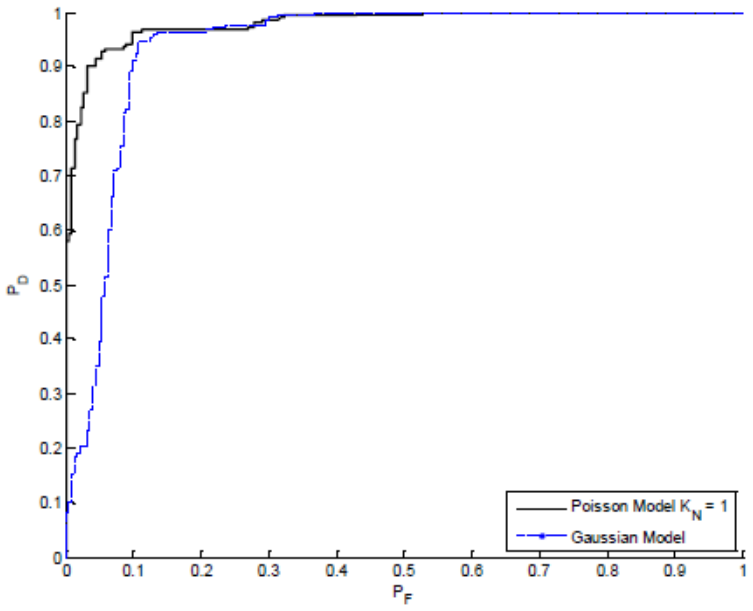

Figure 1: ROC Curve Comparison Between Poisson and Gaussian Assumptions Using Measured Data [5, p. 59].

Other Binary Detection Methods

Radial Velocity is currently in use by NASA to detect binary exoplanets. This works because two planetoids are likely to emit a different spectrum of light. However,

if we are interested in two similar satellites orbiting the Earth in close proximity to one another, the two satellites are likely to emit a similar spectrum. Thus this method will not work for the given area of interest. [6]

Another method NASA employs to detect exoplanets is called Transit Photometry. This method relies upon observing the change in brightness of a star in order to detect a much smaller image (in this case a planet) passing in front of it. While this does potentially parallel the issue of a piece of space debris approaching a satellite, there is a major problem this method will encounter if it were to be used for the purposes presented in this paper. For example, the smaller/dimmer object has to pass directly in front of the larger/brighter object. This requirement renders Transit Photometry ineffective in the scenario of a spy satellite approaching one of our orbital assets. [7]

Multi-Hypothesis Testing

In 2014, a more relevant method of binary object detection was suggested. To detect an object within an image, this method uses a multi-hypothesis test. In this case, the object, o, is assumed to be indistinguishable from a point source, as displayed in Eq. (2.3)**,** where a_n is the brightness of the source and x and y are spatial coordinates, and x_0 and y_0 are the location of the object.

$$o(x, y) = a_n \delta(x - x_0, y - y_0) \tag{2.3}$$

The mean of the data itself, $d(x, y)$, is the brightness of the point source object times the PSF plus the background noise, b_n, as shown in Eq. (2.4)**.**

$$E[d(x, y)] = h(x, y) * o(x, y) + b_n \tag{2.4}$$

Starting with a known object; this method looks at each pixel in that object's vicinity. It begins by assuming an object is present in the selected pixel. Next, it begins an iterative process to estimate what the brightness of the object would be if there were one there. [8] In order to find the brightness, it is important to know the OTF (Optical Transfer Function), $H(fx, fy)$, of the data. The OTF of the atmosphere, $H_{atm}(fx, fy)$, simulates a long exposure time image, so all of the atmospherically induced aberrations are averaged together to yield Eqs. (2.5) and (2.6), where fx and fy are the spatial frequency components [3, p. 428] . This is multiplied by the OTF of the optical system, $H_{aperture}(fx, fy)$, which is modeled here as the Fourier Transform of the clear pupil function (typically a circle), to yield the total transfer function, H .

$$H_{atm}(fx, fy) = \frac{e^{-3.44\left(fx^2+fy^2\right)^{\frac{5}{6}}}}{\lambda * f} \tag{2.5}$$

$$H(fx, fy) = H_{aperture}(fx, fy) * H_{atm}(fx, fy) \tag{2.6}$$

The method for finding the brightness is detailed below in Eq. (2.7), where a_n is the current brightness estimate, $\mathcal{F}^{-1}$ is an inverse Fourier transform. A method like Pan-starrs does not need this brightness detection algorithm, because the brightness data is included in the previous Master Sky image. However, utilizing this algorithm helps to eliminate the need for such an image [8, p. 37].

$$\boldsymbol{a_{n+1} = a_n \sum_x \sum_y \frac{\left(d(x,y) * \mathcal{F}^{-1}(H(fx,fy))\right)}{a_n * \left(\mathcal{F}^{-1}(H(fx,fy))\right) + b_n}} \tag{2.7}$$

A multi-hypothesis test can be used to accomplish object detection. This differs from the binary hypothesis test in that more than two hypotheses are examined. To do this the probability that there are two objects in the image will be calculated, $P(D|H_2)$ and this will be compared with the probability that there is one object, $P(D|H_1)$, or that cell is empty, $P(D|H_0)$. This method uses the same probability equation as SOD, that is, Eq.(1). As with SOD, the algorithm then takes the log of $P(D|H_1)$, to be used later. For the binary object detector the substitutions would be: $E(d(x,y)|\,H_2) = \alpha_{n,1}h(x - u_1, y - v_1) + \alpha_{n,2}h(x - u_2, y - v_2) + b_n$, $E(d(x,y)|H_1) = \alpha_{n,1}h(x - u_1, y - v_1) + b_n$ and $E(d(x,y)|H_0) = b_n$, where u_1 and v_1 are pixel positions for the first object(in the x and y directions, respectively) to object 1, u_2 and v_2 are coordinates (in the x and y directions, respectively) to object 2, and $\alpha_{n,1}$ and $\alpha_{n,2}$ are the brightnesses of object 1 and 2, respectively. This is summarized in a flow chart, shown below in Figure 2. Finally, a LRT, similar to what is performed in SOD, is computed using the new probability values.

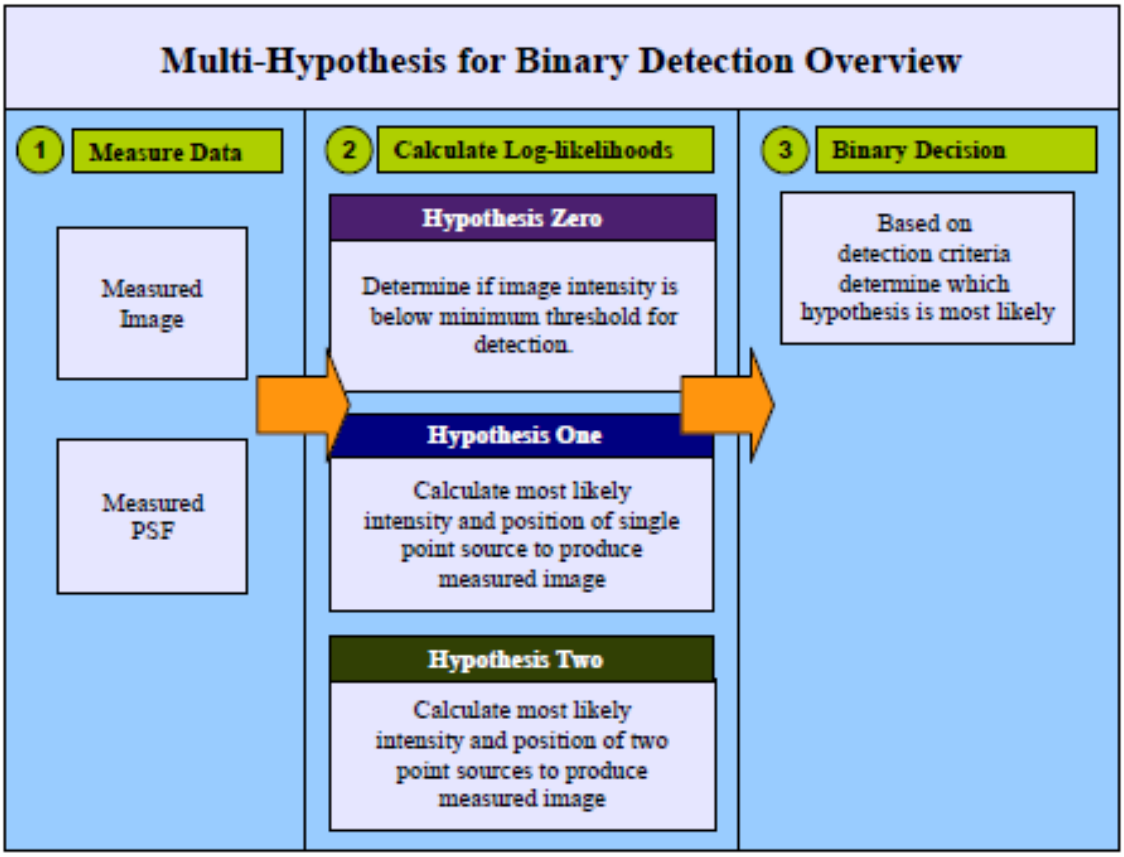

Figure 2: Flowchart of Previous BOD Method [8].

The ROC curve itself is a plot of the probability of a false alarm, P_{fa}, at a given threshold, t, on the x-axis, with the probability of detection, P_d, (for the same threshold) on the y-axis. In order to find the P_{fa}, the image being analyzed would need to not have a second object. Thus if the algorithm says a second object exists, it is a false alarm. This probability is equal to the area under the curve of the Probability Density Function (PDF) of the LRT, where Λ_1 is the LRT produced with only a single object present, up to the current threshold value, t. This is shown below in Eq. (2.8).

$$P_{fa} = \int_{-\infty}^{t} \mathrm{P}(\Lambda_1)\mathrm{d}\,\Lambda_1 \tag{2.8}$$

P_d is calculated the in the same way as P_{fa} except that the PDF of Λ_2 is utilized instead of Λ_1. Λ_2 is the LRT produced when both objects are present. This is represented below in Eq. (2.9).

$$P_d = \int_{-\infty}^{t} \mathrm{P}(\Lambda_2)\mathrm{d}\Lambda_2 \tag{2.9}$$

The algorithm was tested on both simulated and measured data. The simulated data utilized 85 Zernike polynomials to represent atmospherically induced aberrations and thus makes the assumption that the image data will be collected via a short exposure camera. The images themselves consisted of a 128x128 pixel frame; simulating a snapshot that has already detected a single object through SOD, and truncated the snapshot to an area around that object for binary object detection. Two-hundred different, random images were produced with Poisson background noise, to generate a complete ROC curve. This was done both with only one object (for calculating P_{fa}) and with two objects (for calculating P_d).

The measured data was acquired from the Space Surveillance Telescope (SST). Some images were found in which a star passes by a geostationary object. This allowed the binary detection algorithm to run on a measured data set which included an actual binary object. Other frames in which the star was not present, the same geostationary satellite could be located. With both sets of images, a ROC curve was able to be generated for the collected data as well as the simulated data.

In order to avoid counting the same object multiple times, once the program has detected an object, it looks around that object to see if there are any pixels which are brighter than that object by a predefined threshold factor. This threshold was eventually set at a level that ensured a maximum false alarm detection of 10%. Also, the assumption was made that the two point sources had to be separated by at least one pixel. This decreased the false alarm rate further. Some of the results of the simulation are shown below in Figure 3 and Figure 4.

P_{fa} results, point source with 1000 photons								
		atmospheric seeing, D/r_0						
		1.25	1.43	1.67	2.00	2.50	3.33	5.00	10.00
added background noise in photons	1	0%	0%	0.5%	0%	2%	5%	8.5%	8%
	2	0%	0%	0%	0%	0%	0.5%	0%	0%
	3	0%	0%	0%	0%	0%	0%	0%	0%
	4	0%	0%	0%	0%	0%	0%	0%	0%
	5	0%	0%	0%	0%	0%	0%	0%	0%
	6	0%	0%	0%	0%	0%	0%	0%	0%
	7	0%	0%	0.5%	0%	0%	0%	1%	0%
	8	0.5%	0%	0.5%	0%	0%	0%	0%	0%

Figure 3: Probability of False Alarm with 1000 photon source [8, pp. 53-57].

P_D results, binary source 1000/500 photons									
		atmospheric seeing, D/r_0							
		1.25	1.43	1.67	2.00	2.50	3.33	5.00	10.00
added background noise in photons	1	100%	100%	100%	100%	100%	100%	100%	97.5%
	2	100%	100%	100%	100%	98%	90.5%	67.5%	10.5%
	3	97%	95%	92.5%	85%	65.5%	44.5%	8%	0%
	4	30%	26.5%	21.5%	15%	9%	2%	0%	0%
	5	0%	0%	0%	0%	0%	0%	0%	0%
	6	0%	0%	0%	0%	0%	0%	0%	0%
	7	0%	0%	0%	0%	0%	0%	0%	0%
	8	0%	0%	0%	0%	0%	0%	0%	0%

Figure 4: Probability of detection of 2 objects with a 1000 and a 500 photon source [8, pp. 53-57]

A disadvantage of this approach is that it attempts to solve everything simultaneously. That is, it looks at all possible combinations of where the two objects may be at once and calculates that combination with the greatest probability. If this were to be expanded to a third object, the computation time would increase exponentially. If, instead, you locate a known object using SOD and scan each pixel nearby for a second object, the computations could not only be faster individually, but they could be run in parallel (since each pixel does not depend on the outcome of another). This method was used as a starting point for developing the BOD approach described in this paper.

Source Extractor

Another common method of object detection is the Source Extractor, or SExtracor. While this method is not designed primarily for binary object detection, it does focus on faint object detection which is a component of solving the problems stated in the introduction. This method assumes the noise has a Gaussian distribution. Therefore, a different PDF is used. The PDF in Eq. (2.10) is used instead of the PMF in Eq. (1) for the situation in which there is an object present, and is simplified to Eq. (2.11) if the brightness is 0 (meaning no object is present).

$$P(D|H_1) = \prod_x \prod_y \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{\left(d(x,y)-(\alpha h(x,y)+b_n)\right)^2}{2\sigma^2}} \tag{2.10}$$

$$P(D|H_0) = \prod_x \prod_y \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(d(x,y)-(0+b_n))^2}{2\sigma^2}} \tag{2.11}$$

Λ_3, where $\ln(\Lambda_3) = \Lambda$ from Eq. (2), is defined as:

$$\Lambda_3 = \frac{\prod_x \prod_y \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{\left(d(x,y)-(\alpha h(x,y)+b_n)\right)^2}{2\sigma^2}}}{\prod_x \prod_y \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(d(x,y)-(b_n))^2}{2\sigma^2}}}$$

$$= \frac{\prod_x \prod_y e^{-\frac{\left(d(x,y)-(\alpha h(x,y)+b_n)\right)^2}{2\sigma^2}}}{\prod_x \prod_y e^{-\frac{(d(x,y)-(b_n))^2}{2\sigma^2}}}$$

$$= \prod_x \prod_y e^{\frac{-\left(d(x,y)-(\alpha h(x,y)+b_n)\right)^2+(d(x,y)-(b_n))^2}{2\sigma^2}}$$

Expanding these terms yields:

$$\Lambda_3 = \prod_x \prod_y e^{\frac{-\left(d(x,y)^2 - 2(\alpha h(x,y)+b_n)d(x,y)+(\alpha h(x,y)+b_n)^2\right)+d(x,y)^2-2(b_n d(x,y))+{b_n}^2}{2\sigma^2}}$$

$$= \prod_x \prod_y e^{\frac{-(d(x,y)^2 - 2(\alpha h(x,y)+b_n)d(x,y)+(\alpha^2 h^2(x,y)+2\alpha h(x,y)b_n+{b_n}^2)+d(x,y)^2-2(b_n d(x,y))+{b_n}^2}{2\sigma^2}}$$

Now distributing the negative for the first term:

$$= \prod_x \prod_y e^{\frac{-d(x,y)^2+2\alpha h(x,y)d(x,y)+2b_n d(x,y)-\alpha^2 h^2(x,y)-2\alpha h(x,y)b_n-{b_n}^2+d(x,y)^2-2(b_n d(x,y))+{b_n}^2}{2\sigma^2}}$$

Simplifying by cancelling terms gives:

$$\Lambda_3 = \prod_x \prod_y e^{\frac{2\alpha h(x,y)d(x,y)-\alpha^2 h^2(x,y)-2\alpha h(x,y)b_n}{2\sigma^2}}$$

$$\Lambda_3 = \prod_x \prod_y e^{\frac{2\alpha h(x,y)(d(x,y)-b_n)-\alpha^2 h^2(x,y)}{2\sigma^2}}$$

Converting Λ_3 to $\Lambda_4$by taking the natural log of both sides would now give

$$\Lambda_4 = \sum_x \sum_y \frac{2\alpha h(x,y)(d(x,y)-b_n)-\alpha^2 h^2(x,y)}{2\sigma^2}$$

$$= \frac{2\alpha}{2\sigma^2}\sum_x \sum_y h(x,y)(d(x,y)-b_n) - \frac{\alpha^2}{2\sigma^2}\sum_x \sum_y h^2(x,y)$$

$$= \frac{2\alpha}{2\sigma^2}\left(\sum_x \sum_y h(x,y)(d(x,y)-b_n) - \frac{\alpha}{2}\sum_x \sum_y h^2(x,y)\right)$$

Since Λ_4 is compared to a threshold value, a variant of it can be defined as $\Lambda_5 = \Lambda_4 \frac{2\sigma^2}{2\alpha}$

so that

$$\Lambda_5 = \sum_x \sum_y h(x,y)(d(x,y) - b_n) - \frac{\alpha}{2} \sum_x \sum_y h^2(x,y)$$

This step can be repeated to eliminate the remaining term that does not depend on the data, creating another term, $\Lambda_6 = \Lambda_5 - \frac{\alpha}{2} \sum_x \sum_y h^2(x,y)$, simplifying the equation further to the expression found in Eq. (2.12).

$$\Lambda_6 = \sum_x \sum_y h(x,y)(d(x,y) - b_n) \tag{2.12}$$

While this method would ignore the PSF interference created by a second object, it does have an advantage that it does not need to know the brightness of the object at position (x,y) in order to determine there is an object there. The new threshold, Λ_6, can be set to yield the desired P_{fa}, just like SOD.

Summary

The current method for object detection in GEO has shortcomings in the area of binary object detection. While several methods exist for detecting binary objects that are currently in use, they are not relevant to the proposed problems. There is another method which functions similarly to the current detection method but is geared solely towards binary object detection in order to overcome SOD's shortcomings.

III. Methodology

Chapter Overview

In this thesis the application of BOD is applied as a second pass scan, once SOD has detected a single object. This BOD will then be run on the area around each detected object to check for any additional objects that may or may not be present.

First this was run in simulation. This involved creating a simulated image of one source and then simulating a second field with two objects present. The images are then passed through a brightness detection algorithm to compute the brightness of the object, if there is one present. Then, the new BOD algorithm was used to calculate P_{fa} and P_d. These two arrays are used to produce a ROC curve. The simulation was repeated using a Source Extractor like algorithm to produce a ROC curve and that curves were compared. Finally, the entire process was run a second time using a laboratory experiment to collect real data, in place of the simulated data.

Source Field

The source field that was simulated had a pair of objects. One object, placed in the center is 10 times brighter than the dim object, spaced two pixels up and two pixels to the left of object one. The wavelength (λ) of the light was assumed to be $0.5 * 10^{-6}\ m$. The diameter of the aperture was assumed to be 0.5 m. The angular displacement per pixel, for a Nyquist sampled image, is defined as $\theta = \frac{\lambda}{2D} = 0.5 * 10^{-6}\ rad$. By making a small angle approximation ($\Delta x = \theta \Delta z$, where Δz is the height and Δx is the distance in the object plane), in geosynchronous orbit (so $\Delta z = 3.66 * 10^{8}\ m$), this θ translates to a distance of $183\ m$. This means that each pixel of separation corresponds to a physical

displacement of 183 m in geosynchronous orbit, which is consistent with a Raven class telescope. The inverse Fourier transform of the product of is $d(x, y)$ we will be looking at equation (3.1). Then, a small amount of background noise, b_n is added to the system. The last step in creating the source data is to create Poisson noise for $d(x, y)$. The two source fields are shown below in Figure 5 and Figure 6.

$$d(x, y) = \mathcal{F}^{-1}\left(H(fx, fy) * \mathcal{F}\big(o(x, y)\big)\right) + b_n \tag{3.1}$$

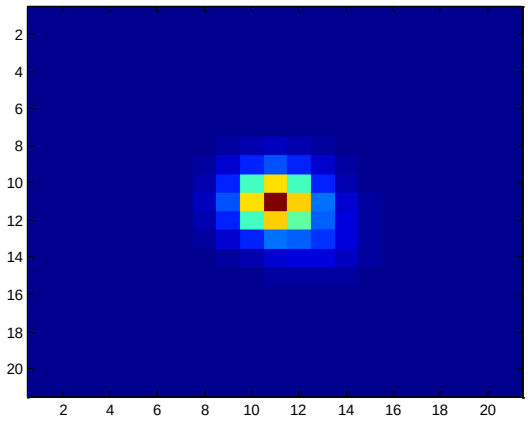

Figure 5: Binary Object Source Field

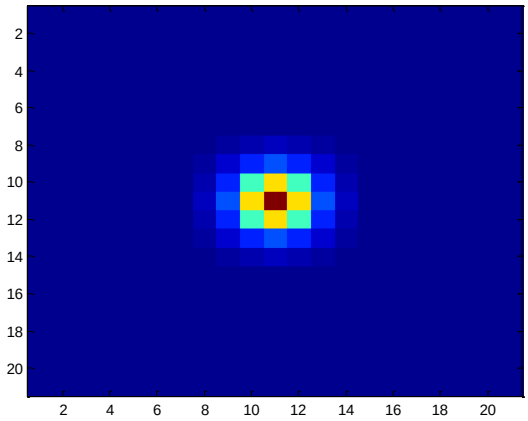

Figure 6: Single Object Source Field

Object Detection

Once the source field is created, the proposed algorithm proceeds to run through a brightness estimation algorithm similar to that in Capt Gessel's work (Eq. (2.7)), for 1000 iterations. However, since BOD begin with the assumption that an object has already been detected and that it is looking for a second object, the LRT calculation is slightly different. In this case the source image is assumed to be centered about the detected image (thus the detected image is at position (0,0) in the (x,y) coordinate system). Now the code will check each pixel in a 128x128 grid to see if a second object is present. In this case, we substitute $E(d(x,y)|H_1) = \alpha_{1,1000}\delta(0,0) * h(x,y) + \alpha_{2,1000}\delta(x,y) * h(x,y) + b_n$, and $E(d(x,y)|H_0) = \alpha_{1,1000}\delta(0,0) * h(x,y) + b_n$, where $\alpha_{1,1000}$ is the 1000th iteration of (Eq. (2.7)) for the object identified by SOD, $\alpha_{2,1000}$ is the 1000th iteration of (Eq. (2.7)), and in this case * represents convolution, for the hypothetical new object, for $P(D|H_1)$, and $P(D|H_0)$ in (Eq. (2.2)). This step is also where we see the biggest difference between BOD and SOD. In SOD the above substitution would simply be $E(d(x,y)|H_1) = a_{1,1000}$ and $E(d(x,y)|H_0) = b_n$ since SOD makes the assumption that there are no other objects present. Since the results of each calculation are not dependent upon any of the others, each pixel can be calculated in parallel, and thus simultaneously, with the rest, to save computation time. This also enables the above substitution to be easily expanded for adding a third object.

One-hundred frames were created with a second object and randomly generated Poisson noise, each of which were processed by the proposed algorithm, producing an

array of Λs. The Cumulative Distribution Function (CDF) of a Gaussian is equal to the area under the curve, up to a threshold value [3, pp. 9-11]. The CDF to calculate the integrals described in Eqs. (2.8) and (2.9).

In order to verify that the Λ array was Gaussian, the values were put through the Lilliefors test, which is designed to tell how normal a particular distribution is. The test first estimates the actual mean and standard deviation of the data. It then calculates what the CDF of the data would be if the data were completely Gaussian. Next, it computes the maximum error between the empirical distribution function and the CDF. Lastly it assesses if this maximum error is statistically significant. If it is not, the data is determined to be Gaussian. [9]

In every case examined in this research, the Λ array was determined to be Gaussian. Therefore, the CDF of the Λ array is equal to the P_{detect} (see Eq. (2.8)). This was then repeated on 100 frames without the second object, with the CDF yielding the P_{fa}. In this case the threshold was given a range of $0.95 < t < 1.05$. P_{detect} and P_{fa} were then linked together to produce a ROC curve for both BOD and SOD. The resulting ROC curves are shown below, in Chapter IV.

Additional Tests

Several additional source fields were simulated to observe the differences between BOD and SOD under different conditions. This involved varying both the brightness and the distance between the points. This was done to verify the hypothesis that BOD will be much better than SOD when the two sources are both close together and

one is much brighter than the other, and that the results will be more similar when the two sources are farther apart and closer in brightness. Three different distances and three different brightness ratios were examined. The results of this test can be found in Chapter IV.

This was done at two pixels separation, 10 pixel separation, and 100 pixel separation. In this context, separation refers to both x and y coordinate difference, thus if the initial object is at (0,0), and the separation is 10 pixels, the second object is at pixel (-10,-10). This was performed at equal brightness, a 10:1 brightness ratio (meaning that the second object is $1/10^{th}$ the brightness of the first), and 100:1 brightness ratio (meaning that the second object is $1/100^{th}$ the brightness of the first).

Experimentation

In addition to the simulated input data listed above, the code was modified slightly process measured data. To test this function, a laboratory level experiment was conducted involving a simulated star field created by viewing a LED (Light Emitting Diode) through a screen of pinholes. A digital photograph was taken of this, through a telescope. The resulting image was substituted for the single point simulated data described above. Then a second pinhole was added and a second photograph was taken of this setup. The result was substituted for the two point simulated source data listed above. Images of the two point data are shown below in Figure 7 and Figure 8. Figure 7 shows what a single frame looks like and Figure 8 shows what several frames averaged together look like.

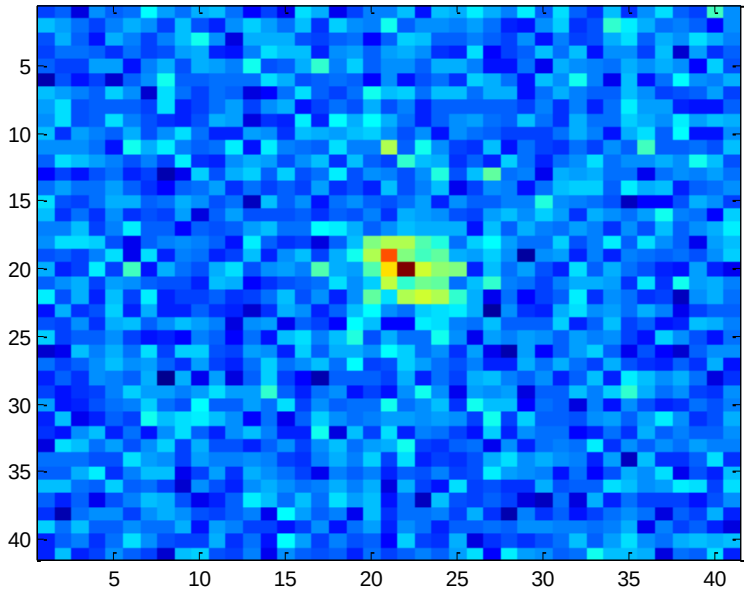

Figure 7: Single Frame of Collected Data for Two Nearby Sources of Differing Brightness

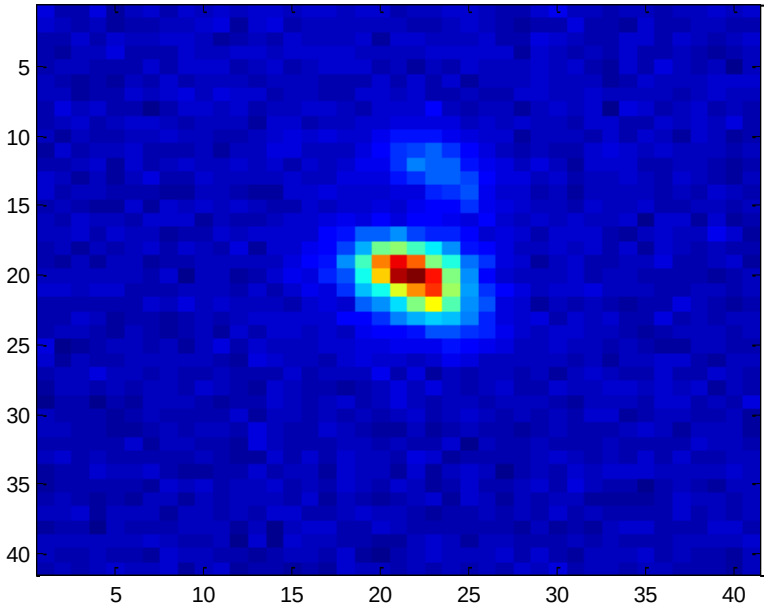

Figure 8: Averaged Image of 100 frames of Collected Data of Two Nearby Sources of Differing Brightness

Since the code assumes that the PSF is known, the code infers the PSF from the single point source by assuming that it is simply a point source, and thus its Fourier transform is a two-dimensional Dirac function. That being the case, since the Fourier transform of the OTF is the PSF convolved with the object (in this case a point source), then the OTF is the inverse Fourier transform of the PSF times the object, which in this case simplifies to (Eq. (3.2)).

$$H(fx, fy) = \mathcal{F}\big(h(x,y) * o(x,y)\big)$$

$$H(fx, fy) = \mathcal{F}\big(h(x,y)\big)\mathcal{F}\big(\delta(x,y)\big) = \mathcal{F}\big(h(x,y)\big)$$

$$\mathcal{F}^{-1}\big(H(fx, fy)\big) = h(x,y) \tag{3.2}$$

The background level was removed by taking the average value of the image and subtracted that from every cell in the image. Each value that is below zero is then set to zero. This normalized the background to zero, removing much of the noise. Without the background noise, the image of the single object is much closer to the PSF, thus the object is closer to actually being a point source. The rationale for subtracting out the noise this way has to do with the Camera Calibration, as described below

After the OTF is calculated we can use (Eq.(2.7)). After this, the procedure is followed just like above in the Object Detection section using the supplied data instead of a simulated input. The resulting ROC curves are displayed below in Chapter IV along with the simulated results.

Camera Calibration

In calibrating a camera, it is important to remove the electrical bias and gain from the equation. These values will be different for each photo-detector, and thus each pixel. To begin, the data for a specific cell (coordinates (x, y) in the photodetector array) with a specific brightness level (denoted by the subscript 1) $(d_1(x, y))$ is expressed below in Eq. (3.3). In this equation, $\gamma(x, y)$ is the gain attached to the pixel being analyzed, $k_1(x, y)$ is the photocount for data set 1 at this location, $B(x, y)$ is the bias for this photo-detector and $n_1(x, y)$ is the noise for this data set at this location. The gain and bias are properties of the pixel itself, whereas the photocount and the noise arenot. The data set will have M samples, so in reality, instead of the data being simply $d_1(x, y)$, the set includes $d_{1,1}(x, y)$ (for the first sample) through $d_{1,M}(x, y)$ (for the last sample). However, for most of this process, the data will be analyzed as an array, rather than looking at the individual sample values.

$$d_1(x, y) = \gamma(x, y)k_1(x, y) + B(x, y) + n_1(x, y) \tag{3.3}$$

In order to find the values of $\gamma(x, y)$ and $B(x, y)$, it is necessary to collect a second set of data, referred to as $d_2(x, y)$ (which will also contain M samples, as explained above). Since we are looking at the same pixel, $\gamma(x, y)$ and $B(x, y)$ do not change. In the second image, $k_2(x, y)$ is the photocountand $n_2(x, y)$ is the background noise , as described in equation (3.4).

$$d_2(x, y) = \gamma(x, y)k_2(x, y) + B(x, y) + n_2(x, y) \tag{3.4}$$

Since $k_1(x,y)$ and $k_2(x,y)$ are photon counts, they are both Poisson [3, p. 90]. This means that their mean and variance will be the same. Here the mean of the photocount of the first data set will be described as $E\big(k_1(x,y)\big) = \overline{k_1(x,y)}$ and that of the second data set is $E\big(k_2(x,y)\big) = \overline{k_2(x,y)}$. Since the data is not simply Poisson, however, the expected value requires some computation (where the mean of $d_1(x,y)$ is $\overline{d_1(x,y)}$ and the mean of $d_2(x,y)$ is $\overline{d_2(x,y)}$). The expected value of $d_1(x,y) - d_2(x,y)$ can be expressed sd: $E[d_1(x,y) - d_2(x,y) = E\big[\big(\gamma(x,y)\, k_1(x,y) + B(x,y) + n_1(x,y)\big) - \big(\gamma(x,y)k_2(x,y) + B(x,y) + n_2(x,y)\big)\big] = E\big[\gamma(x,y)\big(k_1(x,y) - k_2(x,y)\big) + \big(n_1(x,y) - n_2(x,y)\big)\big]$.

Since Gamma is a constant it can get moved outside the expectation operator:

$$E[d_1(x,y) - d_2(x,y) = \gamma(x,y)E[k_1(x,y) - k_2(x,y)] + E[n_1(x,y) - n_2(x,y)]$$

Although the two noise values are different, they should have the same average value, and will thus on average cancel each other out, leaving:

$$\gamma(x,y)\left(\overline{k_1(x,y)} - \overline{k_2(x,y)}\right) = E[d_1(x,y) - d_2(x,y)] \qquad (3.5)$$

While equation (3.5) is a much more concise way of looking a $\gamma(x,y)$ it still contains a hurdle. While $d_1(x,y)$ and $d_2(x,y)$ are known, $\overline{k_1(x,y)}$ and $\overline{k_2(x,y)}$ are not. However, they can be found by looking back at the variance of the data sets. The variance of the two data sets can be expressed below in equation (3.6) and equation (3.7).

$$\sigma_1^2(x,y) = E[\left(d_1(x,y) - \overline{d_1(x,y)}\right)^2] \tag{3.6}$$

$$\sigma_2^2(x,y) = E[\left(d_2(x,y) - \overline{d_2(x,y)}\right)^2] \tag{3.7}$$

Expanding equation (3.6) yields:

$$\sigma_1^2(x,y) = E\left[\left(\gamma(x,y)k_1 + B(x,y) + n_1(x,y) - \left(\gamma(x,y)\overline{k_1(x,y)} + B\right)\right)^2\right]$$

$$= E\left[\left(\gamma(x,y)\left(k_1 - \overline{k_1(x,y)}\right) + n_1(x,y)\right)^2\right]$$

$$= E\left[\gamma(x,y)^2\left(k1 - \overline{k_1(x,y)}\right)^2 + 2n_1(x,y)\gamma(x,y)\left(k_1 - \overline{k_1(x,y)}\right)\right.$$

$$\left. + n_1(x,y)^2\right]$$

$$= \gamma(x,y)^2\overline{k_1(x,y)} + \sigma_{n,1}^2(x,y) \tag{3.8}$$

In equation (3.8), σ_n^2 is the variance of the additive noise. .

By subtracting the two variances and dividing by the difference of the means, it now becomes possible to solve for gamma [10, p. 23]. This is expressed in equation (3.9).

$$\sigma_1^2(x,y) - \sigma_2^2(x,y) = \gamma(x,y)^2\left(\overline{k_1(x,y)} + \sigma_{n1}^2(x,y) - \overline{k_2(x,y)} - \sigma_{n,2}^2(x,y)\right)$$

$$\sigma_1^2(x,y) - \sigma_2^2(x,y) = \gamma(x,y)^2\left(\overline{k_1(x,y)} - \overline{k_2(x,y)}\right)$$

Then dividing by the difference of the means of the data,

$$\gamma(x,y) = \frac{\sigma_2^2(x,y) - \sigma_1^2(x,y)}{E\left(d_2(x,y) - d_1(x,y)\right)} \tag{3.9}$$

Now if $\overline{k_1(x,y)} = 2\overline{k_2(x,y)}$, then $\sigma_2^2 \geq \sigma_1^2$, but if $\sigma_2^2 = \sigma_1^2$ $\overline{k_1(x,y)} = 0$ and $\overline{k_2(x,y)} = 0$ then there is no light reaching the photo-collector. In this instance it is possible to calculate $B(x,y)$. If $\overline{k_1(x,y)} = 0$ and the mean of $\overline{n_1(x,y)} = 0$, then equation (3.3) simplifies to $E[d_1(x,y)] = B(x,y)$. So by finding a data set in which the pixel is dark, and subtracting out the data value of that pixel, you can subtract out the bias value of that pixel from future data sets.

Three Object Analysis

As stated above, it is possible to extrapolate this method further and detect a third object after already finding two binary objects. While it is possible to run the algorithm, as is to find a third object, there are some improvements that can be made, if that is the end goal. To do this requires a third pass after detecting the second object. After the second object has been detected, the scene is run through the process one more time, this time accounting for the second identified object. To do this, substitute$E(d(x,y)|H_1) = \alpha_{1,1000}\delta(0,0) * h(x,y) + \alpha_{2,1000}\delta(x_1,y_1) * h(x,y) + \alpha_{3,1000}\delta(x,y) * h(x,y) + b_n$, (where x_1 & y_1 are the x and y coordinates of the known second object) and $E(d(x,y)|H_0) = \alpha_{1,1000}\delta(0,0) * h(x,y) + \alpha_{2,1000}\delta(x_1,y_1) * h(x,y) + b_n$. This method will be referred to as Three Object Detector (TOD). As you can see, it should be simple enough to expand this method further to include even more objects in a cluster, should such a need arise. This procedure is useful for a scenario in which we have already identified two bright objects and a dim object is between them (as shown in Figure 9), as well as if there is one bright object and two dim objects hiding in its vicinity (as shown in Figure 10). For the experiment, both of these scenarios were tested. For the

first scenario (with two bright objects and one dim object) the two bright objects were the same brightness and were about half of a PSF apart from one another (four pixels in this case), with a dim object (1% the brightness of the two bright objects) halfway in between them. In the other example, the two dim objects (each 1% the brightness of the bright object) are equally spaced around the bright object, about one fourth of a PSF from the bright object (two pixels, in this case).

Due to the difficulties faced by both algorithms in the most severe case above (with one dim object in between two bright objects) the threshold values needed to be changed. This is because in the original threshold range (0.95 to 1.05) SOD was completely unable to detect the middle object. The threshold range used in this experiment was 0.1 to 1.1. Additionally, the number of thresholds tested needed to be increased because TOD's P_{fa} jumped from 0 to 1 in only a few samples, creating a very low resolution curve. The number of sample threshold was increased from 10,000 to 1,800,000 accounting for both the increased resolution and the increased range. While these changes were only needed for the most severe case, they were left in place for the less severe case in order to create a fair comparison.

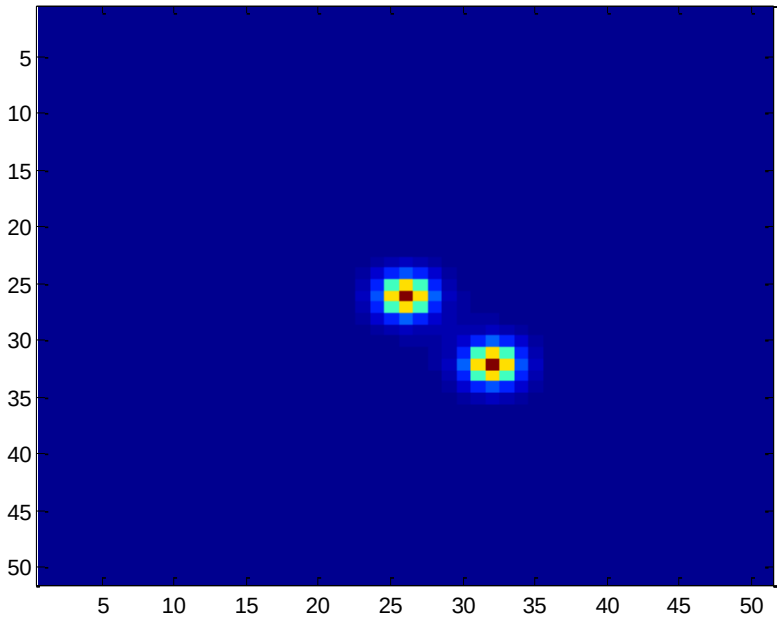

Figure 9: Three Object Cluster With Two Bright Objects and One Dim

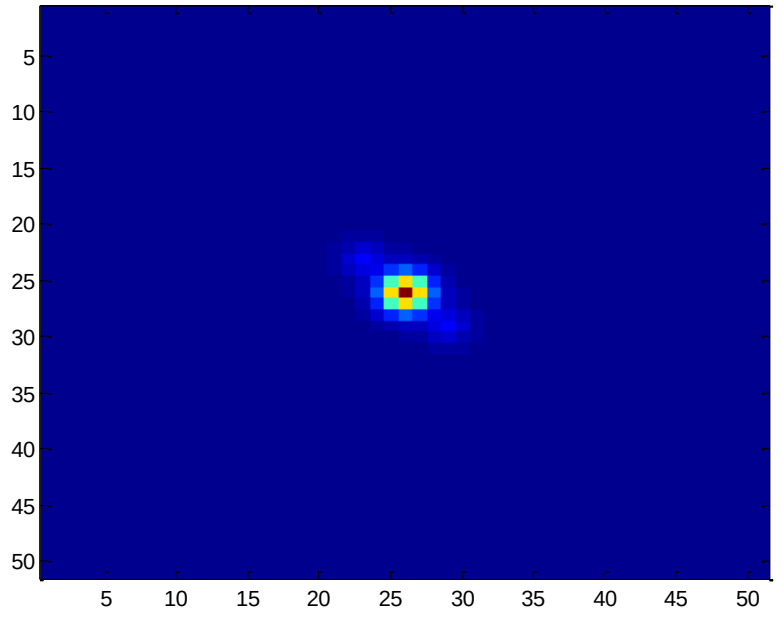

Figure 10: Three Object Cluster with Two Dim Objects around a Bright Object

Summary

The simulated data is made by creating a source field which it can use in a simulated experiment. The algorithm is initialized by looking at each pixel and estimating what the brightness would be if there were an object present in that pixel. Next, BOD looks at the probability that an object of the estimated brightness is present in the given pixel versus the probability that there is no object in the given pixel. By making the assumption that there is already one object in the vicinity, the algorithm takes into account the effect of that detected object's PSF in calculating this probability. This assumption also allows for a reduction in calculation time and should be easily converted into a triple object detection algorithm, should the need arise.

The resulting LRT is compared against a threshold value. If it is greater than that value, an object is said to be present, if it is less than the value the pixel is said to be empty. This gives us the P_{fa} (if the source has only one object) and P_d (if the source has two objects and the second object is in the pixel being analyzed). By matching the P_{fa} and P_d to their respective thresholds, a ROC curve can be produced. This curve is compared against the output of the same source fields in SOD.

Next, a laboratory level experiment was conducted and the images taken were used as inputs to BOD, showing its ability to detect dim objects close to brighter ones using measured data. The PSF was calculated from a single point image. The PSF was then used to find the P_{fa}. P_d was calculated by an experiment with two point sources of light present. The image was processed by both BOD and SOD algorithms and the resulting ROC curves were compared, like the simulated data.

IV. Analysis and Results

Chapter Overview

BOD performed better than or equal to SOD in all cases. As predicted, the difference was greatest when the two objects were closest together, and when one was significantly dimmer than the other. The basis for comparison was to look at the ROC curves side by side as well as P_{detect} when $P_{fa} = 10^{-9}$ (which is a P_{fa} level commonly used in orbital detection). This P_{detect} value will be referred to as the "detection rate" for a given scenario.

Comparing the area under the curves gives an objective comparison method for the two curves. However, in some cases, it is not possible to reasonably differentiate between the two ROC Curves, at least to the level of precision to which Matlab is capable. So, in these cases, P_{detect} at was plotted against the $\log_{10}(P_{fa})$, instead of the P_{fa}. This serves to amplify the differences between the two graphs to the point where they are discernable. To give a numerical comparison between the graphs, the area under each curve was calculated (thus the higher area was the superior method). Due to variances in the curves, 100 iterations were generated each sample containing different random Poisson noise, with each iteration tested by both BOD, SOD, and Source Extractor. This generated a set of three ROC curves (as well as detection rates), one for each method. This process was repeated 20 times for each scenario, creating 20 sets of ROC curves, allowing for a mean and variance to be calculated for the area under the curve.

Results of Simulation Scenarios

The first set of data to be examined will be the two object simulated scenarios. The bright object was set to 10,000 photons for all scenarios, and the background level was set to an average of 10 photons per pixels in all simulations. The brightness of the second object was varied to give different brightness ratios. The second objects location was also varied, to give data points at different distances.

Of the two object cases, the scenario with the brighter object being ten time brighter than the dim object will be first. This scenario was divided into two subcategories, based upon the offset of the two objects (two pixel and ten pixel offsets). This means that if object one is located at position (0,0), then object two will be located at position (2,2) or (10,10) respectively. Since the PSF is approximately eight pixels across (that is, it spreads from the object, four pixels in each direction), the closer of the two conditions is still within one PSF length and the longer of the two is about two and a half PSF's from one object to the other. An image of this is shown in Figure 11

For this scenario, ROC curves are shown in Figure 12. The average areas under the two sets of curves are 1 (to the maximum precision available in Matlab) and a standard deviation of 0 (creating a series of "perfect" ROC curve) for BOD and 1 with a standard deviation of 0 for SOD. The Source Extractor, also had an average area 1 with a standard deviation of 0. Since no clear difference can be determined (to within Matlab's displayed significant digits) semilog-x plots were also produced. This is done, as discussed above, by taking the $\log_{10}(P_{fa})$ and plotting it against the P_{detect} from the previous plots. This will apply more emphasis to the left side of the plot, which is the area we are most interested in. The semilog-x plots are shown in Figure 13. The

detection rate at $P_{fa} = 10^{-9}$ was 0.999999999999895 (shown to full precision to denote that it is not equal to 1) for SOD with a standard deviation of $2.92 * 10^{-13}$ and the average detection rate was 1 with a standard deviation of 0 (to machine precision) for BOD. This means that if the threshold value, Λ, from Eq. (2.2), were set to a value such that $P_{fa} = 10^{-9}$, the BOD method would be able to detect an object in this scenario, every time. (For the Source Extractor method, the average detection rate was only $7.62 * 10^{-24}$ with a standard deviation of $2.65 * 10^{-36}$. At this distance and brightness level, the two newer methods perform very similarly.

Using a semilog-x plot enables the viewer to visibly detect the differences between the SOD and BOD. This is a useful tool when the two graphs appear similar initially and it is difficult tell which is actually better in a given range. In this case, Source Extractor's performance is so bad, in logspace, that it doesn't even appear on the graph. In some future examples, semilog-x plots will no longer be necessary because it will be apparent from the standard ROC curve which method is superior. As the light differential and distance between the two sources becomes greater, the original ROC curves will look more and more different.

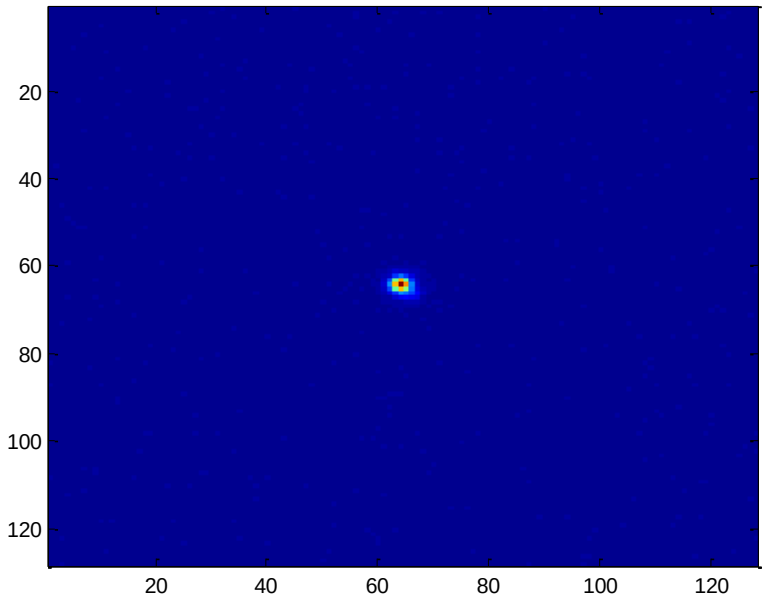

Figure 11: Image of 2x2 pixel offset where the bright object is 10,000 photons the dim object is 1,000 photons with an average background level of 10 photons per pixel.

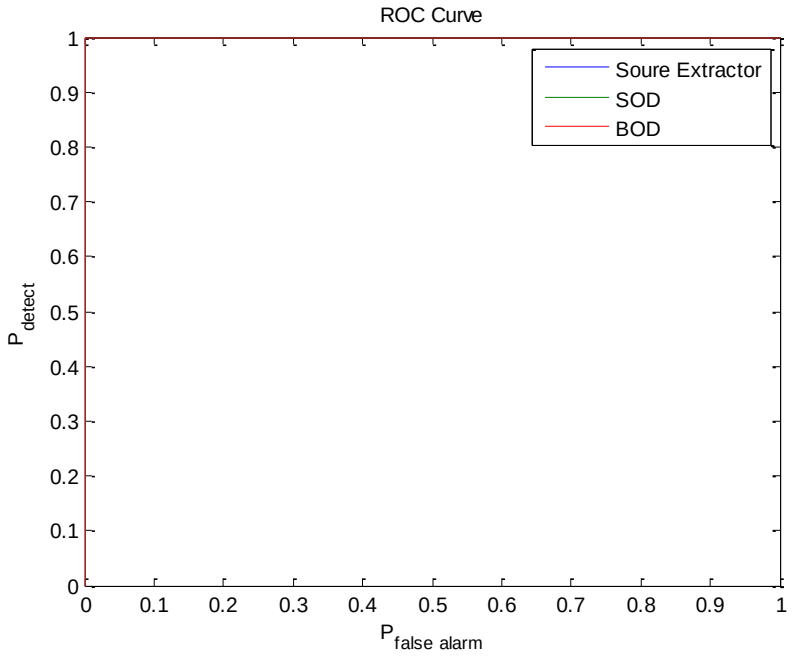

Figure 12: ROC Curves of two sources separated by 2x2 pixels where the bright object is 10,000 photons the dim object is 1,000 photons with an average background level of 10 photons per pixel.

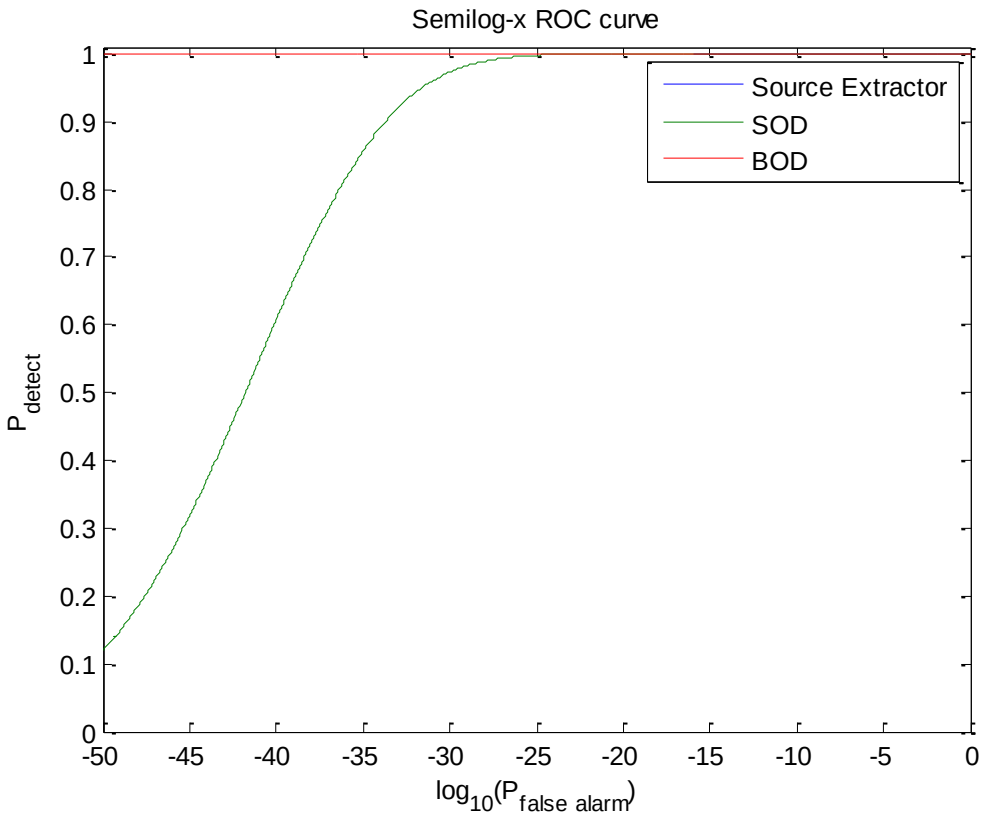

Figure 13: Semilog-x ROC Curve of two sources separated by 2x2 pixels where the bright object is 10,000 photons the dim object is 1,000 photons with an average background level of 10 photons per pixel.

The next scenario to be examined involves moving the second source 10x10 pixels away from the first source. The relative brightness remained unchanged, so that the second source was still 1/10 the brightness of the first source. An image of this scenario is displayed in Figure 14. The average area under these two sets of curves is 1 for SOD with a standard deviation of $4.412 * 10^{-17}$, and about 1 with a standard deviation of $2.547 * 10^{-17}$for SOD. In this case, both tests yield a "perfect" ROC Curve, up to 15 decimal places (Matlab's precision level) and thus the two are visually indistinguishable from one another. The detection rate, in this case yields 1, with a

standard deviation of 0, for all three methods. This means that in all tests, the second object was able to be detected all of the time when $P_{fa} = 10^{-9}$. The similarities are not surprising considering how closely the two methods performed when the second object was only 2x2 pixels offset with the same brightness.

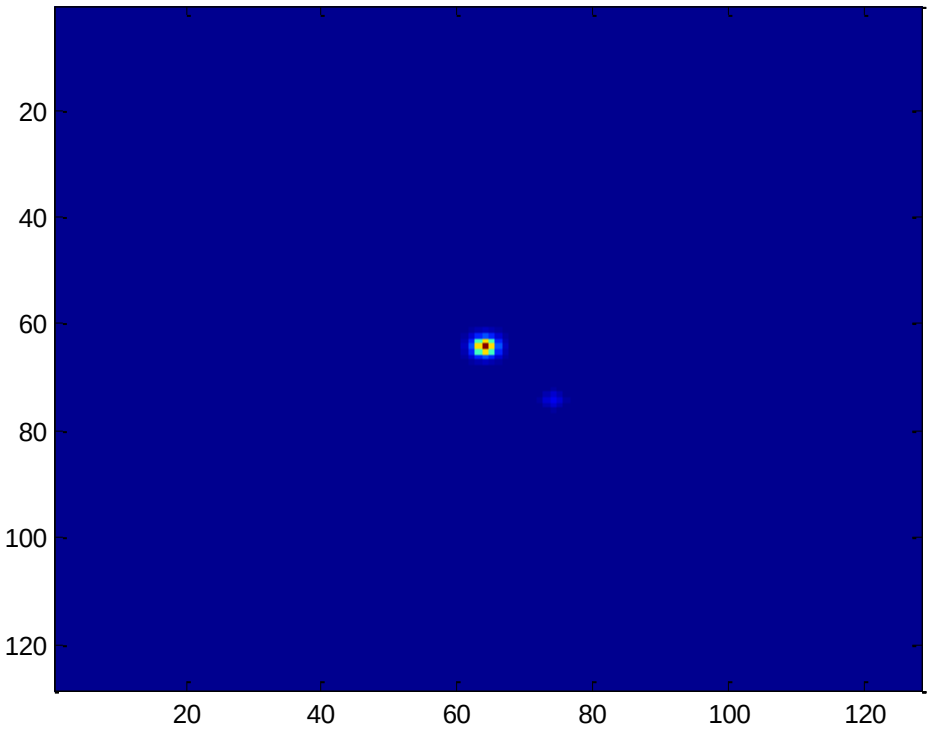

Figure 14: Image of 10x10 pixel offset where the bright object is 10,000 photons the dim object is 1,000 photons with an average background level of 10 photons per pixel.

The next experiment was to change the brightness of the dim object to 1% that of the bright object. This allows the PSF of the bright object to better mask that of the dim object. As before, the pixel offset was initially set to 2x2 to look at the most severe case. An image is displayed below in Figure 15. Here we see a much greater difference between the ROC curves, so the semilog-x portion of the experiment will not be

necessary. Sample ROC curves are shown in Figure 16. The average of the areas under the 20 samples of the BOD ROC curves was 0.910 with a standard deviation of 0.0197. The average under the corresponding SOD ROC curves was 0.843 with a standard deviation of 0.0266. Source Extractor had a mean area of 0.915 with a standard deviation of 0.0415. The average detection rate at $P_{fa} = 10^{-9}$ was 0.616 for BOD, $1.71 * 10^{-5}$ for SOD, and $2.5 * 10^{-4}$ for Source Extractor. This paints a much clearer picture that BOD outperforms SOD when the brightness differences are severe and the objects are close together. Since these areas and detection rates were lower, this was the most difficult scenario for both methods to detect due to the dimness of the second object coupled with their proximity to each other.

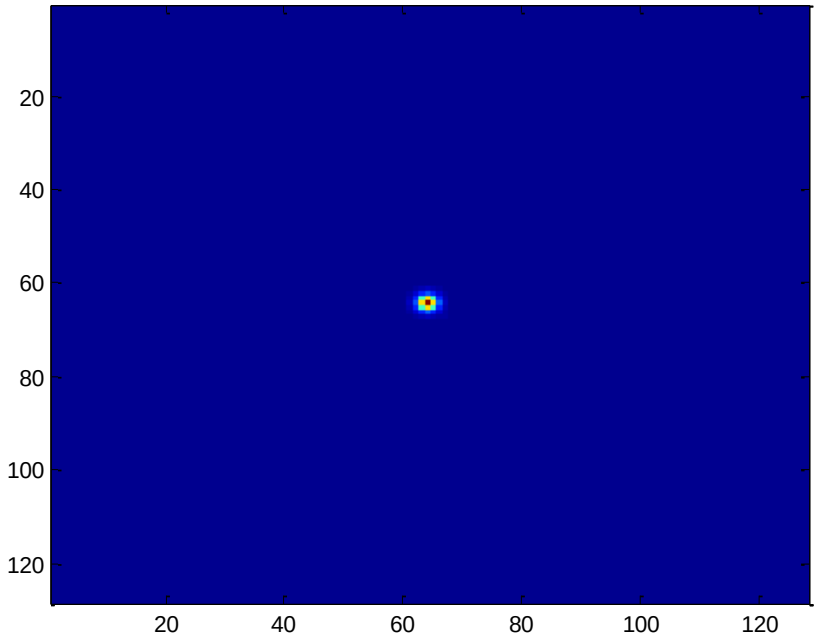

Figure 15: Image of 2x2 pixel separation where the bright object is 10,000 photons the dim object is 100 photons with an average background level of 10 photons per pixel.

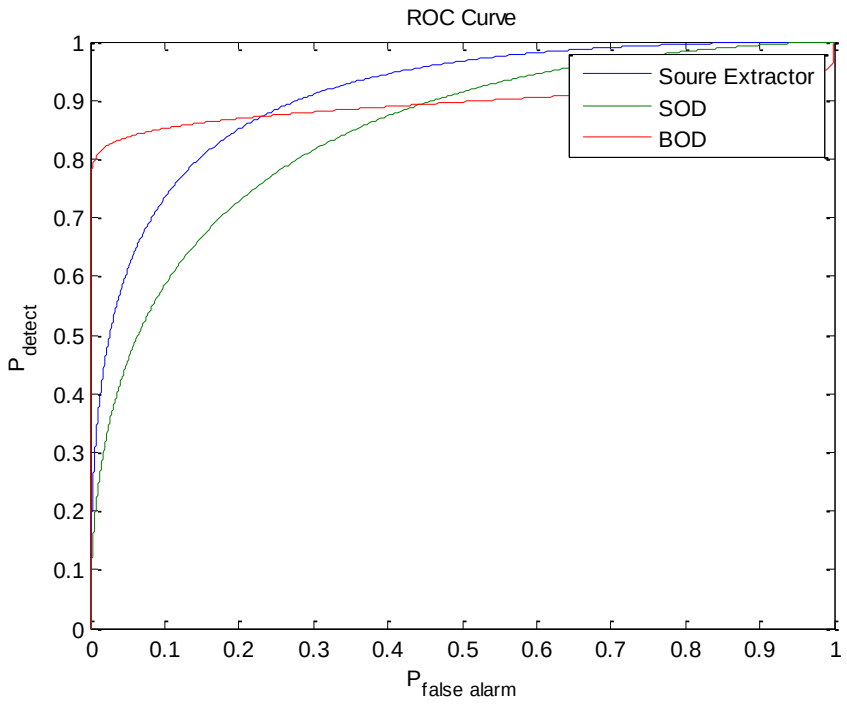

Figure 16: ROC Curve of two sources separated by 2x2 pixels where the bright object is 10,000 photons the dim object is 100 photons with an average background level of 10 photons per pixel.

The final scenario for the two-object system was one with a 10 pixel offset and 1:100 brightness ratio between the two objects. Though the second object is visually imperceptible at this brightness level, an image of this setup is shown in Figure 17. The semilog-x ROC curve can be found in Figure 18 and the regular ROC curves can be found in Figure 19. The average area under the ROC curve for the BOD method was 0.9967 with a standard deviation of 0.0019. The average area under the ROC curve for the SOD method was 0.9983 with a standard deviation of 0.0011. For Source Extractor the average area was 0.99995 with a standard deviation of $4.42 * 10^{-5}$. The performance

for BOD was 0.9876 and 0.9861 for SOD. The detection rate for Source Extractor was 0.349 at that point. Once again, the BOD method proves to be the superior object detection algorithm. Additionally, these four tests demonstrate that the displacement between the two objects as well as their relative brightness to one another affect the outcome in the same direction as expected. This demonstrates that the greater the relative brightness difference, the greater the difference between the two methods, and the greater the distance between the two objects, the closer the two methods will perform to one another.

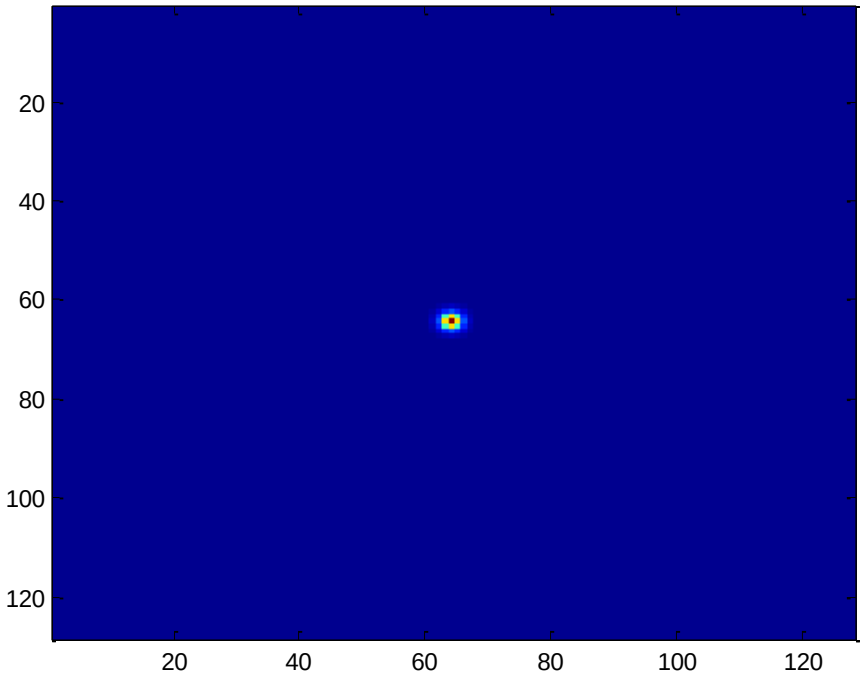

Figure 17: Image of 10x10 pixels where the bright object is 10,000 photons the dim object is 100 photons with an average background level of 10 photons per pixel.

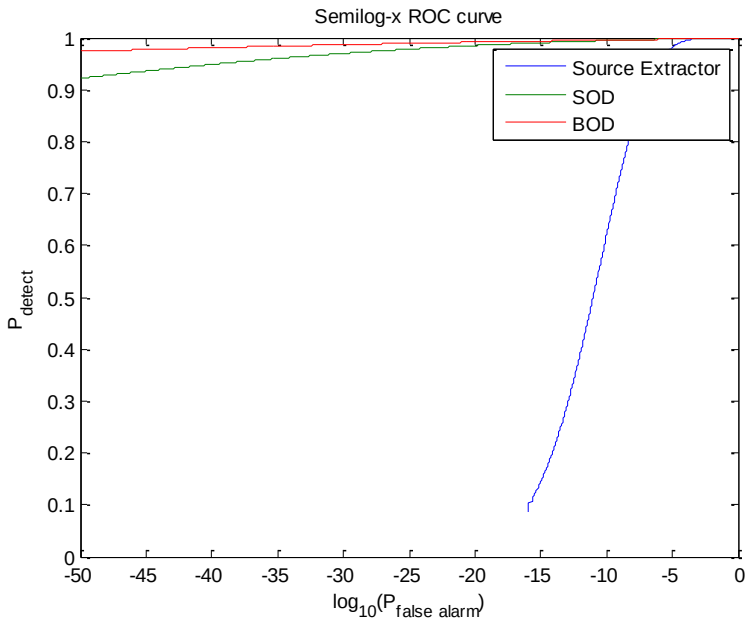

Figure 18: Semilog-x ROC Curve of two sources separated by 10x10 pixels where the bright object is 10,000 photons the dim object is 100 photons with an average background level of 10 photons per pixel.

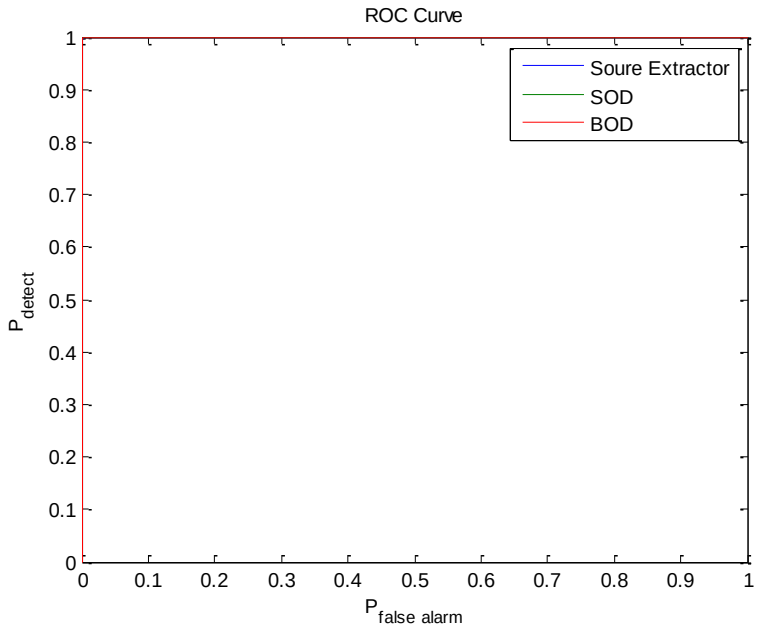

Figure 19: ROC Curve of two sources separated by 10x10 pixels where the bright object is 10,000 photons the dim object is 100 photons with an average background level of 10 photons per pixel.

Results of Laboratory Data

For the laboratory results, 100 samples were utilized to create each ROC curve, but only one ROC curve per method was produced. The image of the sources itself (after averaging all frames together) is displayed in Figure 20. The ROC curves are displayed in Figure 21 and semilog-x ROC curves are displayed in Figure 22. The area under the curve for the BOD method is 0.936 and the area under the curve for SOD was 0.937. The Source Extractor method had an average area of 0.987. The detection rate for BOD is 0.779, 0.780 for SOD, and for the Source Extractor method, effectively 0. Measurements of the brightness of the two objects relative to one another reveals that the bright object, in this scenario, is a little more than two times brighter than the dim object (814 photoelectrons as opposed to 357 photoelectrons). This ratio was found by taking the sum of the image with only the first light and comparing it to the sum of the image with both lights. Looking at the length of the PSFs reveals that they are 1.5 PSF lengths apart. Since the two objects are more than a PSF separated, and since the brightness are close in magnitude, it is expected that the two methods would perform roughly the same.

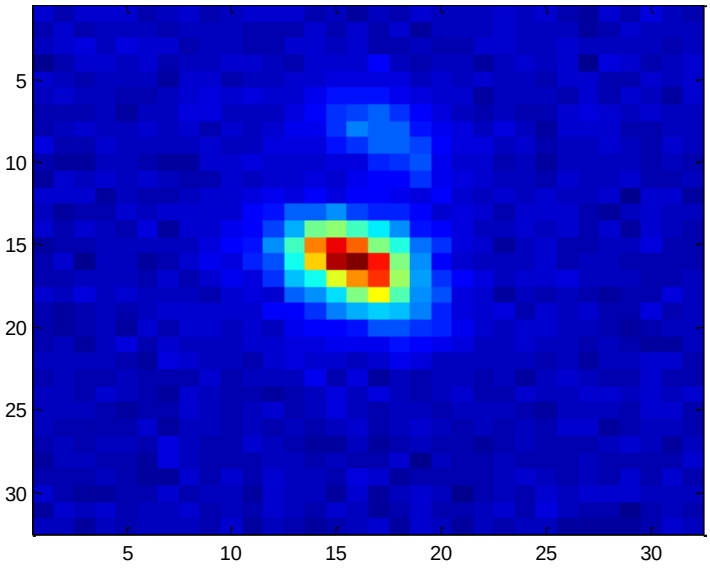

Figure 20: Image of First Experiment, with a bright object of approximately 814 photons, a dim object of approximately 357, and a background level of about 134 photons per pixel.

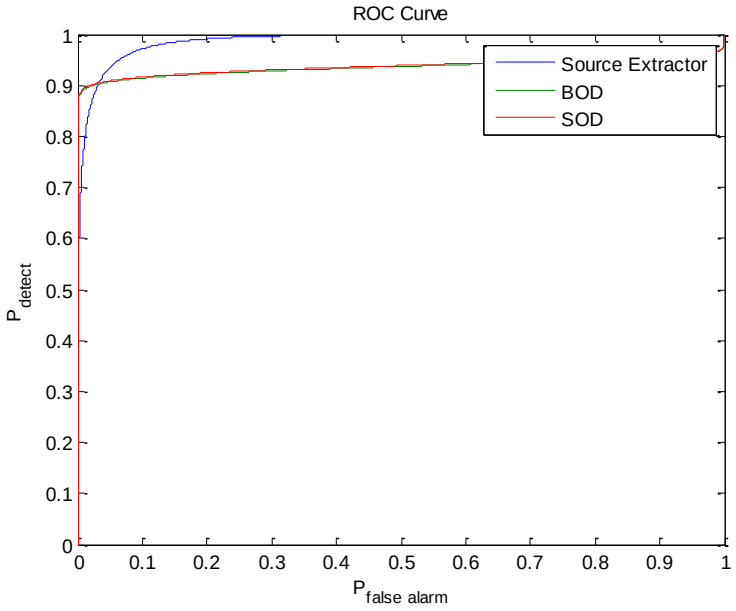

Figure 21: ROC Curve of First Experiment, with a bright object of approximately 814 photons, a dim object of approximately 357, and a background level of about 134 photons per pixel.

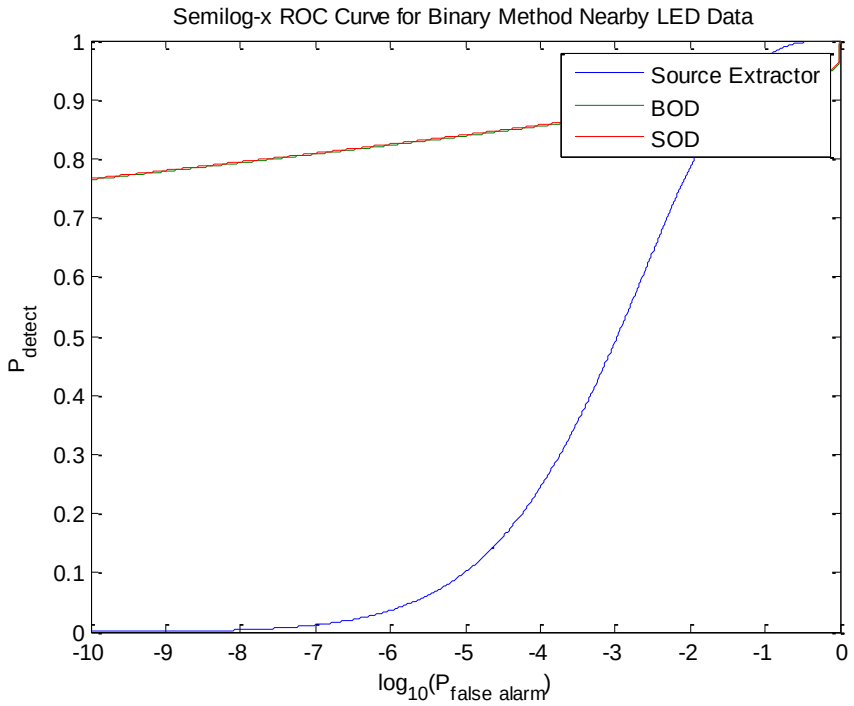

Figure 22: Semilog-x ROC Curve of the first experiment.

In the second laboratory test, the PSFs of the two objects were much more spread out. This enabled more masking of the dim object by the bright object. Additionally, they were closer together. An image of this is shown below in Figure 23. The area under the curve for the BOD method is 0.838 and the area under the curve for SOD was 0.503. The Source Extractor method had an area of 0.504. This is depicted in Figure 24. The detection rate for BOD is 0.363 and $1.28 * 10^{-15}$ for SOD. The Source Extractor method had a detection rate of 0. This is shown in the semilog-x ROC curve in Figure 25.

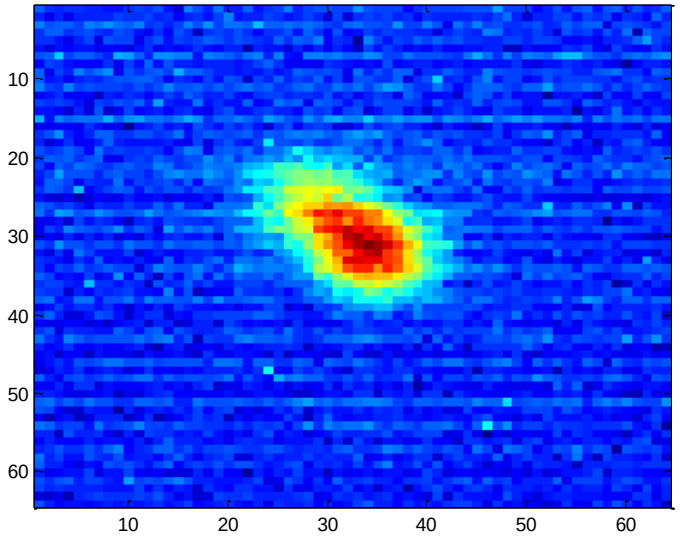

Figure 23: Image of Second Experiment, with a bright object of approximately 3,134 photons, a dim object of approximately 972, and a background level of about 376 photons per pixel.

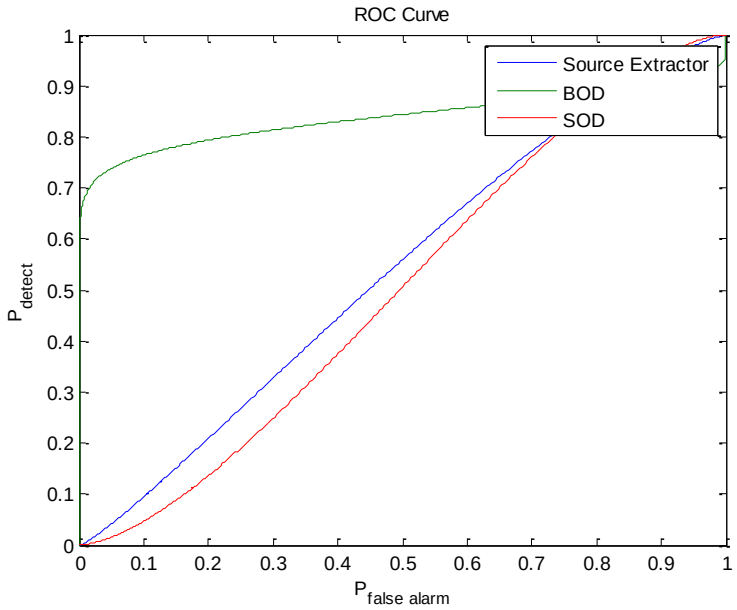

Figure 24: ROC Curve for Second Experiment, with a bright object of approximately 3,134 photons, a dim object of approximately 972, and a background level of about 376 photons per pixel.

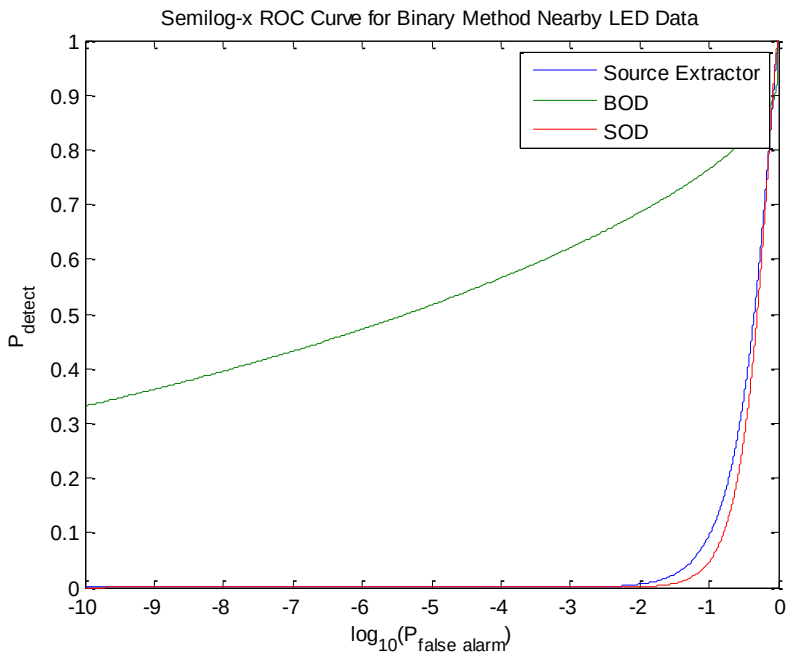

Figure 25: Semilog-x ROC Curve for Second Experiment.

Three Object Detection

Lastly, the BOD algorithm was modified slightly to detect a third image. The new method, TOD (Three Object Analysis), was subjected to two different scenarios. In the first, and arguably more interesting of the two scenarios, there are two bright objects on either side of a dim object. In both cases, the dim object was 1% the brightness of the bright object. As in the previous simulations, 20 samples were generated using random Poisson noise.

The first scenario to be analyzed is that with two bright objects and a dim object midway between them. The results of TOD, with the assumption that both bright objects are already detected, were compared to both that of BOD, with the assumption that only one of the two have been detected, and that of SOD. The "bright" objects are 100x brighter than the dim object, and are spaced 2x2 pixels on either side of it. An image of

this scenario is displayed in Figure 26. The ROC curves are displayed in Figure 27 the semilog-x ROC curve is displayed in Figure 28. The areas under the three curves were tabulated for each method. The average area under the ROC curves for TOD were 0.7962 with a standard deviation of 0.02469. The average for BOD was 0.7416 with a standard deviation of 0.03291, and for SOD the average was 0.7581 with a standard deviation of 0.03650. The average detection rates were 0.1623, $3.200 * 10^{-06}$, and $1.741 * 10^{-6}$ for TOD, BOD, and SOD, respectively. With this great of a brightness difference, TOD's benefits are apparent.

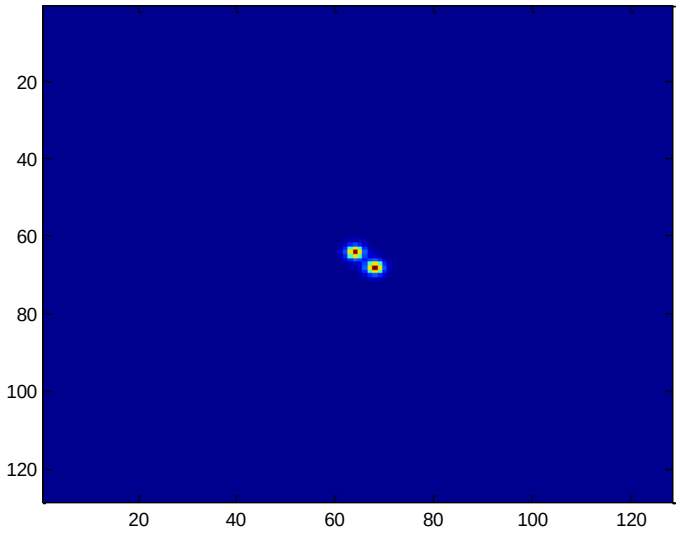

Figure 26: Image of Three Object Scenario, the two bright objects are 10,000 photons and are spaced 4x4 pixels apart with the dim object is 100 photons halfway between them and an average background level of 10 photons per pixel.

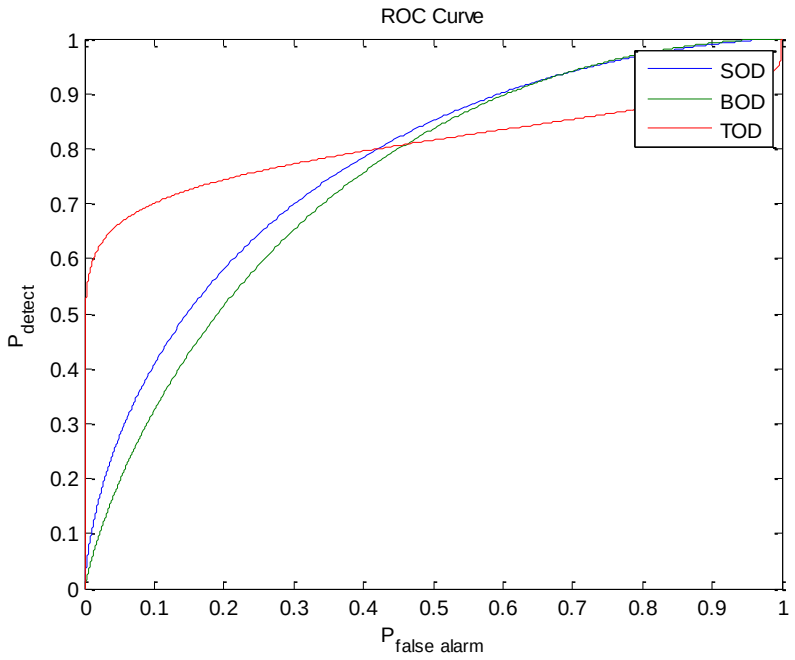

Figure 27: ROC Curves for Three Object Scenario, the two bright objects are 10,000 photons and are spaced 4x4 pixels apart with the dim object is 100 photons halfway between them and an average background level of 10 photons per pixel.

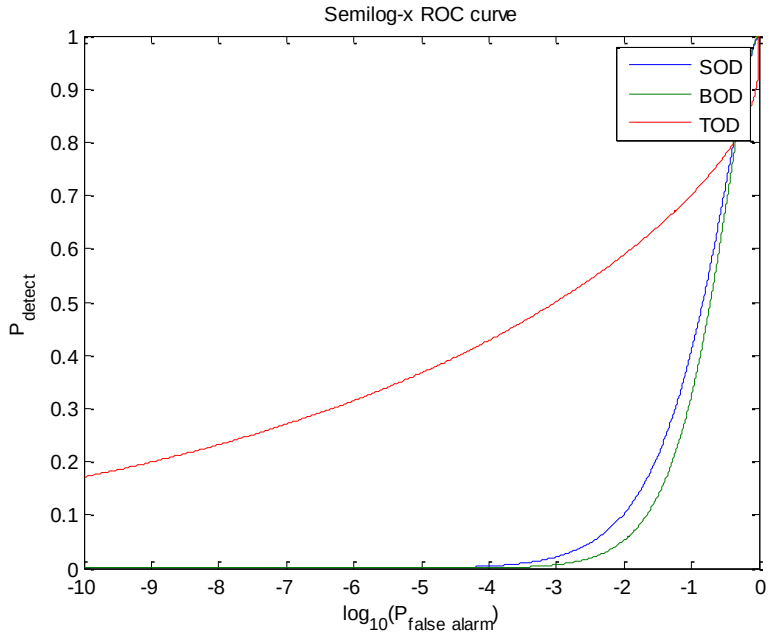

Figure 28: Semilog-x ROC Curves for Three Object Scenario, the two bright objects are 10,000 photons and are spaced 4x4 pixels apart with the dim object is 100 photons halfway between them and an average background level of 10 photons per pixel.

Finally, this paper examined the scenario in which there is one bright object in between two dim objects. The results of TOD, with the assumption that the bright object and one dim object have been detected, were compared to both that of BOD, with the assumption that only the bright object has been detected, and that of SOD. An image of the scenario is displayed in Figure 29. Sample semilog-x ROC curves are shown below in Figure 30. The areas under the three curves were tabulated for each method. The average area under the ROC curves for TOD was always exactly 1 (so there was no standard deviation). The average for BOD was 1 with a standard deviation of $3.602 * 10^{-17}$, and for SOD the average was almost 1 with a standard deviation of $2.547 * 10^{-17}$. Because the area was approximately 1 in each case, the ROC curves themselves are not displayed, as they are visually indistinguishable from one another. For BOD and TOD, the detection rate in all trials was 1. For SOD it was close, at 0.999999999997207 (again, shown to full precision to denote that it is not quite 1). It was expected that the three methods would perform similarly in this scenario because it closely matches the two object scenario with the dim object being 1:10 the brightness of the bright object, with a pixel offset of 2x2. In this case, the third object barely interferes with the PSF at all, at the point in question.

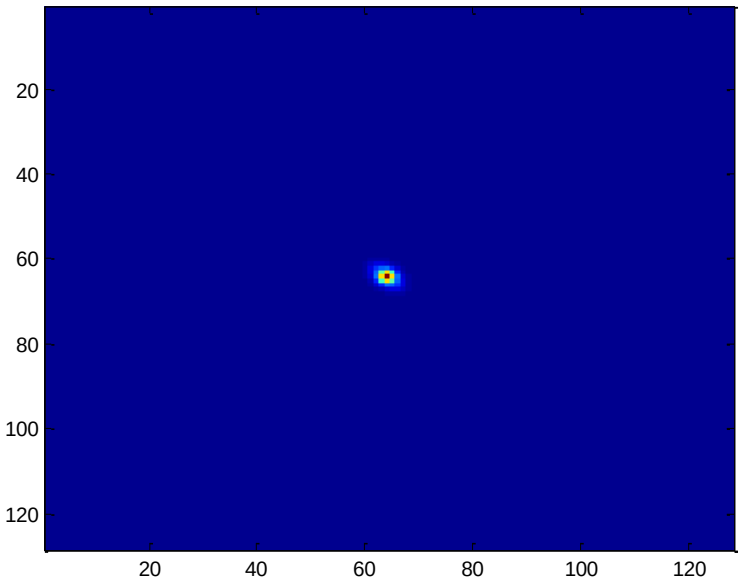

Figure 29: Image of Three Object Scenario, the two dim objects are 100 photons and are spaced 4x4 pixels apart with the bright object is 10,00 photons halfway between them and an average background level of 10 photons per pixel.

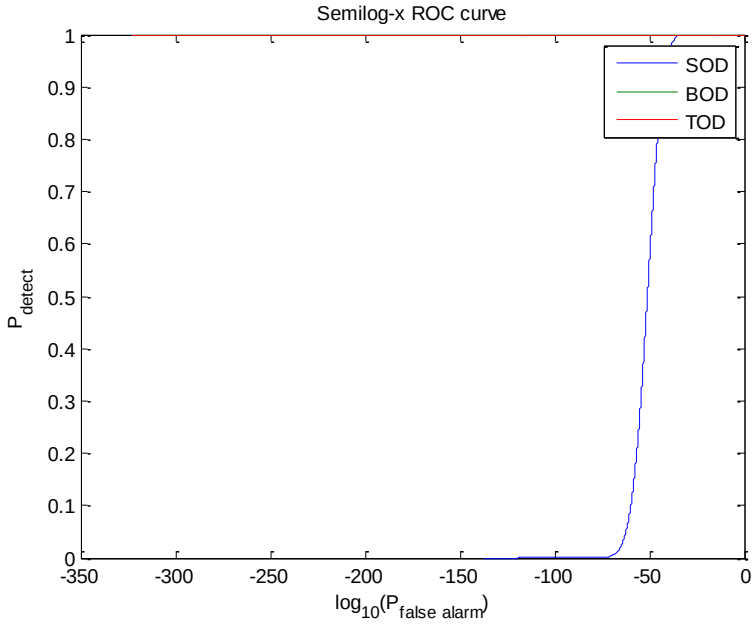

Figure 30: Semilog-x ROC Curves for Three Object Scenario, the two dim objects are 100 photons and are spaced 4x4 pixels apart with the bright object is 10,00 photons halfway between them and an average background level of 10 photons per pixel.

Varying Parameters

The last simulation that was run was one in which the entire simulation was run multiple times at each distance. They started at a distance of one pixel, and went to 10 pixels (so that it exceeded a full PSF width apart). For this, a fixed brightness ratio of 100:1 was used in all experiments. 20 simulations were run at each distance, with different random noise generated in each scenario. In all cases, Source Extractor had a higher area under the curve, but had a much lower detection rate. BOD's detection rate was better in all cases, although as the points get gets farther apart, the advantage diminishes.

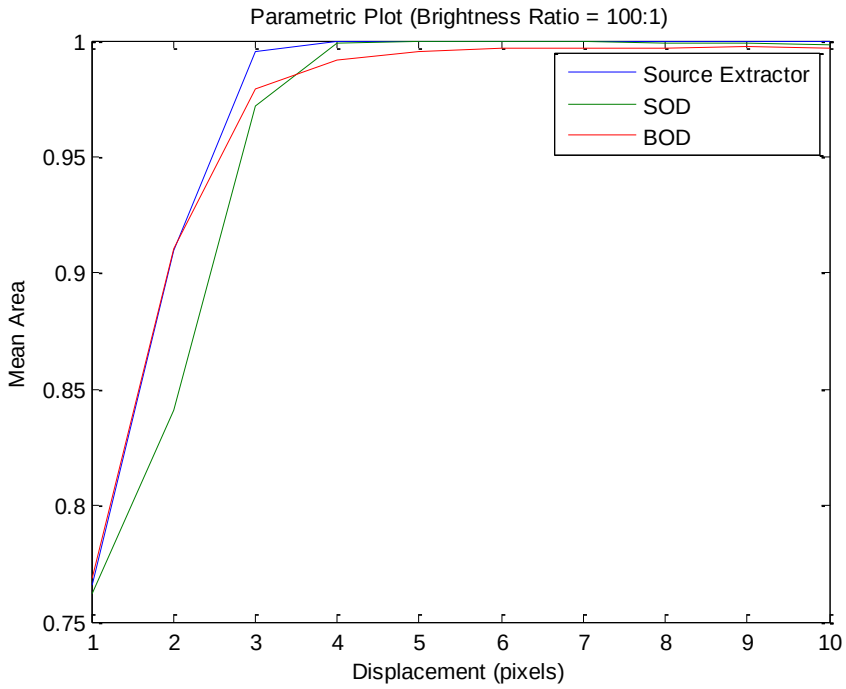

Figure 31:ROC Curve Areas for Varying Distance Simulations with 1:100 Brightness Ratio

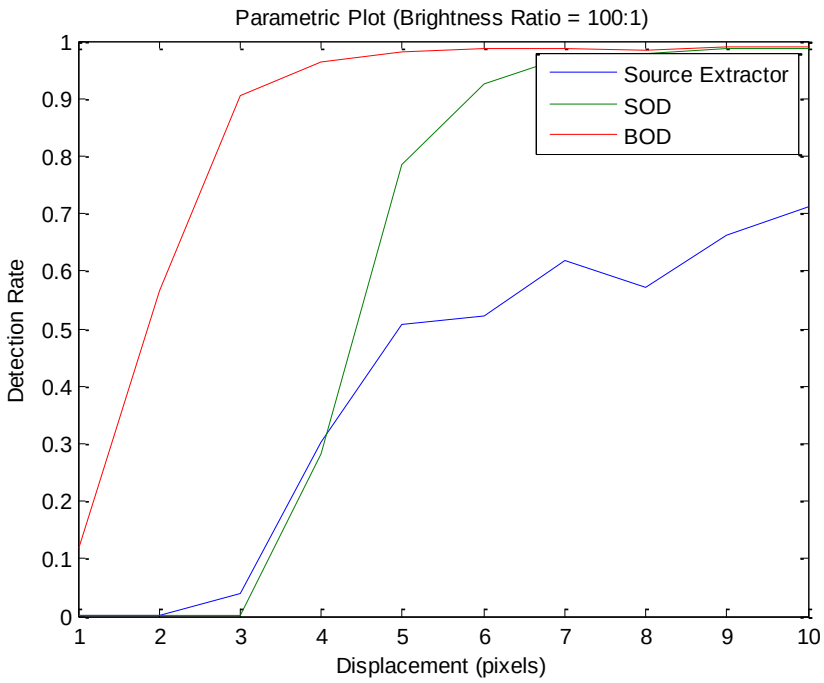

Figure 32: Detection Rates for Varying Distance Simulation for 1:100 Brightness Ratio.

Finally a series of experiments were run where the distance was held constant at 2x2 pixels and the brightness was varied. The number of trials were identical to that above except that each trial had a greater brightness difference than the once before. The ROC curve areas for each case are shown below in Figure 33 and the detection rates are shown in Figure 34. In this case the brightness ranged from 10:1 (which is displayed in Figure 33 and Figure 34 as 0.1) to 100:1 (which is displayed in Figure 33 and Figure 34 as 0.01). The bright object still remained 10,000 photons in all cases and the background noise stayed at an average of 10 photons per pixel. While the Source Extractor method had a higher area in most cases, its detection rate was actually negligible in all cases.

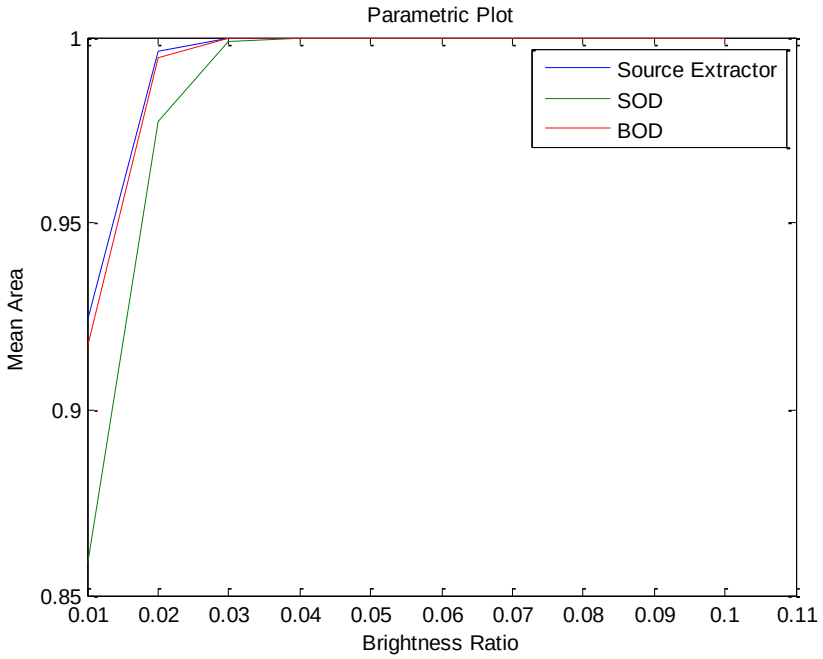

Figure 33: ROC Curve Areas for Varying Brightness Simulation with 2x2 pixel offset.

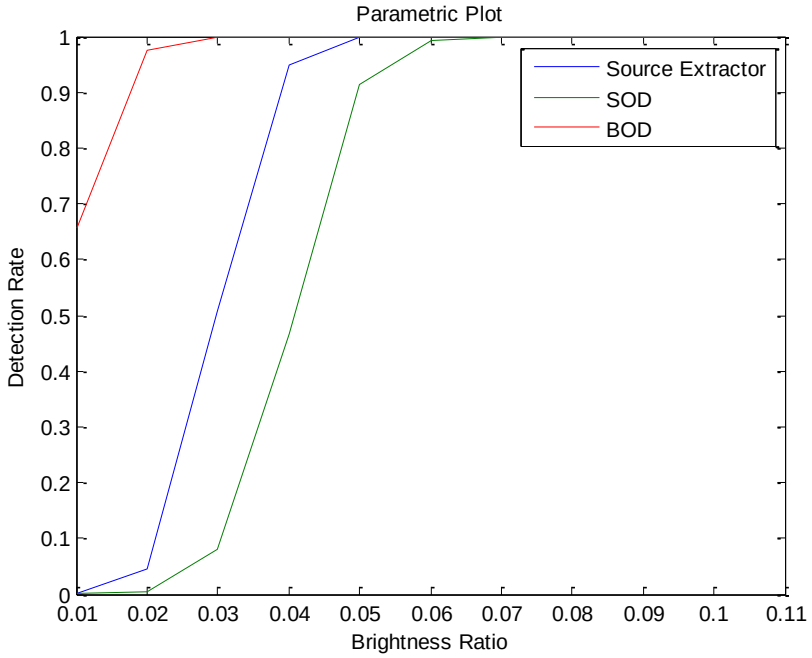

Figure 34: Detection Rates for Varying Brightness Simulation with 2x2 pixel offset.

Summary

In all two object cases, the BOD method performs equal to or better than the SOD method. However, SOD did perform better than expected, especially in the 10:1 brightness cases. In some cases it was necessary to look at the plot as a semilog-x scenario in order to be able to visually discern the differences between the two ROC curves. Source Extractor demonstrated remarkably poor detection rates in all scenarios, when compared with BOD, or even SOD. As expected, the farther apart the two objects were the closer the results of the two methods, and the greater the difference in brightness, the greater the difference in the results of the two methods. The brightness seemed to play a greater role than proximity, in determining how much better BOD would perform than SOD. TOD performed as expected, beating out BOD and SOD in both three object cases in ROC curve area, as well as detection rate. The first test scenario demonstrated that, outside of one PSF-width, SOD and BOD will bot h perform almost identically.

V. Conclusions and Recommendations

Chapter Overview

This chapter will begin by reviewing the results of the study as well as drawing conclusions from the results. Next it will be used to comment upon the significance of these results. Then recommendations will be made for action based upon these conclusions. Finally recommendations will be made for continued research in this area.

Conclusions of Research

Reviewing the results, it should be obvious that BOD outperforms SOD in general. In the first laboratory experiment, they both performed similarly, due to the lack of PSF masking occurring. Therefore it can be concluded that BOD is a superior method for detecting binary images than the methods currently in use. Additionally, while BOD can detect a third object easier than SOD, BOD is also easily expanded into TOD, which is superior to both methods, for detecting three object clusters.

Significance of Research

Comparing BOD to other methods, such as Pan-starrs, this method creates its own "master image" rather than depending on what that had been created a previous night. The advantage of recreating its own image each time is that it will not be adversely affected if the image changed. A disadvantage is that it will take slightly longer to process the image, since two scans will be necessary.

Another advantage of BOD over SOD deals with detecting more than two objects. The previous binary detection method also fails to perform in this area. While BOD can outperform SOD in detecting three objects on its own, it can easily be expanded to scan for additional objects, such as using TOD for a three object cluster. It would be fairly simple, should the need arise, to expand this to as many terms as necessary. It is important to acknowledging that an additional scan would be necessary for each additional object to be scanned for which could be time consuming if the number of clustered objects is too high.

Recommendations for Action

Since this method has not only been tested on actual data, but has outperformed existing methods of binary object detection, it would be logical to implement this algorithm on SSA platforms. This would give us the ability to potentially detect a spy satellite in close proximity to one of our orbital assets, as well as detecting space debris that might be clustered together. Without the ability to reliably detect these threats we would have no means of countering them. Since this can be accomplished without the use of adaptive optics or additional hardware and since Matlab is a fairly universal software tool, this methods implementation should be fairly straightforward and cost effective.

Recommendations for Future Research

The next step for this project could be code optimization. Due to the way the code calculates each pixel independently of the others, parallelization should substantially decrease computation time for implementation. Many other function might

be able to be cleaned up to increase the programs speed as well. Perhaps an investigation from a computer science point of view might be the most valuable next step.

Summary

While methods currently exist for detecting objects in GEO, these methods have difficulty detecting binary objects, such as a spy satellite hiding near one of our satellites, or space debris near one of our assets. This is especially notable when one object is much dimmer than the other. However, the BOD method outlined in this paper is not only adept at detecting objects in these scenarios, but can easily be expanded into new algorithms should the need arise to detect more than two objects clustered together. Since it does not rely upon a previously created master image, it is also more able to adapt to changes than other current methods. Additionally, since it does not depend on any additional hardware it should be relatively inexpensive to implement. Therefore, it would be logical to use BOD with current SSA assets.

Appendix

```
clear all
close all
clc
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%Begin testing for Probability of Detection
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
testsize = 20;   %sets the number of "tests" to run

for s = 1:testsize
%user defined variables
N = 128;             %size of source field matrix
bn = 10;             %level of background noise
Offset = -2;          %distance from central object to tertiary
object(in
                     %pixels of source plane)
Offset2 = 2;         %distance from central object to the object you are
                     %looking for (in pixels of source plane)
star = 10000;        %brightness of center object
samples = 100;       %number of expirements
star2 = 0;           %brightness of tertiary object
                     %NOTE: set equal to 0 for binary object scenario
star3 = star/100;     %brightness of the object you are looking for
R0 = 20;              %seeing parameter for atmospheric turbulance

binary=bn*ones(N,N); % make source plane, with background noise

pos = floor(N/2);        %defines the center of the matrix
binary(pos,pos)=star+bn;   %places bright star at center of source
plane
binary(pos+Offset,pos+Offset)=star2+bn;    %places dim star at offset
binary(pos+Offset2,pos+Offset2)=star3+bn;  %places third object in
plane

otf1=Make_otf(64,0,N,1,zeros(N,N));     %creates OTF of telescope pupil

otf2=Make_long_otf(50,50/128,N,R0);     %creates atmospheric OTF
                                        %NOTE: assumes long exposure

tot_otf=otf1.*fftshift(otf2);           %computes total OTF of system
                                        %NOTE: uses Frouenhoffer
                                        %approximation

data_mean=real(ifft2(fft2(binary).*tot_otf));  %computes image

testsum = sum(sum(data_mean));          %sanity check for total
photocount
                                        %should be
star+star2+star3+N^2*bn
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%Set up for Binary Hypothesis Test
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%sets up denominator for "null" hypothesis
x0 = zeros(N,N);
x0(pos,pos) = 1;
xf = fft2(x0);

%sets up denominator for "detection" hypothesis
x2 = zeros(N,N);
x2(pos+Offset,pos+Offset) = 1;
xf2 = fft2(x2);

x3 = zeros(N,N);
x3(pos+Offset2,pos+Offset2) = 1;
xf3 = fft2(x3);

%sets up the "bases" for the two hypothesis, base = one object, base2 =
%second object.  For three object scenarios, the third object is
ignored
%since this detection algorithm only looks for one object.
base = real(ifft2(tot_otf.*xf));
base2 = real(ifft2(tot_otf.*xf2));
base3 = real(ifft2(tot_otf.*xf3));

brightS = ones(samples,1)*star;  %sets up initial guess of dim object
gammaS = zeros(samples,1);   %initializes matrix of PH1/PH0 in logspace

bright = ones(samples,1)*star;  %sets up initial guess of dim object
gamma = zeros(samples,1);   %initializes matrix of PH1/PH0 in logspace


figure(4)
imagesc(data_mean)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%Begin testing for Probability of Detection
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
for i=1:samples

data(:,:,i)=poissrnd(data_mean);

%begins iterative process of determining brightness of dim object (if
there
%were one at this location).  n=number of iterations
for n = 1:1000
    iiS = real(base3*brightS(i)+bn);
```

```
    brightS(i) = brightS(i).*sum(sum(real(data(:,:,i)./iiS).*base3));

    ii = base*star+base3*bright(i)+bn;
    bright(i) = bright(i).*sum(sum(real(data(:,:,i)./ii).*base3));

end

%Assigns the two hypotheses (PH1 = probability that threre is a second
%object, PH0 is the probability that there is not)
PH1S = sum(sum(data(:,:,i).*log(bn+brightS(i)*base3)))...
    -brightS(i)-bn*N^2;
PH0S = sum(sum(data(:,:,i).*log(bn)))-bn*N^2;

PH1 = sum(sum(data(:,:,i).*log(bn+star*base+bright(i)*base3)))...
    -star-bright(i)-bn*N^2;
PH0 = sum(sum(data(:,:,i).*log(bn+star*base)))-star-bn*N^2;

%gamma = log likelihood ratio
gammaS(i) = PH0S/PH1S;

gammaSa(i)=sum(sum((data(:,:,i)-bn).*base3));

gamma(i) = PH0/PH1;

%This segment is used to display progress of computation
disp([num2str(i/samples*100),'% done with pdetect, first half of test
',num2str(s),' out of ',num2str(testsize)])
end




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%Begin testing for Probability of False Alarm
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%


%remove second star from binary field (replaces cell with background
noise)
binary(pos+Offset2,pos+Offset2)=bn;

data_mean=real(ifft2(fft2(binary).*tot_otf));  %creates new image

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%Set up for Binary Hypothesis Test
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
```

```
brightS = ones(samples,1)*star;  %resets guess of dim object
gamma2S = zeros(samples,1);   %reinitializes matrix of PH1/PH0 in
logspace
%begins null experiment

gamma2Sa = zeros(samples,1);   %reinitializes matrix of PH1/PH0 in
logspace
%begins null experiment

for i=1:samples

data(:,:,i)=poissrnd(data_mean);

%begins iterative process of determining brightness of dim object (if
there
%were one at this location).  n=number of iterations. 'base2' remains
the
%same as in the detection section
for n = 1:1000
    iiS = base3*brightS(i)+bn;
    brightS(i) = brightS(i).*sum(sum(real(data(:,:,i)./iiS).*base3));

    ii = base*star+base3*bright(i)+bn;
    bright(i) = bright(i).*sum(sum(real(data(:,:,i)./ii).*base3));

end
%Assigns the two hypotheses (PH1 = probability that threre is a second
%object, PH0 is the probability that there is not)
PH1S = sum(sum(data(:,:,i).*log(bn+brightS(i)*base3)))...
    -brightS(i)-bn*N^2;
PH0S = sum(sum(data(:,:,i).*log(bn)))-bn*N^2;

PH1 = sum(sum(data(:,:,i).*log(bn+star*base+bright(i)*base3)))...
    -star-bright(i)-bn*N^2;
PH0 = sum(sum(data(:,:,i).*log(bn+star*base)))-star-bn*N^2;

%gamma = log likelihood ratio (inverted because we are in log
likelihood)
gamma2S(i) = PH0S/PH1S;

gamma2Sa(i)=sum(sum((data(:,:,i)-bn).*base3));

gamma2(i) = PH0/PH1;

%This segment is used to display progress of computation
disp([num2str(i/samples*100),'% done with pfa, second half of test
',num2str(s),' out of ',num2str(testsize)])
end

detected_brightness = mean(brightS);
```

```
H = lillietest(gammaS); %test for normality of data
mu = (mean(gammaS));    %the mean threshold value for detection
stdev = (std(gammaS));  %the standard deviation for threshold values
H2 = lillietest(gamma2S); %test for normality of data
mu2 = (mean(gamma2S));    %the mean threshold value for detection
stdev2 = (std(gamma2S));  %the standard deviation for threshold values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%                      Calculating The threshold for SOD
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

%calculates possible endpoints of threshold array
t1 = mu+10*stdev;
b1 = mu-10*stdev;
t2 = mu2+10*stdev2;
b2 = mu2-10*stdev2;
%turns endpoints into an array and sorts them
threshmat = sort([t1 b1 t2 b2]);
%finds the smaller of the two standard deviations
stepsize = sort([stdev stdev2]);

%assigns the lowest of the above endpoints as the minimum threshold
value,
%the highest as the maximum value, and the smallest of the two standard
%deviations as the stepsize to create the threshold array
threshS = [threshmat(1):stepsize(1)/100:threshmat(4)];
%computes probability of false alarm per threshold
pfaS = cdf('norm',threshS,mu2,stdev2);
%computes probability of detection per threshold
pdetectS = cdf('norm',threshS,mu,stdev);

%finds the 10% probability of false alarm level.  Doesn't always work
right
%based upon the exact value of pfa(end10) (the larger pfa(end10) is,
the
%less accurate the result)
end10S = find(pfaS>10^-9,1);
pfa10S = pfaS(1:end10S);
pdetect10S = pdetectS(1:end10S);
detection_rateS(s) = pdetectS(end10S)

%computes the area under the curve for this experiment
firstS = find(pfa10S>0,1);
ROC_intS(s) = trapz(pfaS,pdetectS);
LOG_ROC_intS(s) = trapz(log10(pfaS(firstS:end)),pdetectS(firstS:end));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%                      Calculating The threshold for "old-SOD"
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
```

```
detected_brightness = mean(brightS);
H = lillietest(gammaSa); %test for normality of data
mu = (mean(gammaSa));    %the mean threshold value for detection
stdev = (std(gammaSa));  %the standard deviation for threshold values
H2 = lillietest(gamma2Sa); %test for normality of data
mu2 = (mean(gamma2Sa));    %the mean threshold value for detection
stdev2 = (std(gamma2Sa));  %the standard deviation for threshold values

%calculates possible endpoints of threshold array
t1 = mu+10*stdev;
b1 = mu-10*stdev;
t2 = mu2+10*stdev2;
b2 = mu2-10*stdev2;
%turns endpoints into an array and sorts them
threshmat = sort([t1 b1 t2 b2]);
%finds the smaller of the two standard deviations
stepsize = sort([stdev stdev2]);

%assigns the lowest of the above endpoints as the minimum threshold
value,
%the highest as the maximum value, and the smallest of the two standard
%deviations as the stepsize to create the threshold array
threshSa = [threshmat(1):stepsize(1)/100:threshmat(4)];
%computes probability of false alarm per threshold
pfaSa = 1-cdf('norm',threshSa,mu2,stdev2);
%computes probability of detection per threshold
pdetectSa = 1-cdf('norm',threshSa,mu,stdev);

%finds the 10% probability of false alarm level.  Doesn't always work
right
%based upon the exact value of pfa(end10) (the larger pfa(end10) is,
the
%less accurate the result)
end10Sa = find(pfaSa>10^-9,1);
pfa10Sa = pfaSa(1:end10Sa);
pdetect10Sa = pdetectSa(1:end10Sa);
detection_rateSa(s) = pdetectS(end10Sa)

%computes the area under the curve for this experiment
firstSa = find(pfa10S>0,1);
ROC_intSa(s) = -trapz(pfaSa,pdetectSa);
LOG_ROC_intSa(s) =
trapz(log10(pfaSa(firstSa:end)),pdetectSa(firstSa:end));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%                     Calculating the Threshold for BOD
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

detected_brightness = mean(bright);
H = lillietest(gamma); %test for normality of data
```

```
mu = (mean(gamma));    %the mean threshold value for detection
stdev = (std(gamma));  %the standard deviation for threshold values
H2 = lillietest(gamma2); %test for normality of data
mu2 = (mean(gamma2));    %the mean threshold value for detection
stdev2 = (std(gamma2));  %the standard deviation for threshold values


%calculates possible endpoints of threshold array
t1 = mu+10*stdev;
b1 = mu-10*stdev;
t2 = mu2+10*stdev2;
b2 = mu2-10*stdev2;
%turns endpoints into an array and sorts them
threshmat = sort([t1 b1 t2 b2]);
%finds the smaller of the two standard deviations
stepsize = sort([stdev stdev2]);

%assigns the lowest of the above endpoints as the minimum threshold
value,
%the highest as the maximum value, and the smallest of the two standard
%deviations as the stepsize to create the threshold array
thresh = [threshmat(1):stepsize(1)/100:threshmat(4)];
%computes probability of false alarm per threshold
pfa = cdf('norm',thresh,mu2,stdev2);
%computes probability of detection per threshold
pdetect = cdf('norm',thresh,mu,stdev);

%finds the 10% probability of false alarm level.  Doesn't always work
right
%based upon the exact value of pfa(end10) (the larger pfa(end10) is,
the
%less accurate the result)
end10 = find(pfa>10^-9,1);
pfa10 = pfa(1:end10);
pdetect10 = pdetect(1:end10);
detection_rate(s) = pdetect(end10)

%computes the area under the curve for this experiment
first = find(pfa10>0,1);
ROC_int(s) = trapz(pfa,pdetect);
LOG_ROC_int(s) = trapz(log10(pfa(first:end)),pdetect(first:end));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%                    Computing the statistical outputs
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%the statistics for the area under the ROC curves:
mean_area_Single = mean(ROC_intS)
std_area_Single = std(ROC_intS)
```

```
%the statistics for the detection rate at 10% probability of false
alarm:
mean_detection_rate_10_Single = mean(detection_rateS)
std_detection_rate_10_Single = std(detection_rateS)

%the statistics for the area under the ROC curves:
mean_area_Old_SOD = mean(ROC_intSa)
std_area_Old_SOD = std(ROC_intSa)

%the statistics for the detection rate at 10% probability of false
alarm:
mean_detection_rate_10_Old_SOD = mean(detection_rateSa)
std_detection_rate_10_Old_SOD = std(detection_rateSa)

%the statistics for the area under the ROC curves:
mean_area = mean(ROC_int)
std_area = std(ROC_int)

%the statistics for the detection rate at 10% probability of false
alarm:
mean_detection_rate_10_Double = mean(detection_rate)
std_detection_rate_10_Double = std(detection_rate)



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%                            Produces the Plots
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

%plots ROC curve on semilog-x plot
figure(1)
plot(log10(pfaSa),pdetectSa,log10(pfaS),pdetectS,log10(pfa),pdetect)
legend('Source Extractor','SOD','BOD')
title('Semilog-x ROC curve')
xlabel('log_1_0(P_f_a_l_s_e _a_l_a_r_m)')
ylabel('P_d_e_t_e_c_t')


%looks at the false alarm detection and detection probability based on
%threshold
figure(2)
subplot(2,1,1)
plot(threshS,pfaS,thresh,pfa)
legend('SOD','BOD')
title('False Alarm Rate')
xlabel('Threshold')
ylabel('P_f_a_l_s_e _a_l_a_r_m')
subplot(2,1,2)
plot(threshS,pdetectS,thresh,pdetect)
legend('SOD','BOD')
```

```
title('Object Detection Rate')
xlabel('Threshold')
ylabel('P_d_e_t_e_c_t')

%plots ROC curve
figure(3)
plot(pfaSa,pdetectSa,pfaS,pdetectS,pfa,pdetect)
legend('Soure Extractor','SOD','BOD')
title('ROC Curve')
xlabel('P_f_a_l_s_e _a_l_a_r_m')
ylabel('P_d_e_t_e_c_t')
```

Bibliography

[1] B. Gessel and S. Cain, "Effects of star crossings on the detection of dim objects in orbit and mitigation strategies for improving detection," in *Proceedings of the SPIE, Vol. 9054*, Baltimore, 2014.

[2] E. Bertin and S. Arnouts, "SExtractor: Software for source extraction," *Astronomy & Astrophisics Supplement*, pp. 317, 393, 1996.

[3] J.W.Goodman, Statistical Optics, New York: John Wiley and Sons, 2000.

[4] R. D. Richmond and S. C. Cain, Direct-Detection LADAR Systems, Bellingham, WA: SPIE, 2010.

[5] C. J. R. Peterson, "Near Earth Object Detection Using a Poisson Statistical Model for Detection on Images Modeled From the Panoramic Survey Telescope & Rapid Response System," Air Force Institute of Technology, Wright Patterson Air Force Base, Ohio, 2014.

[6] G. W. Marcy, R. P. Butler, D. Fischer, S. S. Vogt, J. J. Lissauer and E. J. Rivera, "A Pair of Resonant Planets Orbiting GJ 876," *The Astrophysical Journal*, vol. 556, no. July 20, pp. 296-301, 2001.

[7] D. Charbonneau, T. M. Brown, D. W. Latham and M. Mayor, "Detection of Planetary Transits Across a Sun-like Star," *The Astrophysical Journal*, vol. 529, no. January 20, pp. L45-L48, 2000.

[8] B. H. Gessel, "Binary Detection Using Multi-hypothesis Log-Likelihood, Image Processing," Air Force Institute of Technology, Wright Patterson Air Force Base, Ohio, 2012.

[9] H. Lilliefors, "On the Kolmogorov-Smirnov test for Normality with Mean and Variance Unknown," *Journal of the American Statistical Association, Vol 62*, vol. 62, no. April 10, pp. 399-402, 1967.

[10] A. M. Catarius, "Static Scene Statistical Non-Uniformity Correction," Air Force Institute of Technology, Wright Patterson Air Force Base, Ohio, 2015.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to an penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
24-03-2016	Master's Thesis	March 2015 – March 2016

TITLE AND SUBTITLE

Geosynchronous Binary Object Detection

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Cunningham, Patrick B. Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-16-M-010

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Intentionally left blank

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

DISTRUBTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

This paper will compare two competing methods for optically detecting binary objects. This is mostly intended for use in Space Situational Awareness (SSA), though has the potential to be used in other applications. The first method, "point detection" is a versatile algorithm which is currently used to detect extraterrestrial objects. However, it does not take into account interference by a nearby object. Therefore a second algorithm is investigated, referred to as "binary object detection", which does. The binary detection algorithm proved to have a superior Receiver Operating Characteristic (ROC) curve (based upon the area under the curve) in all cases, not just when the two points were near. The algorithm was tested with both simulated and measured data containing single points and binary objects.

15. SUBJECT TERMS

Space Situational Awareness, Geosynchronous Object Detection, Binary Object Detection, Space Domain Awareness

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Stephen Cain, AFIT/ENG
U	**U**	**U**	**UU**	80	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-6565, ext 4716 Stephen.Cain@afit.edu)

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18