ENHANCED SPACE OBJECT DETECTION WITHOUT PRIOR KNOWLEDGE OF THE POINT SPREAD FUNCTION

THESIS

Grant F. Graupmann, Captain, USAF

AFIT-ENG-MS-21-M-042

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-21-M-042

ENHANCED SPACE OBJECT DETECTION WITHOUT PRIOR KNOWLEDGE OF THE POINT SPREAD FUNCTION

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

Grant F. Graupmann, B.S.E.E.

Captain, USAF

March 2021

AFIT-ENG-MS-21-M-042

ENHANCED SPACE OBJECT DETECTION WITHOUT PRIOR KNOWLEDGE OF THE POINT SPREAD FUNCTION

THESIS

Grant F. Graupmann, B.S.E.E.

Captain, USAF

Committee Membership:

Dr. Stephen C. Cain, PhD

Chair

Maj David Becker, PhD

Member

Maj Ronald Aung, PhD

Member

Abstract

The Space domain is becoming increasingly more important with each passing year. For decades, object detection in space has been done so by some organizations using a simple algorithm, the point detector. This detection algorithm is used more so to reduce the amount of false positive images of expected space objects collected.

Since the point detector was created, other detection algorithms have been created that increase the probability of detection, while still keeping the same probability of false alarm. The main difference between the point detector and other detection algorithms is that the point detector does not need to know the point-spread function (PSF) of the object it is looking for.

The matched filter correlator (MFC) detector has been used in many studies, and is reliant on prior knowledge of the PSF. This has been an issue in cases where the PSF information is potentially inaccurate or unknown.

This thesis utilizes MFC detector in a manner that it has never been used before, along with a new detection algorithm, the Pearson's correlation coefficient (PCC) detector, in order to estimate Fried's Seeing Parameter (r_0) for a captured image

This new method of estimating r_0 could yield higher probability of detection rates among certain space objects with little or no prior knowledge about the space object in question.

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Dr. Stephen Cain, for his incredible guidance and support throughout the course of this thesis effort. His insight and experience were invaluable to my research.

I would also like to thank my wife for her unwavering support throughout all the long days and late nights of research, writing, and testing.

Grant F. Graupmann

Table of Contents

List of Figures

Page

List of Tables

Page

ENHANCED SPACE OBJECT DETECTION WITHOUT PRIOR KNOWLEDGE OF THE POINT SPREAD FUNCTION

I. Introduction

The Space domain is becoming increasingly more important with each passing year. For decades, object detection in space by some organizations has been done so using the point detector. This detection algorithm is used more so to reduce the amount of false positive images of space objects collected, also known as a false alarm. However, by reducing the probability of false alarm, the probability of detection for certain objects, especially those that are fainter, are reduced greatly. Because one goal of the space community is to continually get better at detecting space objects, this algorithm has very limited uses.

Since the point detector was created, other detection algorithms have been created that increase the probability of detection, while still keeping the same probability of false alarm. The main difference between the point detector and other detection algorithms is that the point detector does not need to know the point-spread function (PSF) of the object it is looking for. But what if the other detection algorithms did not require prior knowledge of the PSF either? The matched filter correlator (MFC detector) detector has been used in many studies, but is reliant on prior knowledge of the PSF. This has been

an issue in cases where the PSF information is potentially inaccurate or unknown.

With space domain awareness (SDA) becoming increasingly more important, detection algorithms must continue to improve if America's new United States Space Force (USSF) plans to dominate the space domain. This improvement in detection can be accomplished through a creation of new detection algorithms, by enhancing current detection techniques, or by utilizing already established algorithms through different means than what they were intended to be used for to achieve a desired result.

1.1 Motivation

Robust and reliable operations in the space domain are increasingly important to the United States (U.S.). According to the 2011 U.S. National Security Space Strategy, "Space is vital to U.S. national security and our ability to understand emerging threats, project power globally, conduct operations, support diplomatic efforts, and enable global economic viability." [1]. The U.S. National Space Policy also expressed the importance to develop technologies to "detect, identify, and attribute actions in space that are contrary to responsible use and the long-term sustainability of the space environment" [2]. Finally, the U.S. Congress mandated that the National Aeronautics and Space Administration (NASA) coordinate with the

Department of Defense (DoD) and all other organizations to catalogue 90% of all asteroids and comets larger than 140m that are in a trajectory close to earth by the year 2020 [3]. With such a massive importance put on detecting objects in space, any research that can make gains in this area is valuable to the U.S.

This thesis will be focused on estimating Fried's Seeing Parameter (r_0) for an image with no prior knowledge of the PSF. Three different detection algorithms will be thoroughly explored, the point detector, the MFC detector, and a newly created Pearson's correlation coefficient (PCC detector) detector. The point detector has been used to conduct space object (SO) detection for decades among many in the space community. However, in order to reduce the probability of false alarm to an acceptable level, the probability of detection is also reduced, which leads to imaging systems missing some of the dimmer or smaller SO.

In recent years, new detection algorithms, such as the MFC detector, have been created in order to increase the probability of detection while maintaining the same probability of false alarm. When comparing the MFC detector to the point detector, it is not a fair comparison because the MFC detector assumes there is prior knowledge of the PSF in question, while the point detector has no such requirement. In this comparison, the MFC detector, along with the PCC detector will attempt to estimate the r_0 of the PSF in order to even the playing field between these algorithms and the point

detector. Peak correlation and signal-to-noise ratio (SNR) will be metrics of comparison.

1.2 Thesis Overview

Accurately estimating the seeing parameter of an unknown PSF has never been considered when utilizing the MFC detector and could allow the space community the ability to increase the probability of detection compared to the current detection method.

Chapter II dives deep into the background information required to understand current detection algorithms, as well as create a new detection algorithm. The chapter is wrapped up with a preliminary comparison between the three detection algorithms used during this study.

Chapter III explains how the detection algorithms are used with simulated data in order to get the results for this study. This chapter also covers how probability of detection and probability of false alarm are calculated for all algorithms used in this research, and explains how the results of the simulated data are analyzed for both the MFC detector and the PCC detector. The chapter is summed up with a discussion of the results from the simulated data.

Chapter IV follows trends from Chapter III, but instead of using simulated data, measured data sets are used. Although the code used for both simulated and measured data have many similarities, there are also

some differences, which are called out in this chapter. Results of the measured data are covered to conclude this chapter.

Chapter V will sum up all of Chapter IV's results into conclusions and key takeaways from the comparisons of the three detection algorithms. Potential for future work is also discussed.

II. Literature Review

Chapter Overview

The purpose of this chapter is to go over the Poisson distribution, fundamentals of Fourier optics [4], and the probability of detection and the probability of false alarm will be defined in regards to space object detection. Finally, this chapter will delve into the three detection algorithms used in this research: the point detector [5], MFC detector [6], and PCC detector.

2.1) The Poisson Distribution

"The Poisson random variable is extremely important as it describes the behavior of many physical phenomena… the Poisson random variable plays a fundamental role in our development of a probabilistic description of noise" [7]. Noise is present in every image taken of space, and thus will be adequately represented in all simulations and measured data throughout this study.

The Poisson distribution expresses the probability of a certain number of events occurring within a fixed time with each event being independent of the next. When looking at a captured image, each pixel is considered independent of the rest, and the noise within each pixel is random.

No manipulations of the Poisson distribution are required for this study, rather just an understanding of why image data is considered Poisson

when making assumptions for the PCC detector. The probability mass function of the Poisson distribution can be described with (1) below [7].

$$P_X(k) = \begin{cases} \frac{b^k}{k!} e^{-b}, & k \geq 0 \\ 0, & k < 0 \end{cases} \tag{1}$$

Where a discrete random variable (X) has a Poisson distribution with a mean value (b) greater than zero for realization values k = 0, 1, 2… It is important to note that b is equal to the expected value (mean) of X and also the variance. These principles of the Poisson distribution will be crucial for simplifying equations used within the PCC detector later on in Chapter II.

2.2) Fundamentals of Fourier Optics

Fourier optics is the study of optics using Fourier transforms (FTs), in which the waveform being considered is thought to be made up of a combination of plane waves. "The Fourier transform is perhaps the most important analytical tool needed for work in statistical optics, or for that matter in the field of modern optics in general" [8]. Goodman defines the one-dimensional and two-dimensional Fourier transforms below in (2) & (3).

$$F(\nu) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi\nu x} dx \tag{2}$$

$$F(\nu_x, \nu_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(\nu_x x + \nu_y y)} dx dy \tag{3}$$

Where (ν_x, ν_y) is the spatial frequency in the x and y directions, $f(x, y)$ is the signal undergoing a Fourier Transform, and $e^{j2\pi(\nu_x x+\nu_y y)}$ is the phase component.

The inverses of (2) and (3) are shown in (4) and (5).

$$f(x) = \int_{-\infty}^{\infty} F(\nu) e^{j2\pi\nu x} d\nu \tag{4}$$

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(\nu_x, \nu_y) e^{j2\pi(\nu_x x+\nu_y y)} d\nu_x d\nu_y \tag{5}$$

There are a number of relationships that can be useful to manipulate Fourier transforms that Goodman provides the reader. Many of these are useful, but this research will highlight the two-dimensional autocorrelation theorem shown in (6).

$$\mathcal{F}\left\{ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(\xi, \chi) h^*(\xi - x, \chi - y) d\xi d\eta \right\} = \left| H(\nu_x, \nu_y) \right|^2 \tag{6}$$

The h represents a function of one or two variables, H represents the Fourier transforms of h and (ξ,χ) are spatial variables. This manipulation is used in both the MFC detector and PCC detector detection algorithms, which will be discussed later in Chapter II. The significance of the Fourier transform in space object detection comes into play when discussing PSFs and optical transfer functions (OTF).

A PSF is defined as an imaging system's impulse response, or single point object. It is the spatial domain version of the OTF of an incoherent imaging system. The OTF specifies how the spatial frequencies are handled by the imaging system. Goodman defines an OTF with (7) below [4].

$$H\left(fx, fy\right) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left|h\left(u,v\right)\right|^2 e^{-j2\pi\left(fxu+fyu\right)} dudv}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left|h\left(u,v\right)\right|^2 dudv} \tag{7}$$

Where $h(u,v)$ is the field impulse response at image coordinates (u,v). The numerator is in the form of a Fourier transform, and the denominator normalizes the function. Converting an OTF into a PSF is then as simple as performing an inverse Fourier transform on the OTF. This relationship is exploited within the MATLAB code in order to create PSFs for simulations conducted for this research. See the Appendix for the full MATLAB code.

2.3) Atmospheric Effects & Fried's Seeing Parameter (r_0)

Atmospheric turbulence is an irregular air motion caused by winds that vary in both speed and direction. This turbulence reduces image quality of objects being viewed through even the most advanced optical systems. Fried's seeing parameter is a measure of the quality of optical transmission through the atmosphere due to random inconsistencies in the atmosphere's refractive index. This is mainly caused by temperature variations, of both

small and large scales. The Fried's seeing parameter is typically a measure of length in units of centimeters and can be defined by :

$$r_0 = 0.185\left(\frac{\bar{\lambda}^2}{zC_n^2}\right)^{3/5} \tag{8}$$

In this equation C_n^2 is the atmospheric turbulence strength, $\bar{\lambda}$ is the average wavelength of the light source, and z is the distance between the light source and the aperture. Telescopes with apertures smaller than r_0 are less affected by atmospheric seeing than diffraction due to the telescope's small aperture. However, the resolution of telescopes with apertures much larger than r_0 will be limited by atmospheric turbulence. For the purposes of this study, it is assumed the system is a ground-based telescope, and a r_0 value that spans from 0-25 cm, which covers the majority of all ground based seeing parameter values viewable from Earth's surface. These r_0 values can then be used to help solve for the OTF of long-exposure atmosphere as shown in (9).

$$H_{atm}(u_2) = e^{-3.44\left(\frac{\bar{\lambda} f u_2}{r_0}\right)^{5/3}} \tag{9}$$

In this equation, $\bar{\lambda}$ is, again, the average wavelength of the light source, u_2 is the radial spatial frequency, f is the focal length, and r_0 is the

Fried's seeing parameter [8]. The use of this Equation will be discussed in Chapter III.

2.4) Aperture & Zernike Polynomials

The aperture function is crucial for determining the image quality of an optical system. Any flaws or aberrations within it can have significant effects on the captured images. Aberrations can be described through Zernike polynomials, which will produce the refraction error. These polynomial functions are defined over a circular support area, typically the pupil planes in optical imaging systems made up of lenses and mirrors of a certain finite diameter. Zernike polynomials are orthogonal to one another and are used to parameterize specific phase aberrations. Each polynomial carries coefficients to weight their respective type of aberration. The types of aberrations that Zernike polynomials can describe are piston, tilt, focus, astigmatism, coma, and more shown in Figure 1.

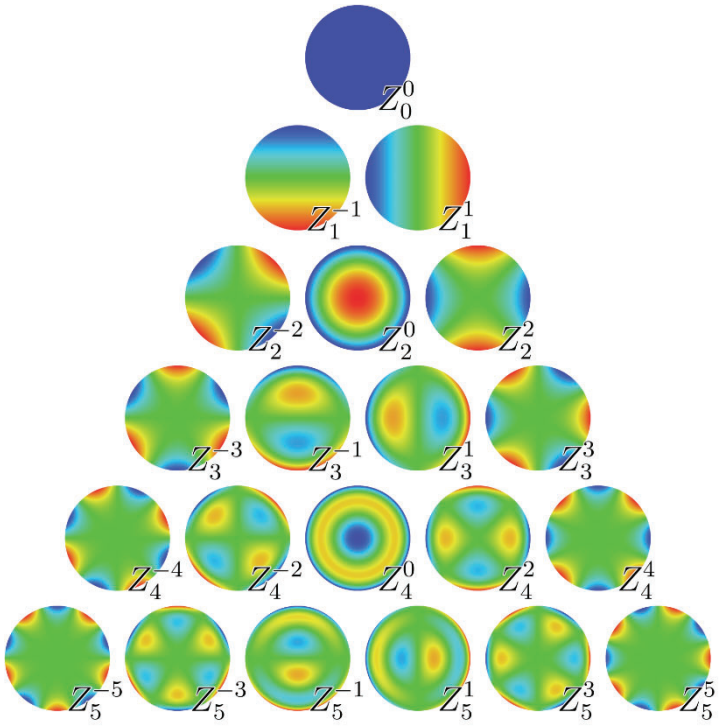

Figure 1. Phase Descriptions of the First 21 Zernike Polynomials

For the purposes of this study, all Zernike polynomials will be ignored when creating the OTF for the aperture. However, future studies could take different aberrations into consideration to see how it effects the estimation of the Fried's seeing parameter. The OTF of the aperture will be discussed in Chapter III.

2.5) Fourier Optics Fundamentals Wrap-up

With these Fourier optics fundamentals, detection algorithms can be created. The PSF and OTF are crucial for creating simulated data to analyze. This is split into two separate OTFs: the OTF of the atmosphere and the OTF of the aperture. The OTF of the atmosphere is dependent on

the seeing parameter, and the OTF of the aperture is dependent on the aberrations. These OTFs can then be converted into a PSF by multiplying the aperture and atmospheric OTFs together, and inverse Fourier transforming them. The PSF can then be used in a specified detection algorithm to test whether an object is considered present within the simulated image.

This study will focus on three detection algorithms: the point detector, the MFC detector, and a newly created algorithm which will be referred to as the PCC detector. The sections following section 2.6 will describe each in detail, and Chapter III will explain how each detection algorithm is used in this study.

2.6) Probability of Detection and False Alarm

Probability of detection and probability of false alarm are defined first and foremost by the likelihood ratio test (LRT). The LRT is a statistical test of how well data fits between two models based on the ratio of their probabilities. The two models used for this LRT are H_1 and H_0, and defined as the case when an object is present in an image (H_1), and the case when an object is absent from an image (H_0). This test can be calculated with Equation 10 below [9].

$$\Lambda = \frac{\ln(P(D \mid H_1))}{\ln(P(D \mid H_0))} < 1 \text{ say } H_1. \text{ Otherwise say } H_0. \tag{10}$$

From Equation 10, $(P(D \mid H_1)$ is the probability mass function (PMF) for the case when an object is present in the image, and $(P(D \mid H_0)$ is the PMF for the case when an object is absent from the image. In this case, it is assumed that the LRTs are Gaussian random variables. Because the mean is equal to the variance, the PMF has the form shown below in Equation 11 [9].

$$P(D \mid H_1) = \frac{1}{\sqrt{2\pi B}} e^{\frac{-(D-B)^2}{2B}} \tag{11}$$

Where B is a non-zero background value and D is the measured signal of S and B combined. Although S is not denoted within this equation, it is the mean value of the target object it can be calculated simply by subtracting the background value (B) from the measured value (D) [9].

Probability of detection is the chance that a detector will find the object when it is present within the image (H_1 is true) and can be described for discrete PMFs by Equation 12 below [9].

$$\mathrm{P_d} = \sum_{D \in D_{\mathrm{object}}} P(D \mid H_1) \tag{12}$$

In this equation, O_{bject} is the set of values for which the LRT is less than 1. For given values S and B, the set of values that caused the LRT to be less than 1 could be determined. In this way, o_{bject} is determined as a function of these signal and background parameters. With o_{bjects} identified, Equation 12 can be used to compute the probability of detection [9].

Probability of false alarm is the chance that if the object is not present, an object will be falsely detected within the image (H_0 is true). It can be computed using Equation 13 below [9].

$$P_{fa} = \sum_{D \in D_{NB}}^{\infty} P(D \mid H_0) \tag{13}$$

The threshold set, D_{NB}, is chosen so that the right side of the equation is equal to the specified probability of false alarm, P_{fa}. The strategy here involves summing over the probability of the measured data, given that no object is present. This PMF is a function of only the average value of the background, B. Although S requires an object to be present, the background can be measured without an object present. With the ability to measure the background, this makes it possible to find the PMF $P(D \mid H_0)$ so that the set D_{NB}, that provides the proper probability of false alarm, can be calculated [9].

In the following sections the LRT for each detection algorithm will be explained and compared against one another.

2.7) The Point Detector

The point detector is a binary hypothesis test (BHT) in which the detection decision is made on a single pixel in a given frame of data. The SNR level is used to make a classification decision (which hypothesis is to be chosen from the observation) using the equations below.

$$H_1 \text{ case: } SNR(cx,cy) = \frac{d(cx,cy) - B}{\sigma_d} > \gamma \tag{14}$$

$$H_0 \text{ case: } SNR(cx,cy) = \frac{d(cx,cy) - B}{\sigma_d} < \gamma \tag{15}$$

Where (cx,cy) represents the pixel location on the CCD array of the image data, d. B is the background, which is the median value of the image in question, and σ_d is the standard deviation of the image. Off to the left of the Equation, the H_1 and H_0 signify two different potential outcomes. For the H_1 case, the SNR value must exceed γ, (the threshold value selected for the detection algorithm) which would result in the point detector marking the pixel in question as having an object present in the image. If, however, the SNR is less than γ, (H_0 case) then the algorithm would consider the image to have no object in the image.

Because this is all based on the outcome from a single pixel, this could yield unfavorable or inaccurate results if a noise spike is present at the test location. It is common for organizations to set a threshold value (γ) that is very high to greatly reduce the chance of a false alarm occurring. This safe approach also significantly reduces the probability of detection as an adverse effect. Figure 2 displays an example of the receiver operator characteristic (ROC) curve for the point detector. It displays the probability of detection versus probability of false alarm of the detection algorithm. The ROC curve will be explained in more detail in Chapter III.

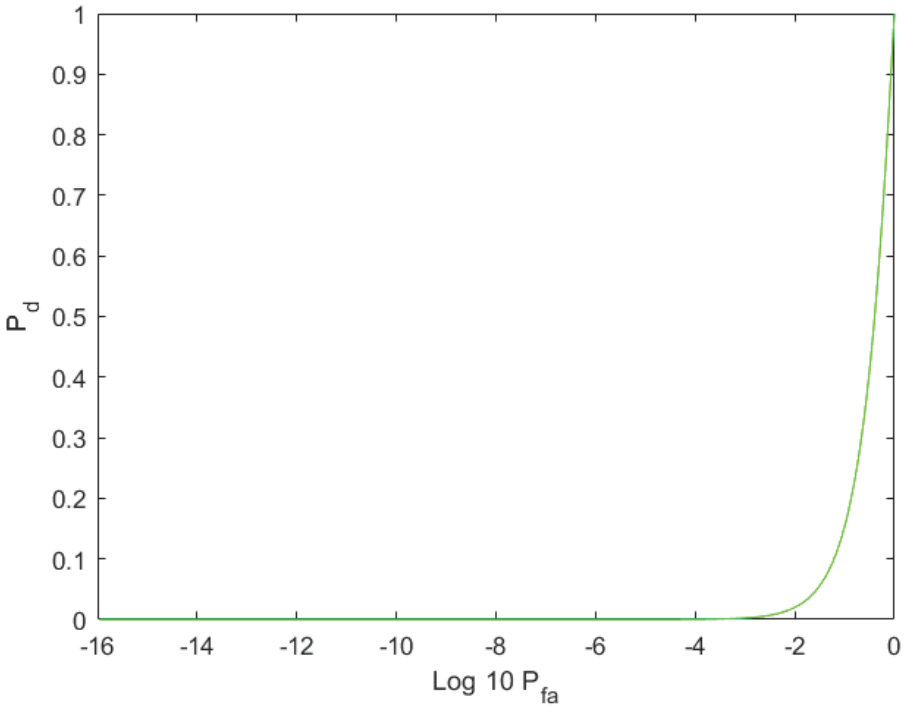

Figure 2. Point Detector ROC Curve Example

In the above figure, the values represent percentages. A value of 0 represents 0%, whereas a value of 1 represents 100%. As the probability of false alarm gets very small ~0.01, there is a probability of detection of ~0.02, which means that on average, there would be 1 false alarm every 100 images, while the algorithm would only detect 2 objects successfully out of every 100 images.

2.8) The Matched Filter Correlator Detector

The MFC detector is another BHT, but instead of using a single point to make the detection decision like the point detector, it instead takes every

pixel within the image into consideration. It does this by double summing the image in both the x and y directions rather than selecting one pixel from the image. Another significant difference between the point detector and the MFC detector is the fact that the MFC detector requires prior knowledge of the PSF. Because of this issue, the point detector is still used for some object detections where no prior data is available on an object in question. shown below represents the MFC detector BHT.

$$H_1 \text{ case: } SNR_{MFC} = \sum_x \sum_y \left(\frac{\left(d(x,y) - B \right)}{\sigma_d} \right) \left(\frac{h(x,y) - \bar{h}}{\sigma_h} \right) > \gamma \tag{16}$$

$$H_0 \text{ case: } SNR_{MFC} = \sum_x \sum_y \left(\frac{\left(d(x,y) - B \right)}{\sigma_d} \right) \left(\frac{h(x,y) - \bar{h}}{\sigma_h} \right) < \gamma \tag{17}$$

As stated earlier, double summing the image in both the x and y directions and subtracting the background value of the image (B), and then divided by the standard deviation of the total captured data (σ_d). This is then multiplied with the difference between the known or hypothesized PSF $h(x,y)$ and its average value, $\bar{h}$, and finally dividing by the square root of the double sum of $h^2(x,y)$ which can be calculated as shown in (18) below.

$$\sigma_h = \sqrt{\sum_x \sum_y h^2(x,y)} \tag{18}$$

If the result is greater than the threshold (γ), an object is considered to be present in the image (H_1 case). If the result is less than γ, then no object is found in the image (H_0 case).

Those familiar with the MFC detector would argue that it has a higher probability of detection, but that is because the MFC detector must have prior knowledge of the actual PSF, whereas the point detector does not require knowledge of the PSF. In Chapter III, the MFC detector will be used in a manner where the PSF is not known and compared against the point detector.

2.9) The Pearson's Correlation Coefficient Detector

The PCC detector measures linear correlation between two variables, η and ε. The value of the PCC detector is always between +1 and -1, where $\rho_{\eta,\varepsilon}$=+1 is perfect positive linear correlation, 0 is no correlation, and -1 is a perfect negative correlation. It is defined as the covariance of the two variables divided by the product of their standard deviations, as shown below [2].

$$\rho_{\eta,\varepsilon} = \frac{\mathrm{cov}(\eta,\varepsilon)}{\sigma_\eta \sigma_\varepsilon} \tag{19}$$

In this equation, σ_η is the standard deviation of random variable η and σ_ε is the standard deviation of random variable ε. The covariance in can be rewritten as shown below.

$$\mathrm{cov}(\eta,\varepsilon) = E\left[(\eta - \mu_\eta)\ (\varepsilon - \mu_\varepsilon)\right] \tag{20}$$

By substituting into the numerator the following equation is produced [8].

$$\rho_{\eta,\varepsilon} = \frac{E\left[(\eta - \mu_\eta)\,(\varepsilon - \mu_\varepsilon)\right]}{\sigma_\eta \sigma_\varepsilon}, \tag{21}$$

where μ_η is the mean of η, and μ_ε is the mean of ε. can then be altered for an image, where $d(x,y)$ is substituted in for η, with a mean of $\overline{d}$ and a standard deviation of σ_d, and similarly, $h(x,y)$ substituted in for ε, with a mean of $\overline{h}$ and a standard deviation of σ_h as shown below:

$$\frac{E[(d(x,y) - \overline{d})(h(x,y) - \overline{h})]}{\sigma_h \sigma_d} \tag{22}$$

The numerator in (22) can be approximated as the double sum of x and y divided by the number of pixels squared as shown in (23).

$$E[(d(x,y) - \overline{d})(h(x,y) - \overline{h})] \approx \sum_{x=1}^{N}\sum_{y=1}^{N}(d(x,y) - \overline{d})(h(x,y) - \overline{h}) / N^2 \tag{23}$$

Substituting (23) back into the numerator of (22) produces the final form of the PCC detector as shown in (24).

$$PCC = \frac{\sum_{x=1}^{N}\sum_{y=1}^{N}(d(x,y) - \overline{d})(h(x,y) - \overline{h}) / N^2}{\sigma_h \sigma_d} \tag{24}$$

The threshold value for the BHT can now be calculated. To begin, (25) below shows the expected value for the PCC detector given the H_0 case.

$$E[PCC \mid H_0] = \sum_{x=1}^{N}\sum_{y=1}^{N}(d(x,y) - \overline{d})(h(x,y) - \overline{h}) / N^2 = 0 \tag{25}$$

As (25) shows, the expected value when no object is present is zero. With a mean (μ) of zero, the calculation of the variance is simplified to (26) below.

$$\sigma_C^2 = E[C^2 \mid H_0] = \frac{E[(\sum_{x=1}^{N}\sum_{y=1}^{N}(d(x,y)-\overline{d})(h(x,y)-\overline{h})/N^2)^2]}{(\sigma_h\sigma_d)^2} \tag{26}$$

The square of the double summation in the numerator can be separated into two similar components, with sums over points (x,y) and (z,w). Each component has a $(h(x,y)-\overline{h})/N^2$ term that is not random and can move outside the expectation. This movement and substitution yields (27) below.

$$\frac{\sum_{x=1}^{N}\sum_{y=1}^{N}\sum_{z=1}^{N}\sum_{w=1}^{N}\frac{(h(x,y)-\overline{h})(h(z,w)-\overline{h})}{N^4}\delta(x-z)\delta(y-w)E[(d(x,y)-\overline{d})(d(z,w)-\overline{d})]}{(\sigma_h\sigma_d)^2} \tag{27}$$

The dirac functions are present because when $x \neq z$ or $y \neq w$, the equation results in a value of zero, due to the assumption that the data is statistically independent from point to point. However, when x = z and y = w, the value shown below is the result.

$$\sigma_C^2 = E[C^2 \mid H_0] = \frac{\sum_{x=1}^{N}\sum_{y=1}^{N}\frac{(h(x,y)-\overline{h})^2}{N^4}E[(d(x,y)-\overline{d})^2]}{(\sigma_h\sigma_d)^2} \tag{28}$$

Equation (28) can be further simplified by finishing the expected value calculation in the numerator. $E[(d(x,y)-\overline{d})^2]$ is actually the variance of

$d(x,y)$, and because the data is Poisson, it is equivalent to $\overline{d}$, so it can be simplified as shown in (29) below.

$$\sigma_C^2 = E[C^2 \mid H_0] = \frac{\sum_{x=1}^{N}\sum_{y=1}^{N}\frac{(h(x,y)-\overline{h})^2}{N^4}\overline{d}}{(\sigma_h\sigma_d)^2} \tag{29}$$

Continuing on with the simplifications, the double sum in the numerator over N^2 is equivalent to σ_h^2. This leaves an N^2 in the denominator, allowing the σ_h^2 terms to cancel out. It was stated earlier, $\overline{d}$ is equivalent to σ_d^2 due to the data being Poisson, so those terms cancel out as well. The final simplification of σ_C^2 is shown in Equation (30).

$$\sigma_C^2 = E[C^2 \mid H_0] = \frac{\frac{\sigma_h^2}{N^2}\overline{d}}{(\sigma_h\sigma_d)^2} = \frac{\sigma_h^2\overline{d}}{N^2(\sigma_h\sigma_d)^2} = 1/N^2 \tag{30}$$

With the variance calculated, a simple square root calculation can be conducted, leaving the standard deviation to be simply $1/N$. This standard deviation allows the MFC detector and PCC detectors to be evenly compared against one another, which is very important when comparing the probability of detection of both detection algorithms after setting the probability of false alarm to be the same.

2.10) Detection Algorithm Review

Seeing the different methods used by each detection algorithm goes to show there are many ways that space object detection can be accomplished. It is likely that there are better methods out there that have yet to be discovered.

Chapter III will dive deeper into the MATLAB simulations ran using these detection algorithms in order to attempt to estimate r_0 for the MFC PCC detection algorithms when the PSF is unknown.

III. MATLAB Simulation

Chapter Overview

In this chapter, simulation via MATLAB R2019b was used to test, evaluate and compare the three detection algorithms described in Chapter II. One PSF was generated for this simulation, and was used for all three detection algorithms. The methodology for its creation will be described in the following section. The main purpose of this chapter is to showcase the different detection algorithms using a controlled and known PSF, with the only variation from one run to the next being random Poisson noise. After evaluating each detection algorithm individually, this chapter will conclude with a comparison between all three detection algorithms in a controlled simulated environment.

3.1) MATLAB Simulation Methodology

This section will dissect and explain the MATLAB code used to run the simulation that was used to compare each algorithm at the end of this Chapter. Each simulation was run with 1000 trials in order to have a large enough sample size to give confidence in results, but small enough so that the processing time was not significant.

3.1.1) Setup

For the remainder of this section, refer to the MATLAB code in the Appendix. Starting at the top of the code in Section One, the photon count (k_bar) was set to 1000. Based on descriptions of the detection algorithms from Chapter II, this value was selected to be low so that all detection algorithms did not have perfect detection rates, which would allow for a comparison to be made among the algorithms. Because the PCC detector is normalized, initial results were expected to be comparable to the MFC detector and better than the point detector when using a low photon count.

A background value (B) was set to 10. It is important that this value is a nonzero value, but not so high as to compete with the photon count from the object. In the H_0 case for all three detection algorithms, if the background value plus random noise exceeds the set threshold, a false alarm would be recorded. Due to the nature of the space object detection custody problem, recording false alarms can be very costly. Because of this, the probability of false alarm is set to be very low, which reduces the probability of detection as well. Probability of false alarm can be set through the threshold value for each detection algorithm. The threshold values are calculated differently for each detection algorithm, as shown in Chapter II, so in order to compare the algorithms' probability of detection, the threshold values used for each algorithm must be fine-tuned so that the probability of false alarm is the same for each detection algorithm. Higher threshold values equate to a lower

probability of false alarm, because it takes more noise in order to overcome the higher threshold set for the data being analyzed. For the photon count and background value set for this simulation, the detection rates are expected to be high among the MFC detector and PCC detectors, and lower for the point detector.

In order to decrease processing time further, a 100x100 window (m) is used, as opposed to a larger window, such as 1000x1000. Larger window sizes are important when the telescope has a larger aperture, as it would result in a loss of resolution. This window size is sufficient for the size of the simulated telescope used, with an aperture diameter of 10 cm.

In order to introduce a simulated image with atmospheric turbulence, the Make_otf.m and Make_long_otf.m MATLAB scripts were implemented using

(& Equation 9 from Chapter II and as shown in the Appendix. The atmospheric OTF is created by the Make_long_otf.m script. There are four variables called out in the Make_long_otf.m function: r1 is the radius of the receiver aperture in units of centimeters and was set to 5 cm so that the maximum seeing parameter, 5 cm, being estimated could be achievable with the data tested, dx is the width of the pixel in the pupil plan, and is chosen so that the number of pixels in your source array is greater than 4 times the number of pixels in your aperture radius which was set to 0.002 meters, si is the number of pixels across the array and was set to the m value of 100, and r_0 is the seeing parameter of the atmosphere and varied

from 1 cm to 5 cm in increments of 1 mm, which are typical lower quality seeing parameter values from Earth's surface, typically at low elevation or near cities.

The aperture function was created by using the Make_otf.m MATLAB script, which utilizes five input arguments. The first is r1 which is the radius of the aperture. For the purposes of this study, it was set to 25 pixels, and was not changed for the entirety of testing. The second is r2, which is meant to describe the radius for obstructions in the pupil plane like a secondary mirror. Obstructions can be described through Zernike polynomials as described in Chapter II, but for the purposes of this study, the obstruction value was set to 0, and was not changed throughout testing. The si term is the same as the one used for the Make_long_otf.m function, which is the number of pixels across the array (m=100). The function is not scaled, so that value is set to 1. For the purposes of this study, the phase term was set to all zeros across the 100x100 array. Future work could implement a phase component into the optical OTF to further test the efficacy of the MFC detector and PCC detector detection algorithms. These parameters were chosen to mimic the specifications of a small telescope used to collect measured data to test the algorithms under study in this thesis.

3.1.2) PSF Figures

This Section displays the background image and the actual PSF image that both the MFC detector and PCC detector attempt to estimate. For this

simulation, the PSF was tested at five different r_0 values, 1 cm to 5 cm, as stated in Section 3.1.1. Figures 5 through 7 show the background image, the PSF with an r_0 value of 1 cm, which is the worst quality image in this test, and the PSF with an r_0 value of 5 cm, which is the best quality image for this test. The PSFs shown in the figures are pictured with no noise acting upon them.

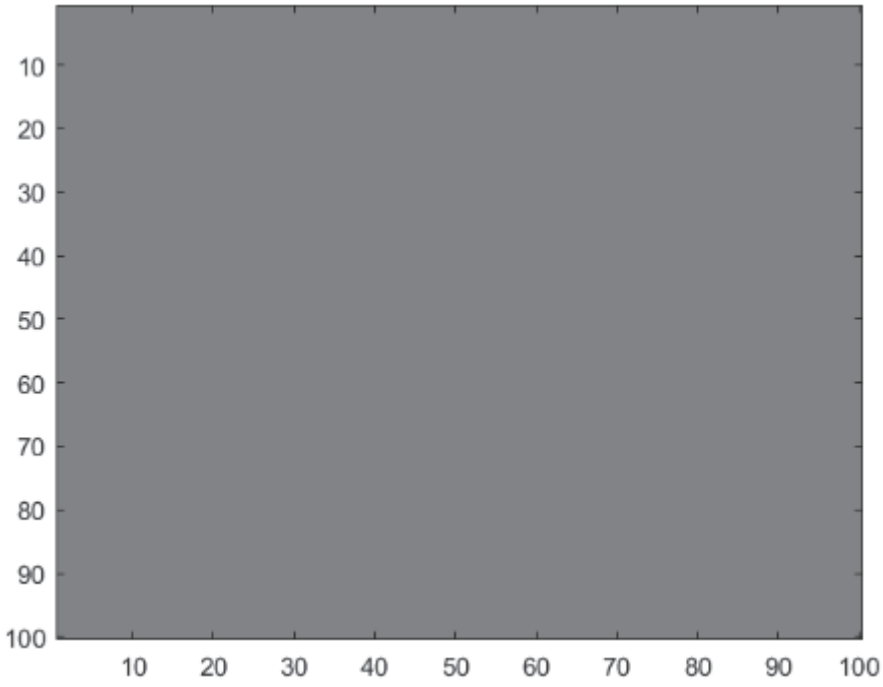

Figure 3. Background Image (No PSF Present)

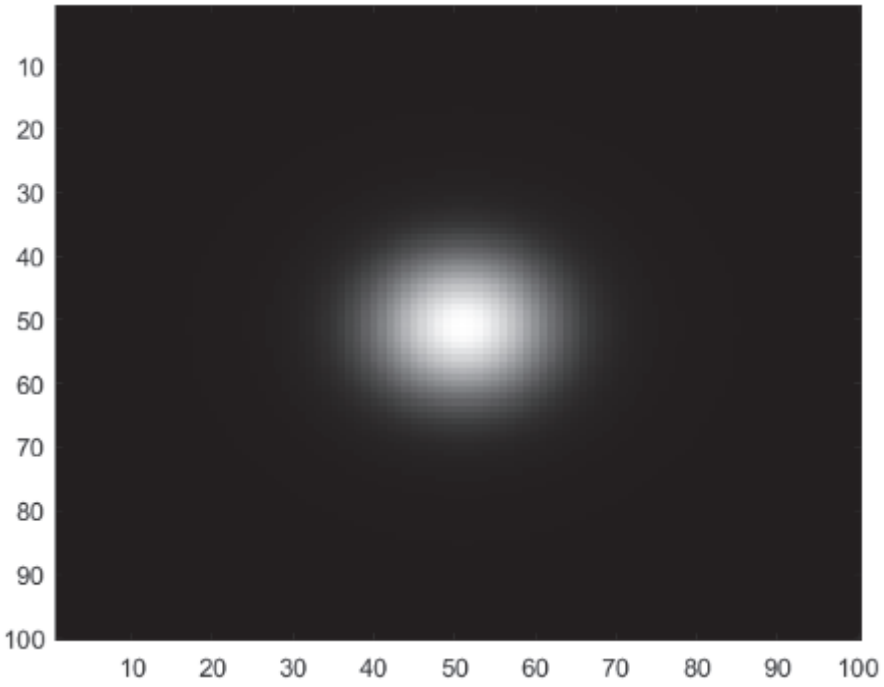

Figure 4. PSF with r_0 Value of 1 cm

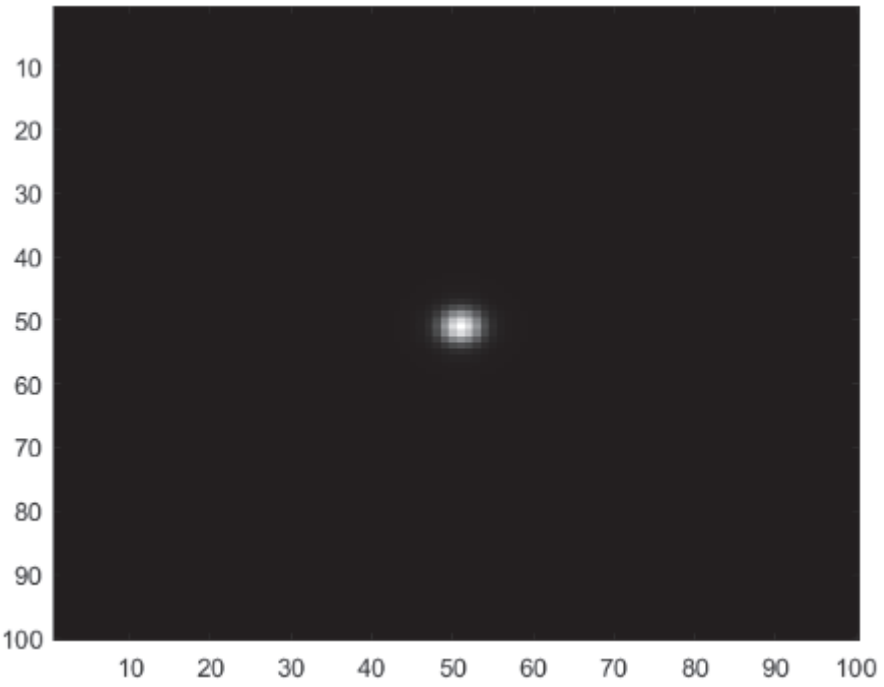

Figure 5. PSF with r_0 Value of 5 cm

When comparing these images, the PSF with a r_0 value of 1 cm appears much larger than the PSF with a r0 value of 5 cm. Recall back to (in Chapter II where the Fried's Seeing Parameter (r_0) was explained. As the atmospheric turbulence increases, the r_0 value decreases. This decreased r_0 value correlates to larger PSF sizes, which can be described as increased distortion of the light source being imaged. The sharper the image (higher r_0 values), the more concentrated the intensity of the PSF becomes, which should lead to an increase in the probability of detection of the light source being imaged. This theory will be confirmed later in this chapter during a review of the results.

3.1.3) H_0 Case

The seeing parameter estimation is described in this section and is accomplished through a series of functions spreading over the next few sections. This begins with a for loop that creates a matrix of PSFs with varying r_0 values ranging from 0.1 cm to 5 cm in iterations of 0.1 cm. This matrix is used later on in the section after noise is added to the image being tested. Because this section is testing the H_0 case, there is no PSF present in the image, and a matrix of zeros is used.

Once the noise has been added to the matrix of zeros, and the median and standard deviation have been calculated for the image, the point detector algorithm can be run. Figure 6 below is an example image for the H_0 case with randomly generated noise.

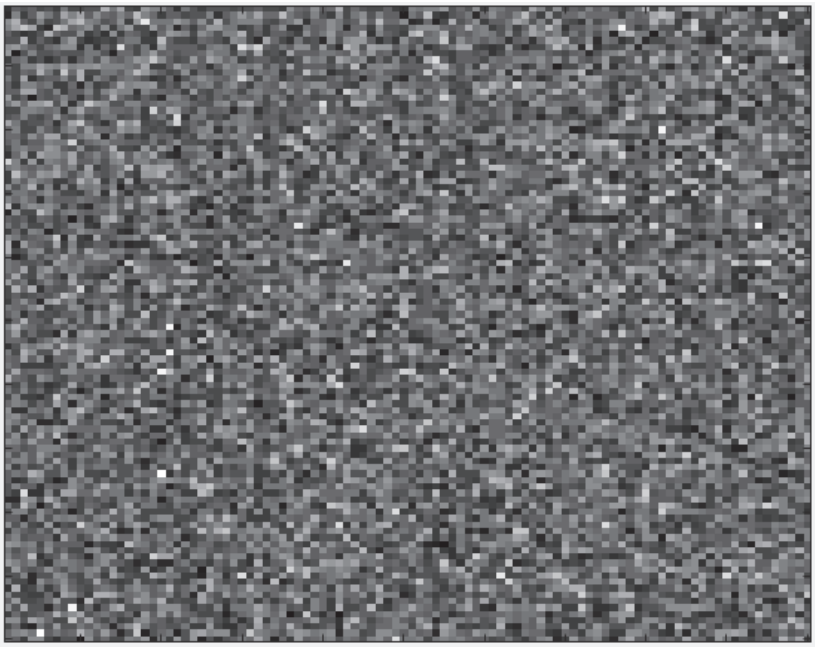

Figure 6. Example Randomly Generated H_0 Image

Moving on to the MFC detector and PCC detector detection algorithms, these are calculated using (16) & (24) respectively and as shown in the MATLAB code. These algorithms are run using the PSF matrix created earlier in Section Three, which creates values for both MFC detector and PCC detector to be compared against their respective threshold values. Once all datasets have been calculated for each r_0 value and for both the MFC detector and PCC detector, the maximum value for each of the 1000 trials for both algorithms are then stored for later usage.

3.1.4) H_1 Case

The H_1 case contains much of the same code as Sub-Section Three, however, there are a few very important differences. Instead of adding noise to a matrix of zeros, the PSF that was created in Sub-Section 1 is utilized. An example image for the H_1 case can be seen below in Figure 7.

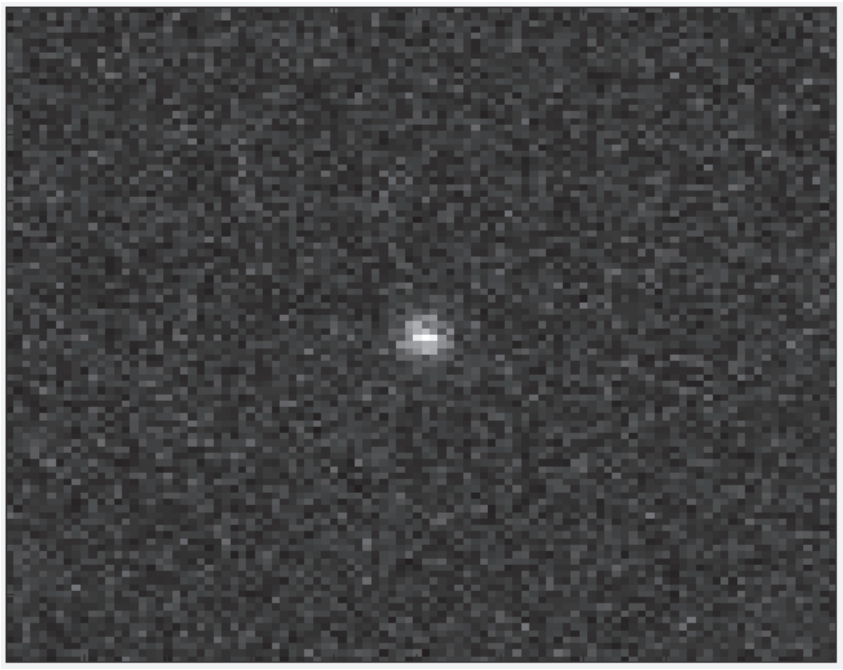

Figure 7. Example Randomly Generated H_1 Image

With an object present in the image, the r_0 estimation can be conducted. But first, instances where no object was detected in the image must be removed prior to locating the maximum values from each trial among the dataset. The mean of all 1000 trials multiplied by the step size change per trial (1 mm) will then yield the estimated r_0 value based on the data collected.

3.1.5) ROC Curves

Receiver operating characteristic (ROC) curves are very useful tools that plot the probability of detection versus the probability of false alarm. As stated in Dr. Stephen Cain's Direct-Detection LADAR Systems, "These plots are used to compare the performance of one detector or detection scheme versus another. In general, a detection method with an ROC value that is higher than another method's ROC value indicates that for the same probability of false alarm, the first method has a higher probability of detection than the second" [9]. Both probabilities are a function of the threshold that defines the presence of an object, and the background image only. "Since the threshold varies, the probability of detection and probability of false alarm change. A graph can be constructed to demonstrate the probability of detection versus the probability of false alarm" [9]. These graphs are generated for the simulated data and shown for the $r_0 = 0.01$ trial below in Figure 8.

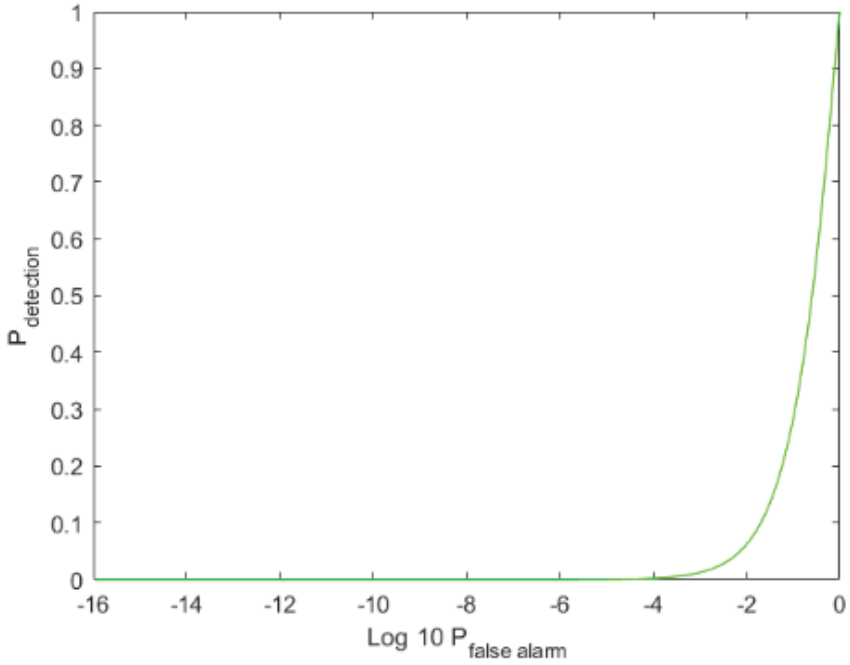

Figure 8. Point Detector ROC Curve

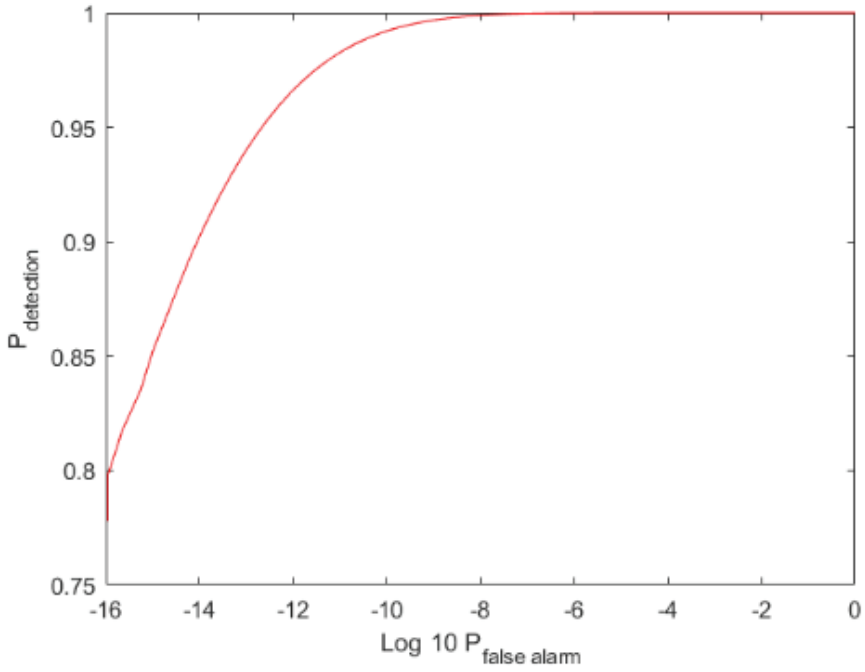

Figure 9. MFC detector ROC Curve

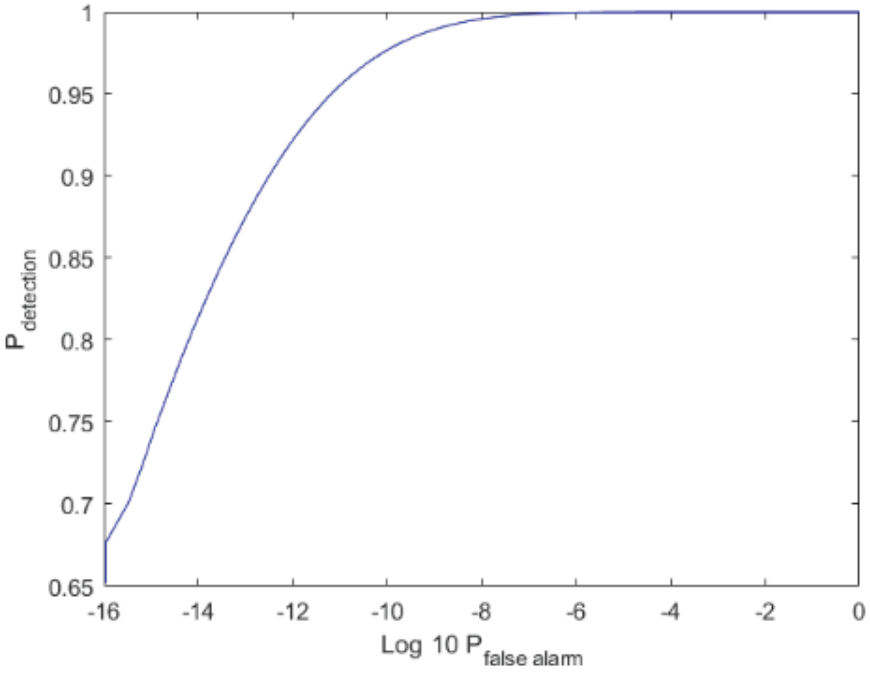

Figure 10. PCC detector ROC Curve

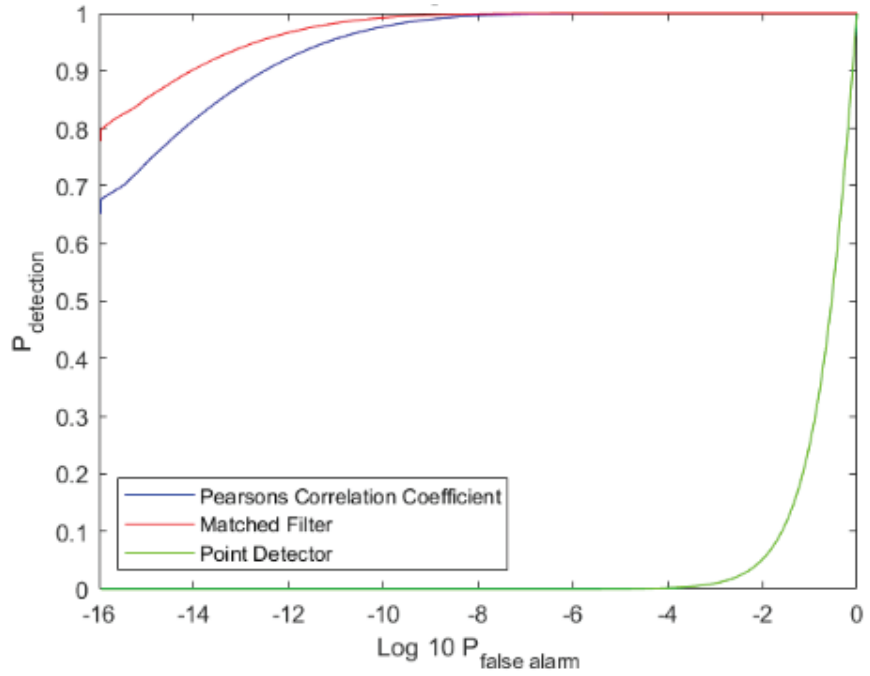

Figure 11. Detectors Comparison ROC Curve

The final figure (Figure 11) shows the comparison of all three detection algorithms. In this Figure, the MFC detector and PCC detector far outperform the point detector. But upon further inspection, the MFC detector has a higher probability of detection than the PCC detector. Although the PCC detector does not outperform the MFC detector in regards to the probability of detection for this specific test, there may be other uses for the PCC detector where it could outperform the MFC detector. This theory will be further explored within this study, and it could also be expanded upon in future work.

3.2) Simulation Results

The simulation was run for r_0 values varying from 1 cm to 5 cm. Five tests were conducted and recorded for the r_0 estimation. The results of these tests are shown below in Table 1 and averaged in Table 6.

Table 1. Estimated r_0 Values for Simulated Data Test 1

Detector	Estimated r_0 Values (cm)				
Actual	1	2	3	4	5
MFC	1.00	2.20	3.20	3.70	5.00
PCC	1.00	2.30	3.10	3.70	5.00

Table 2. Estimated r_0 Values for Simulated Data Test 2

Detector	Estimated r_0 Values (cm)				
Actual	1	2	3	4	5
MFC	0.90	2.00	2.90	4.10	4.90
PCC	0.80	2.00	2.90	4.10	4.80

Table 3. Estimated r_0 Values for Simulated Data Test 3

Detector	Estimated r_0 Values (cm)				
Actual	1	2	3	4	5
MFC	1.10	2.00	3.00	4.10	5.00
PCC	1.00	2.00	3.00	4.10	5.00

Table 4. Estimated r_0 Values for Simulated Data Test 4

Detector	Estimated r_0 Values (cm)				
Actual	1	2	3	4	5
MFC	1.20	2.00	3.20	4.50	4.70
PCC	1.10	2.00	3.30	4.40	4.70

Table 5. Estimated r_0 Values for Simulated Data Test 5

Detector	Estimated r_0 Values (cm)				
Actual	1	2	3	4	5
MFC	0.90	1.90	2.80	4.00	5.00
PCC	0.90	1.90	2.80	4.00	5.00

Table 6. Averaged r_0 Estimation Values

Detector	Estimated r_0 Values (cm)				
Actual	1	2	3	4	5
MFC	1.02	2.02	3.02	4.08	4.92
PCC	0.96	2.04	3.02	4.06	4.90

These results show that estimation of the r_0 value is possible, and quite accurate with 1000 trials per run. These results boast an impressive variance (averaged across all r_0 values) of approximately 0.025 cm and a standard deviation of approximately 0.15 cm further validating the accuracy of this estimator. The MFC and PCC detectors had negligible differences between them as shown above in Table 6, with only a 0.1 cm difference as the largest gap between their estimations, and an average difference of only 0.024 cm. If these estimators can maintain values this precise under a multitude of different images, this could be an incredibly powerful tool for space object detection. In the next chapter, this estimation technique will be put to the test once more using measured data.

IV. Polaris A & B Data Test

Chapter Overview

This chapter focuses on using the code established in Chapter III to evaluate measured data collected of well-known stars, Polaris A and Polaris B, in order to test the validity of the seeing parameter estimation on measured image data. After evaluating each detection algorithm individually, this chapter will conclude with a comparison between the MFC and PCC detection algorithms in this real-world environment.

4.1) Measured Data Description & Methodology

Simulated data can be tailored so that the intended result is achieved, however, it is important to use measured data with a detection algorithm to see how well it performs in a real-world environment. For this chapter, the PCC detector, MFC detector, and point detector will be tested on measured image data collected through a telescope. Polaris A and Polaris B were observed through a CelestronXLT telescope with a 2.8 meter focal length. The telescope had an aperture of 11 inches, but was only opened to 2 inches during data collection. The data was collected by Dr. Stephen Cain in Dayton, Ohio. The Fried's seeing parameter for that night was approximately 2.5 cm, which is shown in Figure 12 below. This was produced

by plotting the Polaris A image across its center against the modeled PSF of the average PSF when r_0 is 2.5 cm.

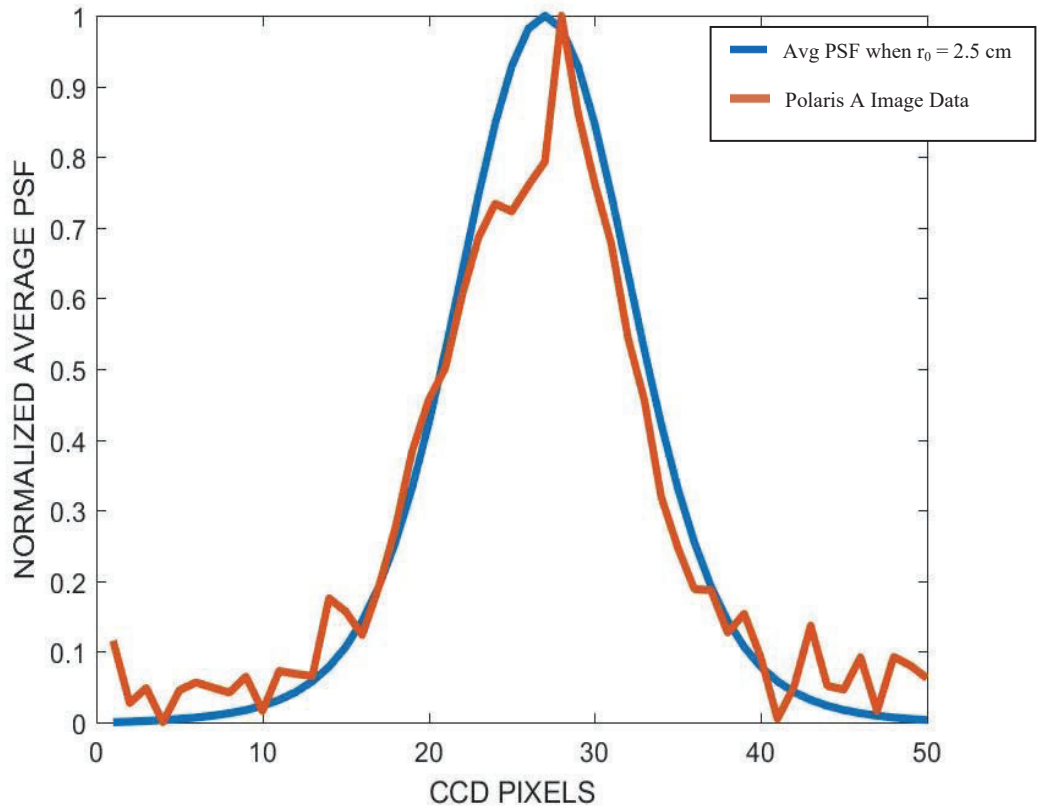

Figure 12. PSF Model at r_0 2.5 cm vs. Polaris A Images

One-thousand images were taken of Polaris A, without Polaris B present in the image as shown by an image of their average in Figure 13. An additional one-thousand images were taken of Polaris A with Polaris B present in the image, which is shown by their average in Figure 14 below.

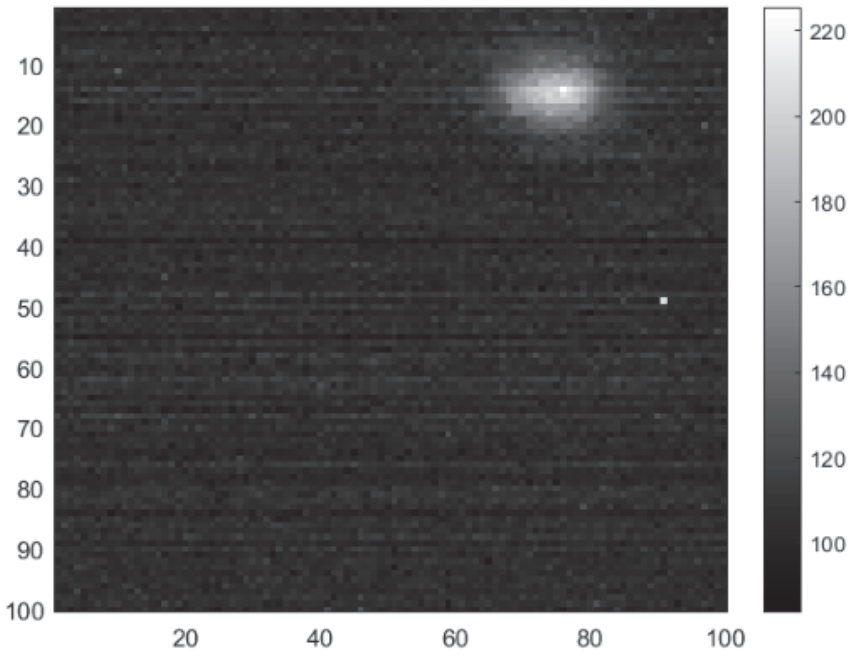

Figure 13. Average Image Data of Polaris A Only

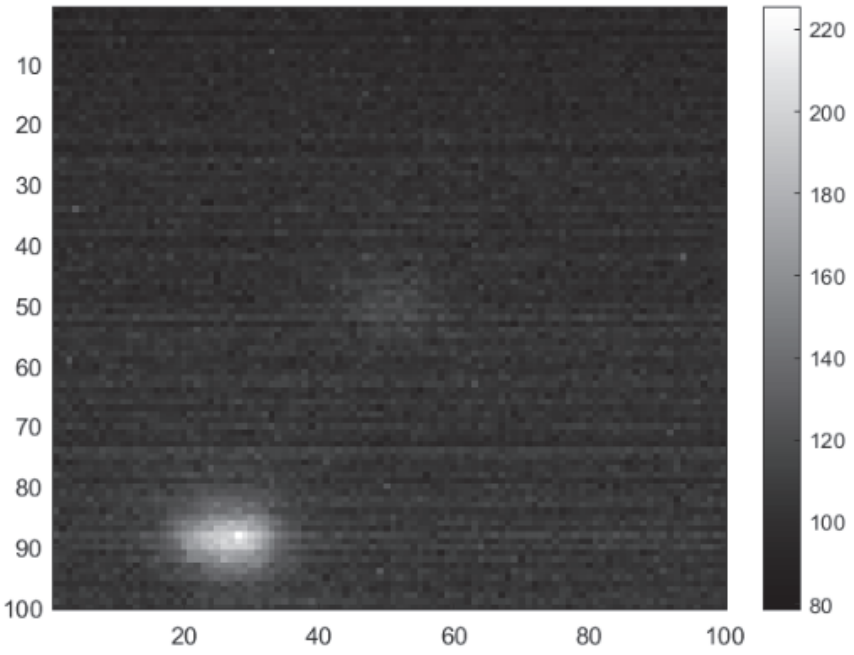

Figure 14. Average Image Data of Polaris A with Polaris B Present

Upon close inspection, Polaris B can be seen in the center of Figure 13. This faint star is the space object in which each detection algorithm attempted to detect within each captured image. The collected data was input into MATLAB R2019b, which was used to test, evaluate and compare the three detection algorithms. The methodology is very similar to Chapter III, with the main difference being within the setup. Because measured data are used instead of simulated data, the lines that created the simulated data were removed, and the data was imported prior to running the code.

The main purpose of this chapter is to check the validity of the seeing parameter estimation using measured image data, but a tertiary goal is to compare PCC detector vs MFC detector for detection of a dimly lit space object with other brighter objects within the image.

4.2) Results

4.2.1) Detections vs. False Alarms

The Polaris data collected was evaluated with a PCC detector threshold set at 3 standard deviations. The MFC detector's threshold was then selected so that the false alarms for the MFC detector was equal to the false alarms for the PCC detector. The detections for each algorithm were then compared as shown in Table 7.

Table 7. Polaris B Detection and False Alarm Results

Detector Algorithm	Detections	False Alarms
PCC	94	3
MFC	30	3

Based on this data set, it is clear that the PCC detector outperformed the MFC detector in detections when false alarms were set to be the same. This could be due to the normalization that occurs within the PCC algorithm. The MFC detector is more dependent on intensity, whereas the PCC detector is more dependent on the shape of the PSF. Because Polaris B is much dimmer than Polaris A, the MFC detector struggles to detect it, and due to the brightness of Polaris A in the images where Polaris B is not present, there are significantly more false alarms than the PCC detector reports.

ROC curves were also created for each detection algorithm based on the data provided. These ROC curves are shown below in Figure 15. Polaris B Point Detector ROC Curve. Figure 18 then shows all three ROC Curves together for a side-by-side comparison.

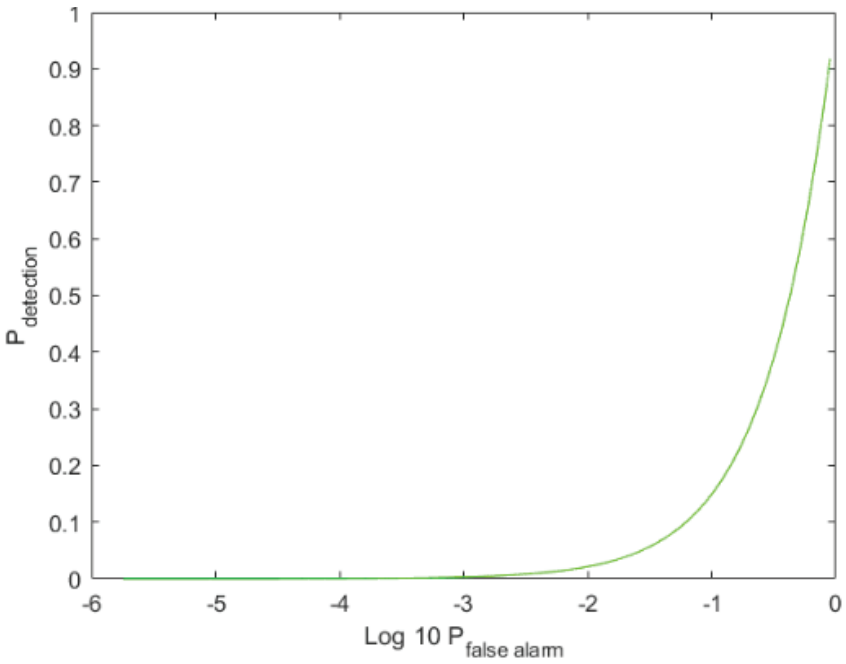

Figure 15. Polaris B Point Detector ROC Curve

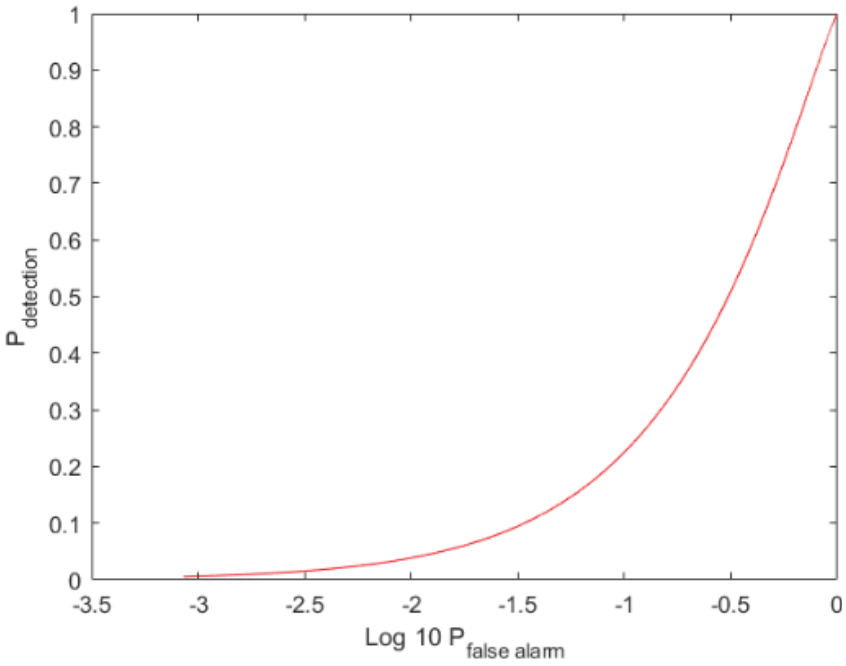

Figure 16. Polaris B MFC ROC Curve

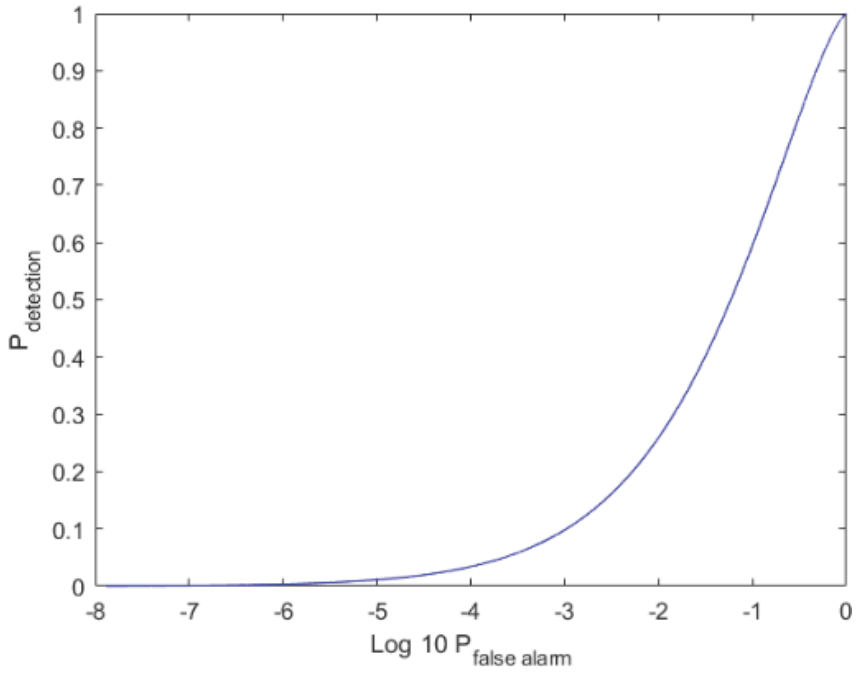

Figure 17. Polaris B PCC ROC Curve

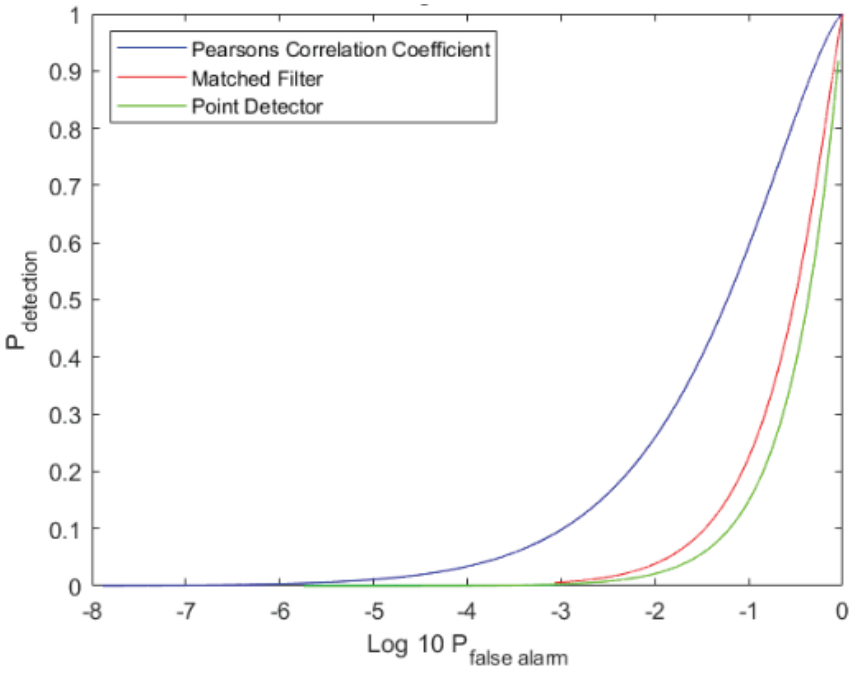

Figure 18. Polaris B Combined ROC Curves Comparison

As Figure 18 shows, the MFC detector is only a small step up in performance from the point detector in this dataset, however, the PCC detector shows its detection prowess in this situation. The results from this test suggests that the PCC detector may be better at space object detection while other brighter objects are near the space object being analyzed. Further testing should be done to explore this practical use for the PCC detector.

4.2.2) Seeing Parameter Estimation

The seeing parameter estimation experienced some issues with this test data, specifically for the MFC detector. Table 8 shows the results for the seeing parameter estimation using the Polaris image data.

Table 8. Polaris Seeing Parameter Estimation Results

Detector Algorithm	Seeing Parameter Estimation
PCC	2.28 cm
MFC	0.15 cm

In this case, the PCC detector detected an object in 94 out of 1,000 images, and the MFC detector only detected an object in 30 out of 1,000 images, which is 9.4% and 3% respectively. The difference in detections did not seem large enough to give the PCC detector a significant advantage in calculating the seeing parameter. Further testing with more measured data

of other space objects is recommended to see if the MFC detector struggles to estimate the seeing parameter of measured image data.

The PCC detector's seeing parameter estimation of 2.28 cm appears to be accurate based on the r_0 value of 2.5 cm modeled in Figure 12. The PCC detector seems to be more robust than the MFC detector in dealing with non-ideal PSF shapes. Further testing of different space objects is recommended for future tests to confirm the accuracy of the PCC detector seeing parameter estimation.

V. Conclusions and Recommendations

As the space domain becomes increasingly contested, it motivates researchers to create more effective methods for space object detection. The point detector has been used as a niche detection algorithm for decades for space objects with no prior knowledge of the object in question. The PCC detector has also shown promise for faint object detection in images with brighter objects present that should be explored further.

5.1) Conclusions of Research

In this thesis, both the PCC detector and MFC detector outperformed the point detector in both the simulated and measured data tests.

When comparing the PCC detector versus the MFC detector using the simulated data, the MFC came out on top, boasting the highest detection rates, and an equally accurate r_0 estimation. But, when comparing both algorithms again using measured data, the PCC detector significantly outperformed the MFC detector in both detection, and r_0 estimation. This discovery could show a practical use for the PCC for dim space object detection while other brighter objects or stray light is present within the image. It is worth noting there were quite a few differences between the two data sets. The measured data had more noise in the image, and there is also the possibility of aberrations in the imaging system. The simulated data did have random noise throughout the data, but not to the same extent, and was

operating under the assumption that the aperture had no aberrations. Another glaring difference was the fact that the simulated data was focused on a single centered PSF with no other objects in the image, whereas the measured data had two objects in the image, and the star being observed, Polaris B, was much dimmer than the star next to it, Polaris A. It is believed that due to these specific circumstances, the PCC detector was poised to outperform the MFC detector. By reviewing more types of measured data in future work, these detectors can be compared further, and possibly combined in order to make a detection algorithm that works effectively under most conditions.

5.2) Significance of Research

The findings of this research bring forth some useful improvements upon current space object detection methods. The long, and widely used point detector is now one step closer to retirement after decades of use across many organizations. There are still some applications where the point detector can be useful. This is mostly in cases where there is insufficient processing power available to implement the more advanced detectors. With the use of the r_0 estimator, the MFC and PCC detectors can be used to accurately estimate the seeing parameter, which allows both detectors to be used on newly discovered or undocumented space objects. This makes these

detection algorithms more practical, since historically, the main objection to the MFC detector was that the PSF must be known.

Based on the results of the measured data evaluated, it appears the PCC detector could be significantly more effective at faint object detection than the MFC detector. If this proves to be true for more cases, the PCC detector could serve a crucial role in some faint object detection cases.

5.3) Recommendations for Future Research

Future work could be done to combine the MFC and PCC algorithms, and to find a way to decipher which is more applicable to use depending on the circumstances of the image being analyzed.

In the measured data test, the PCC detector was found to significantly outperform the MFC detector. In this test, Polaris B was the target object, which was being overshadowed by Polaris A. It is assumed that due to the brightness of Polaris A, this negatively impacted the MFC detector, which works purely off of the brightness level as the detection method. The PCC detector has a normalization factor to it, which is likely the reason to its success over the MFC detector in this case. This should be tested further in future work with other measured and simulated data to confirm this assumption.

Appendix

A.1) Simulated Data MATLAB Code

Section 1 - Variable Setup & PSF defined

```
close all;
clear all;


T = 1000; %number of trials
T2 = 50; %number of r0 steps increasing by 1 mm per step
k_bar = 1000; %value of photon, can be adjusted to fit the needs of the detector (25K
works well for PD vs MFC, 500-2000 for PPCC vs MFC)
B_true = 10;%background value
m = 100; %number of pixels in one dimension of the window
cx = m/2; %pixel of x location on PSF
cy = m/2; %pixel of y location on PSF


otf_opt=Make_otf(25,0,m,1,zeros(m,m));
D=.1;


for step=1:5

r0=.01*step;
otf_atm=Make_long_otf(D/2,2*D/m,m,r0);


psf=fftshift(ifft2(otf_opt.*fftshift(otf_atm)));


psf_real=psf;


h_bar = mean(psf_real(:));
sig_h = std(psf_real(:));
sq_sum_hsq=sqrt(sum(sum(psf_real(:).^2)));
threshold = 6/m; %6 Standard Deviations


h_0=zeros(m,m); %MxM array of background
```

Section 2 - Images

```
figure(1)
```

```
imagesc(h_0) %image of background
title('Background Image');
colormap ('gray');


figure(2)
imagesc(psf) %image of PSF
title('PSF');
colormap ('gray');
```

Section 3 - Matched Filter/Correlator for H0

```
 for its=1:T2
% %
%
r0=.001*its;
otf_atm = Make_long_otf(D/2,2*D/m,m,r0);
% %
psf_mat(:,:,its)=real(fftshift(ifft2(otf_opt.*fftshift(otf_atm))));

 end
for i = 1:T

    data_0 = poissrnd(k_bar.*h_0 + B_true);
    window_0 = data_0(cy-m/2+1:cy+m/2,cx-m/2+1:cx+m/2);
    B_0 = median(window_0(:));
    d_bar_0 = mean(window_0(:));
    sig_0 = std(window_0(:));
    point_detector_0(i)=(window_0(m/2+1,m/2+1)-B_0)/sig_0;
    for its=1:T2
psf_100=psf_mat(:,:,its);
h_bar = mean(psf_100(:));
sig_h = std(psf_100(:));
sq_sum_hsq=sqrt(sum(sum(psf_100(:).^2)));
    ExV_0 = sum(sum((window_0-d_bar_0).*(psf_100-h_bar)/m^2));


    image_PC_0(its,i) = ExV_0/(sig_0*sig_h);
    Keep_image_PC_0(its,i) = ExV_0/(sig_0*sig_h)>threshold;

    correlator_0 = sum(sum(((window_0-B_0)./sig_0).*(psf_100./(sq_sum_hsq))));


    store_correlator_0(its,i) = correlator_0;

    sig_Pfa_Mfc = std(store_correlator_0(:));
    threshold_mfc = 6.06*sig_Pfa_Mfc;

    detect_corr_0(its,i) = ((correlator_0) > threshold_mfc);
```

```
   end
   temp1=store_correlator_0(:,i);
   MFC_output_0(i)=max(temp1);

   temp2=image_PC_0(:,i);
   PCC_output_0(i)=max(temp2);
   end

h_0_done=1
```

Section 4 - Matched Filter/Correlator for H1

```
 for i = 1:T

   data_1 = poissrnd(k_bar.*psf + B_true);
   window_1 = data_1(cy-m/2+1:cy+m/2,cx-m/2+1:cx+m/2);
   B_1 = median(window_1(:));
   d_bar_1 = mean(window_1(:));
   sig_1 = std(window_1(:));
   point_detector_1(i)=(window_1(m/2+1,m/2+1)-B_1)/sig_1;
   for its=1:T2
psf_100=psf_mat(:,:,its);
h_bar = mean(psf_100(:));
sig_h = std(psf_100(:));
sq_sum_hsq=sqrt(sum(sum(psf_100(:).^2)));
   ExV_1 = sum(sum((window_1-d_bar_1).*(psf_100-h_bar)/m^2));


   image_PC_1(its,i) = ExV_1/(sig_1*sig_h);
   Keep_image_PC_1(its,i) = ExV_1/(sig_1*sig_h)>threshold;

   correlator_1 = sum(sum(((window_1-B_1)./sig_1).*(psf_100./sq_sum_hsq)));


   store_correlator_1(its,i) = correlator_1;


   detect_corr_1(its,i) = ((correlator_1) > threshold_mfc);

   combined_PCC(its,i) = (Keep_image_PC_1(its,i)+0.00001).*image_PC_1(its,i);
   combined_MFC(its,i) = (detect_corr_1(its,i)+0.00001).*store_correlator_1(its,i);
   end

   temp1=store_correlator_1(:,i);
   MFC_output_1(i)=max(temp1);
   idx_1(i)=find(temp1==max(temp1));

   temp2=image_PC_1(:,i);
   PCC_output_1(i)=max(temp2);
```

```
    idx_2(i)=find(temp2==max(temp2));


    temp2x=combined_PCC(:,i);
    PCC_output_1x(i)=max(temp2x);
    idx_2x(i)=find(temp2x==max(temp2x));


    temp1x=combined_MFC(:,i);
    MFC_output_1x(i)=max(temp1x);
    idx_1x(i)=find(temp1x==max(temp1x));
 end


h_1_done=1


MFC_avg_ro_est(step)=mean(idx_1)*.001 %estimation of r0 value for MFC
PCC_avg_ro_est(step)=mean(idx_2)*.001 %estimation of r0 value for PCC


MFC_avg_ro_estx(step)=mean(idx_1x)*.001 %estimation of r0 value for MFC
PCC_avg_ro_estx(step)=mean(idx_2x)*.001 %estimation of r0 value for PCC
```

Section 5 - ROC Curves

```
PCCm1=mean(PCC_output_1);


PCCm0=mean(PCC_output_0);


PCCs1=std(PCC_output_1);


PCCs0=std(PCC_output_0);


PCC_thresh_max=max(PCC_output_1*50);


PCC_thresh_min=min(PCC_output_0*50);


PCC_step=(PCC_thresh_max-PCC_thresh_min)/10000;


PCC_thresh=PCC_thresh_min:PCC_step:PCC_thresh_max;
```

```
PCC_Pd_comb=1-cdf('norm',PCC_thresh,PCCm1,PCCs1);

PCC_Pfa_comb=1-cdf('norm',PCC_thresh,PCCm0,PCCs0);

MFCm1=mean(MFC_output_1);

MFCm0=mean(MFC_output_0);

MFCs1=std(MFC_output_1);

MFCs0=std(MFC_output_0);

MFC_thresh_max=max(MFC_output_1*50);

MFC_thresh_min=min(MFC_output_0*50);

MFC_step=(MFC_thresh_max-MFC_thresh_min)/10000;

MFC_thresh=MFC_thresh_min:MFC_step:MFC_thresh_max;

MFC_Pd_comb=1-cdf('norm',MFC_thresh,MFCm1,MFCs1);

MFC_Pfa_comb=1-cdf('norm',MFC_thresh,MFCm0,MFCs0);

PD_m1=mean(point_detector_1);

PD_m0=mean(point_detector_0);

PD_s1=std(point_detector_1);

PD_s0=std(point_detector_0);

PD_thresh_max=max(point_detector_1*50);
```

```
PD_thresh_min=min(point_detector_0*50);


PD_step=(PD_thresh_max-PD_thresh_min)/10000;


PD_thresh=PD_thresh_min:PD_step:PD_thresh_max;


Pd_short=1-cdf('norm',PD_thresh,PD_m1,PD_s1);


Pfa_short=1-cdf('norm',PD_thresh,PD_m0,PD_s0);


plot(log10(Pfa_short),Pd_short,'g')
title('Point Detector ROC Curve', 'fontsize', 20);
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')


plot(log10(MFC_Pfa_comb),MFC_Pd_comb,'r')
title('MFC ROC Curve', 'fontsize', 20);
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')


plot(log10(PCC_Pfa_comb),PCC_Pd_comb,'b')
title('PCC ROC Curve', 'fontsize', 20);
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')


plot(log10(PCC_Pfa_comb),PCC_Pd_comb,'b',log10(MFC_Pfa_comb),MFC_Pd_comb,'r',lo
g10(Pfa_short),Pd_short,'g')
title('Detectors Comparison ROC Curve', 'fontsize', 20);
legend('Pearsons Correlation Coefficient','Matched Filter','Point Detector',
'location','SouthWest')
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')
end
```

A.2) Measured Data MATLAB Code

Section 1 - Variable Setup & PSF defined

```
T = 1000; %number of trials
T2 = 50; %number of r0 steps increasing by 1 mm per step


m = 100; %number of pixels in one dimension of the window
cx = m/2; %pixel of x location on PSF
cy = m/2; %pixel of y location on PSF


data_0_avg = 0;
for i = 1:1000
   data_0_avg = data_0_100(:,:,i)+data_0_avg;
end
data_0_avg = data_0_avg./1000;


imagesc(data_0_avg)
colormap('gray')
colorbar


data_avg = 0;
for i = 1:1000
   data_avg = data_100(:,:,i)+data_avg;
end
data_avg = data_avg./1000;


imagesc(data_avg)
colormap('gray')
colorbar


otf_opt=Make_otf(25,0,m,1,zeros(m,m));


D=.1;


h_bar = mean(psf_100(:));
sig_h = std(psf_100(:));
sq_sum_hsq=sqrt(sum(sum(psf_100(:).^2)));
threshold = 3/m; %3 Standard Deviations


h_0=zeros(m,m); %MxM array of background
```

Section 2 - Images

```
imagesc(h_0) %image of background
title('Background Image');
colormap ('gray');


imagesc(psf_100) %image of PSF
title('PSF');
colormap ('gray');
```

Section 3 - Matched Filter/Correlator for H0

```
 for its=1:T2
% %
%
r0=.001*its;
otf_atm = Make_long_otf(D/2,2*D/m,m,r0);
% %
psf_mat(:,:,its)=real(fftshift(ifft2(otf_opt.*fftshift(otf_atm))));

 end
for i = 1:T

    data_0 = data_0_100(:,:,i);
    window_0 = data_0(cy-m/2+1:cy+m/2,cx-m/2+1:cx+m/2);
    B_0 = median(window_0(:));
    d_bar_0 = mean(window_0(:));
    sig_0 = std(window_0(:));
    point_detector_0(i)=(window_0(m/2+1,m/2+1)-B_0)/sig_0;
    for its=1:T2
psf=psf_mat(:,:,its);
h_bar = mean(psf(:));
sig_h = std(psf(:));
sq_sum_hsq=sqrt(sum(sum(psf(:).^2)));
    ExV_0 = sum(sum((window_0-d_bar_0).*(psf-h_bar)/m^2));


    image_PC_0(its,i) = ExV_0/(sig_0*sig_h);
    Keep_image_PC_0(its,i) = ExV_0/(sig_0*sig_h)>threshold;

    correlator_0 = sum(sum(((window_0-B_0)./sig_0).*(psf./(sq_sum_hsq))));


    store_correlator_0(its,i) = correlator_0;

    sig_Pfa_Mfc = std(store_correlator_0(:));
    threshold_mfc = 4.2357*sig_Pfa_Mfc;
```

```
   detect_corr_0(its,i) = ((correlator_0) > threshold_mfc);
   end
   temp1=store_correlator_0(:,i);
   MFC_output_0(i)=max(temp1);

   temp2=image_PC_0(:,i);
   PCC_output_0(i)=max(temp2);
end
```

Section 4 - Matched Filter/Correlator for H1

```
 for i = 1:T

   data_1 = data_100(:,:,i);
   window_1 = data_1(cy-m/2+1:cy+m/2,cx-m/2+1:cx+m/2);
   B_1 = median(window_1(:));
   d_bar_1 = mean(window_1(:));
   sig_1 = std(window_1(:));
   point_detector_1(i)=(window_1(m/2+1,m/2+1)-B_1)/sig_1;
   for its=1:T2
psf=psf_mat(:,:,its);
h_bar = mean(psf(:));
sig_h = std(psf(:));
sq_sum_hsq=sqrt(sum(sum(psf(:).^2)));
   ExV_1 = sum(sum((window_1-d_bar_1).*(psf-h_bar)/m^2));


   image_PC_1(its,i) = ExV_1/(sig_1*sig_h);
   Keep_image_PC_1(its,i) = ExV_1/(sig_1*sig_h)>threshold;

   correlator_1 = sum(sum(((window_1-B_1)./sig_1).*(psf./sq_sum_hsq)));


   store_correlator_1(its,i) = correlator_1;


   detect_corr_1(its,i) = ((correlator_1) > threshold_mfc);

   combined_PCC(its,i) = (Keep_image_PC_1(its,i)+0.00001).*image_PC_1(its,i);
   combined_MFC(its,i) = (detect_corr_1(its,i)+0.00001).*store_correlator_1(its,i);
   end

   temp1=store_correlator_1(:,i);
   MFC_output_1(i)=max(temp1);
   idx_1(i)=find(temp1==max(temp1));

   temp2=image_PC_1(:,i);
   PCC_output_1(i)=max(temp2);
```

```
    idx_2(i)=find(temp2==max(temp2));


    temp2x=combined_PCC(:,i);
    PCC_output_1x(i)=max(temp2x);
    idx_2x(i)=find(temp2x==max(temp2x));


    temp1x=combined_MFC(:,i);
    MFC_output_1x(i)=max(temp1x);
    idx_1x(i)=find(temp1x==max(temp1x));
end


MFC_avg_ro_estx=mean(idx_1x)*.001 %estimation of r0 value for MFC
PCC_avg_ro_estx=mean(idx_2x)*.001 %estimation of r0 value for PCC
```

Section 5 - ROC Curves

```
PCCm1=mean(PCC_output_1);


PCCm0=mean(PCC_output_0);


PCCs1=std(PCC_output_1);


PCCs0=std(PCC_output_0);


PCC_thresh_max=max(PCC_output_1);


PCC_thresh_min=min(PCC_output_0);


PCC_step=(PCC_thresh_max-PCC_thresh_min)/10000;


PCC_thresh=PCC_thresh_min:PCC_step:PCC_thresh_max;


PCC_Pd_comb=1-cdf('norm',PCC_thresh,PCCm1,PCCs1);


PCC_Pfa_comb=1-cdf('norm',PCC_thresh,PCCm0,PCCs0);


MFCm1=mean(MFC_output_1);
```

```
MFCm0=mean(MFC_output_0);

MFCs1=std(MFC_output_1);

MFCs0=std(MFC_output_0);

MFC_thresh_max=max(MFC_output_1);

MFC_thresh_min=min(MFC_output_0);

MFC_step=(MFC_thresh_max-MFC_thresh_min)/10000;

MFC_thresh=MFC_thresh_min:MFC_step:MFC_thresh_max;

MFC_Pd_comb=1-cdf('norm',MFC_thresh,MFCm1,MFCs1);

MFC_Pfa_comb=1-cdf('norm',MFC_thresh,MFCm0,MFCs0);

PD_m1=mean(point_detector_1);

PD_m0=mean(point_detector_0);

PD_s1=std(point_detector_1);

PD_s0=std(point_detector_0);

PD_thresh_max=max(point_detector_1);

PD_thresh_min=min(point_detector_0);

PD_step=(PD_thresh_max-PD_thresh_min)/10000;
```

```
PD_thresh=PD_thresh_min:PD_step:PD_thresh_max;


Pd_short=1-cdf('norm',PD_thresh,PD_m1,PD_s1);


Pfa_short=1-cdf('norm',PD_thresh,PD_m0,PD_s0);


plot(log10(Pfa_short),Pd_short,'g')
title('Point Detector ROC Curve', 'fontsize', 20);
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')


plot(log10(MFC_Pfa_comb),MFC_Pd_comb,'r')
title('MFC ROC Curve', 'fontsize', 20);
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')


plot(log10(PCC_Pfa_comb),PCC_Pd_comb,'b')
title('PCC ROC Curve', 'fontsize', 20);
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')


plot(log10(PCC_Pfa_comb),PCC_Pd_comb,'b',log10(MFC_Pfa_comb),MFC_Pd_comb,'r',lo
g10(Pfa_short),Pd_short,'g')
title('Detectors Comparison ROC Curve', 'fontsize', 20);
legend('Pearsons Correlation Coefficient','Matched Filter','Point Detector',
'location','NorthWest')
ylabel('P_{detection}')
xlabel('Log 10 P_{false alarm}')
```

Section 6 - Comparing MFC vs PPCC

PD - PSF

```
EPd = mean(image_PC_1(:));
sig_Pd = std(image_PC_1(:));
EPd_Mfc = mean(store_correlator_1(:));
sig_Pd_Mfc = std(store_correlator_1(:));


Pd_PCC = 1-cdf('norm', 3/m, EPd, sig_Pd) % Probability of Detection with 3 std
Pd_MFC = 1-cdf('norm', 4.2357*sig_Pfa_Mfc, EPd_Mfc, sig_Pd_Mfc) % Probability of
Detection to match PCC
```

PFA - H_0

```
PCC_False_Alarms = sum(Keep_image_PC_0(:) == 1)
```

```
MFC_False_Alarms = sum(detect_corr_0(:) == 1)


PCC_Detections = sum(Keep_image_PC_1(:) == 1)
MFC_Detections = sum(detect_corr_1(:) == 1)


EPfa_PCC = mean(image_PC_0(:));
sig_Pfa_PCC = std(image_PC_0(:));
EPfa_MFC = mean(store_correlator_0(:));
sig_Pfa_MFC = std(store_correlator_0(:));


Pfa_PCC = 1-cdf('norm', 3/m, EPfa_PCC, sig_Pfa_PCC) % Probability of False Alarm
with 3 std
Pfa_Mfc = 1-cdf('norm', 4.2357*sig_Pfa_MFC, EPfa_MFC, sig_Pfa_MFC) % Probability of
False Alarm to match PCC
```

A.3) Make_Long_OTF MATLAB Code

```
function [otf] = make_long_otf(r1,dx,si,ro)
% [otf,apeture] = make_long_otf(r1,dx,si,ro);
% r1 is the radius of the receiver apertue in units of centimeters
% dx is chosen so that the number of pixels in your source array is
% greater than 4 times the number of pixels in your aperture radius
% si is the number of pixels across the array
% si*dx>4*r1
% dx>4*r1/si
% ro is the seeing parameter of the atmosphere the effect aperture
diameter
% of the atmosphere for resolution purposes

mi = floor(si/2);
mi=mi+1;
otf = zeros(si,si);
for i = 1:si

for j = 1:si

dist = sqrt((i-mi)^2+(j-mi)^2);
if(dist<=2*r1/dx)

otf(i,j)=exp(-3.44*((dist/(ro/(dx)))^(5/3)));

end
end
end
end
```

A.3) Make_OTF MATLAB Code

```
function [otf,apeture] = make_otf(r1,r2,si,scale,phase)
% [otf,apeture] = make_otf(r1,r2,si,scale,phase);

mi = floor(si/2);
mi=mi+1;
apeture = zeros(si,si);
for i = 1:si

for j = 1:si

dist = sqrt((i-mi)^2+(j-mi)^2);
if(dist<=r1)

if(dist>=r2)
apeture(i,j) = 1;

end
end
end
end

pupil = apeture.*cos(phase) + sqrt(-1)*apeture.*sin(phase);
psf = real(fft2(pupil).*conj(fft2(pupil)));
psf = scale*psf/sum(sum(psf));

otf = fft2(psf);
```

Bibliography

1. U.S. Department of Defense and Office of the Director of National Intelligence, National Security Space Strategy Unclassified Summary (2011), p. 14.

2. Office of the President of the United States of America, National Space Policy of the United Stated of America (United Sates Government, 2010) p. 18.

3. U.S. Congress, National Aeronautics and Space Administration Authorization Act of 2005 (U.S. Congress, 2005) pp. 2135-2321.

4. J. W. Goodman, "Introduction to Fourier Optics, Second Edition," *Opt. Eng.*, vol. 35, no. 5, 1996.

5. D. Becker and S. Cain, "Improved space object detection using short-exposure image data with daylight background," *Appl. Opt.*, vol. 57, no. 14, p. 3968, 2018, doi: 10.1364/ao.57.003968.

6. J. Chris Zingarelli, E. Pearce, R. Lambour, T. Blake, C. J. R. Peterson, and S. Cain, "Improving the space surveillance telescope's performance using multi-hypothesis testing," *Astron. J.*, vol. 147, no. 5, 2014, doi: 10.1088/0004-6256/147/5/111.

7. Scott L. Miller, "Probability and Random Processes, With Applications to Signal Processing and Communications," Edition 2, p. 37 & 575, 2012

8. J. W. Goodman, "Statistical Optics, Library Edition," *Opt. Eng.*, p. 528, 2000.

9. Richard D. Richmond, Stephen C. Cain, "Direct-Detection LADAR Systems," Volume TT85, p. 88, 92-94 & 101-104, 2010

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
25-03-2021	Master's Thesis	August 2019 – March 2021

TITLE AND SUBTITLE
Enhanced Space Object Detection Without Prior Knowledge of the Point Spread Function

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Graupmann, Grant F., 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/ENY)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT-ENG-MS-21-M-042

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Lt Col J. Chris Zingarelli PhD
Commander & Materiel Leader
Air Force Maui Optical and Supercomputing
Air Force Research Labs Det 15
(w) 808-891- 7701 (c) 571-386-8924
(e) john.zingarelli@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RDSM

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

Since the point detector was created, other detection algorithms have been created that increase the probability of detection, while still keeping the same probability of false alarm. The point detector still has uses, such as when there is no prior knowledge of the point spread function (PSF).

The matched filter correlator (MFC) detector is reliant on prior knowledge of the PSF. This has been an issue in cases where the PSF information is potentially inaccurate or unknown.

This thesis utilizes MFC detector in a manner that it has never been used before, along with a new detection algorithm, the Pearson's correlation coefficient (PCC) detector, in order to estimate Fried's Seeing Parameter (r_0) for a captured image. In previous studies, r_0 is known or is assumed to be known. This new method of estimating r_0 could yield higher probability of detection rates among certain space objects with little or no prior knowledge about the space object in question.

15. SUBJECT TERMS
Space Surveillance, Image Processing, Astronomy

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Stephen Cain, AFIT/ENG
U	U	U	UU	77	19b. TELEPHONE NUMBER (937) 255-3636 X4716 stephen.cain@afit.edu

Standard Form 298 (Rev. 8-98)

Prescribed by ANSI Std. Z39-18