SPATIALLY SEPARABLE BLIND DECONVOLUTION OF LONG EXPOSURE ASTRONOMICAL IMAGERY

THESIS

Justin Lee, 1st Lt, USAF

AFIT-ENG-MS-21-M-057

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-21-M-057

SPATIALLY SEPARABLE BLIND DECONCONVOLUTION OF LONG EXPOSURE ASTRONOMICAL IMAGERY

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

Justin S. Lee, B.S.E.E

1st Lt, USSF

March 2021

AFIT-ENG-MS-21-M-057

SPATIALLY SEPARABLE BLIND DECONCONVOLUTION OF LONG EXPOSURE ASTRONOMICAL IMAGERY

THESIS

Justin S. Lee, B.S.E.E.

1st Lt, USSF

Committee Membership:

Dr. Stephen C. Cain, PhD

Chair

Maj David Becker, PhD

Member

Maj Ronald Aung, PhD

Member

AFIT-ENG-MS-21-M-057

Abstract

In this thesis, a spatially separable blind deconvolution algorithm is demonstrated that achieves a significantly faster processing time and superior sensitivity when processing long-exposure image data of unresolvable objects from a ground-based telescope. The proposed approach takes advantage of the structure of the long exposure point spread function's radial symmetric characteristics to approximate it as a product of one-dimensional horizontal and vertical intensity distributions. Objects at geosynchronous or geostationary orbit also can be well approximated as being spatially separable as they are, in general non-resolvable.

The algorithm's performance is measured by computing the mean-squared error compared with the true object as well as the processing time required to perform the blind deconvolution. It will be shown that images processed by the proposed technique will possess, on average, a lower mean-squared error than images that are processed through the traditional two-dimensional blind deconvolution approach. In addition, the one-dimensional algorithm will be shown to perform the deconvolution significantly faster. In both cases the seeing parameter, and thus the point spread function, is treated as an unknown variable in the image reconstruction problem.

Table of Contents

List of Figures

List of Tables

SPATIALLY SEPARABLE BLIND DECONCONVOLUTION OF LONG EXPOSURE ASTRONOMICAL IMAGERY

I. Introduction

This chapter discusses the motivation of this research, the background of the space domain awareness problem, and the goals of this research. Finally, this chapter outlines the thesis organization.

1.1 Motivation

The motivation for this research is to improve the deconvolution of unknown, spatially separable objects with an unknown, long-exposure point spread function (PSF). Some examples of these spatially separable objects include distant stars, debris, and satellites in low earth orbit (LEO), geostationary earth orbit (GEO), and various other earth orbits. Currently, astronomers utilize a two-dimensional blind deconvolution algorithm to deconvolve an unknown object from an unknown point spread function (PSF). This two-dimensional algorithm is highly demanding of the core processing unit (CPU) and is typically applied in post-processing. A faster algorithm could be obtained using a one-dimensional blind deconvolution algorithm on spatially separable objects and PSFs. While there has been research done on the application of one-dimensional algorithms on spatially separable objects, this research focuses on the spatial separability of a PSF [1]. This would significantly decrease processing time through the elimination of all two-dimensional Fourier transforms, drastically decreasing processing time required.

1.2 Background

The capability to identify threats to national security in space is integral to the mission of the United States Space Force (USSF). Blind Deconvolution is a critical tool

used in the identification and categorization of space objects. As the number of satellites being sent into orbit increases exponentially, the ability to identify objects in space quickly and accurately becomes more important every day [2], [3]. Augmenting the problem is the that fact that not only are more objects being launched into orbit each year, they are also becoming smaller [4]. This abundance of small satellites and other objects near earth will test space agencies' ability to quickly and accurately identify threats to military and commercial space assets.

Blind deconvolution algorithms are often used to eliminate the effect of atmospheric turbulence caused by temperature fluctuations throughout the atmosphere. These algorithms are often very slow and thus are generally applied only in post processing [5]. A faster blind deconvolution would allow for on-site corrections or immediate follow-up imagery.

This research proposes a new blind deconvolution algorithm that would be able to blindly deconvolve a collection of objects both quickly and accurately, specifically for situations where both the object and PSF are spatially separable. This algorithm would leverage this separability in order to eliminate two-dimensional Fourier Transforms and replace them with one-dimensional operations. This will significantly decrease the time required to perform blind deconvolution and possibly enable real-time image processing.

1.3 Research Goals

The goal of this research is to derive a new, fully one-dimensional blind deconvolution algorithm for use with long-exposure astronomical imagery. This algorithm will operate under the assumption that both the object and the PSF are spatially

separable. This algorithm will be shown to perform deconvolution faster and more accurately than two-dimensional algorithms. Simulated experiments will show blind deconvolution of a binary star system.

1.4 Organization Overview

We begin by providing the necessary background knowledge required for this research in Chapter II. This background information includes Zernike polynomials, point spread functions, blind deconvolution, spatial separability, and the expectation maximization algorithm. In Chapter III we explore the methodology of performing this research. The results of both simulated and laboratory experiments are shown in Chapter IV. Finally, the research and results are summarized in Chapter V, including possible future work that could be done to build upon this research.

II. Background and Literature Review

This chapter provides the technical background necessary to understand this research. First, we discuss Zernike polynomials, their history, and use describing optical aberrations. Then we explore PSFs, the way they are defined, the different types, and their characteristics. Next, we introduce blind deconvolution, including its various uses and some specific examples. We also discuss the concept of spatial separability and provide examples of spatially separable and non-spatially separable images. Finally, we explore the general Expectation Maximization (EM) algorithm.

2.1 Zernike Polynomials

An effective method of describing optical aberrations was developed by the Dutch physicist Fritz Zernike [6], [7]. Known as Zernike Polynomials, these polynomials are

used to describe increasing higher order aberrations including piston, tilt, astigmatism, defocus, coma, etc. Examples of the various types of aberrations described by Zernike polynomials are shown in Fig. 1.

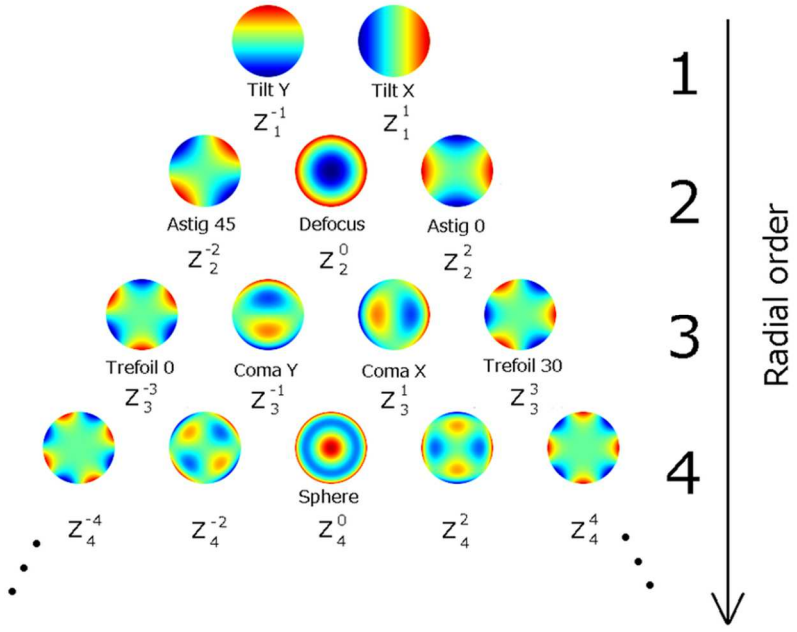

Fig. 1: Zernike Polynomials and Associated Optical Aberrations

The degree to which an optical system imparts each type of aberration is defined by the value of the coefficient in front if its respective ordered Zernike polynomial. The Zernike polynomials Z_n^m are defined in Eq. (1) where m and n are non-negative integers with $n \geq m \geq 0$, ρ is the radial distance (between 0 to 1), θ is the azimuthal angle, and R_n^m are the radial polynomials defined in Eq. (2).

$$m \neq 0 \begin{cases} {Z_n^m}_{even}(\rho,\theta) = \sqrt{n+1}R_n^m(\rho)\cos(m\theta) \\ {Z_n^m}_{odd}(\rho,\theta) = \sqrt{n+1}R_n^m(\rho)\sin(m\theta) \end{cases} \quad (1)$$

$$0 \le \rho < 1 \qquad R_n^m(\rho) = \sum_{k=0}^{\frac{n-m}{2}} \frac{(-1)^k (n-k)!}{k!\left(\frac{n+m}{2}-k\right)!\left(\frac{n-m}{2}-k\right)!} \rho^{n-2k} \tag{2}$$

$$\rho = 1 \qquad R_n^m(\rho) = 1$$

2.2 Point Spread Function

The PSF, also known as the impulse response function, describes an imaging system's response to a point source. In astronomy applications, the PSF describes the form and extent of blurring imparted by the optical system and/or atmosphere through which it is imaging. Both the atmospheric and optical PSF can be described separately and the combined PSF involving both atmospheric and optical factors can be described as the convolution of the two separate PSFs.

The aberrations imposed by the optical system can be clearly described and defined using Zernike polynomials as explained in the previous section. An example of optical elements causing significant aberration is the Hubble Space Telescope (HST). Immediately following the HST's launch in 1990, a flaw was discovered in the observatory's primary mirror [8]. Spherical aberration, Zernike polynomial 11, seriously affected the clarity of the telescope's early images, preventing it from performing many of its planned missions[9]. Using the Corrective Optics Space Telescope Axial Replacement (COSTAR), this issue was resolved in 1993.[10] The PSF both before and after COSTAR was applied is shown in Fig. 2.

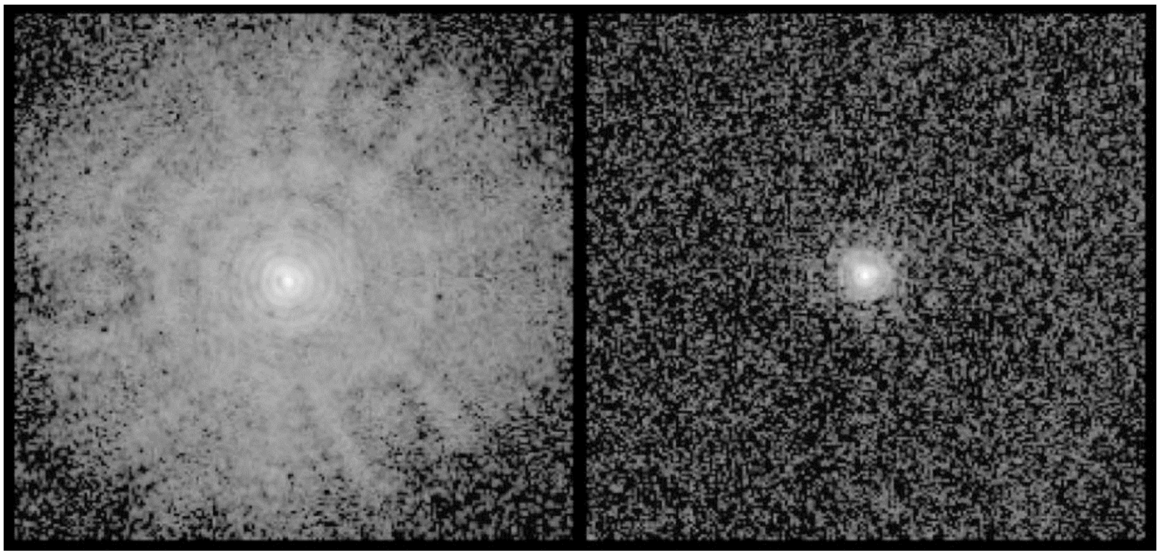

Fig. 2: Hubble Telescope PSF Before (left) and After (right) Corrective Optics

Turbulent atmospheric mixing causes apparent blurring and twinkling of astronomical objects when viewed from earth. As light passes through approximately 100 km of atmosphere, it encounters many areas of differing temperature and humidity, as shown in Fig. 3. [11] This phenomenon was mathematically described in 1941 by Andrey Kolmogorov. [12] These conditions cause light to move at different speeds and to refract in various directions, causing aberrations in astronomical telescopes that can be expressed in terms of the Zernike polynomials that lead to a distorted or blurred image.[13]

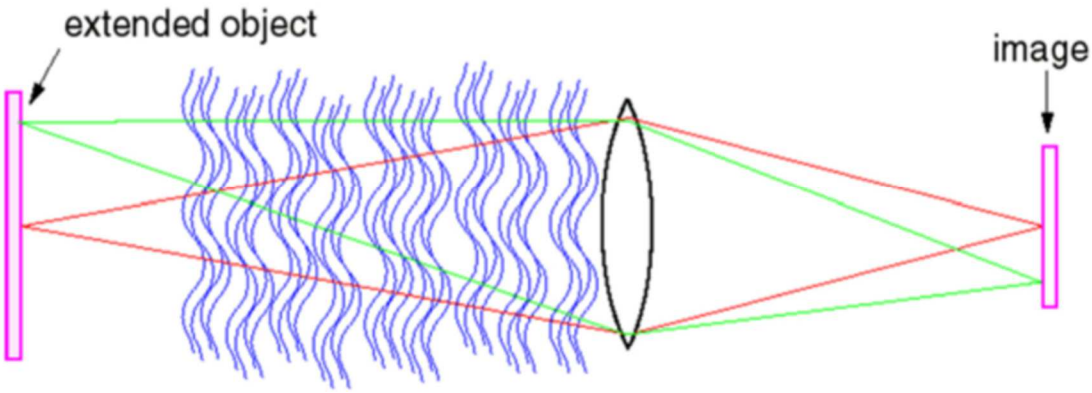

Fig. 3: Atmospheric Turbulence Model

When an image is taken using a short exposure time, the atmospheric PSF can appear random and sporadic, with examples provided in Fig. 4 [13].

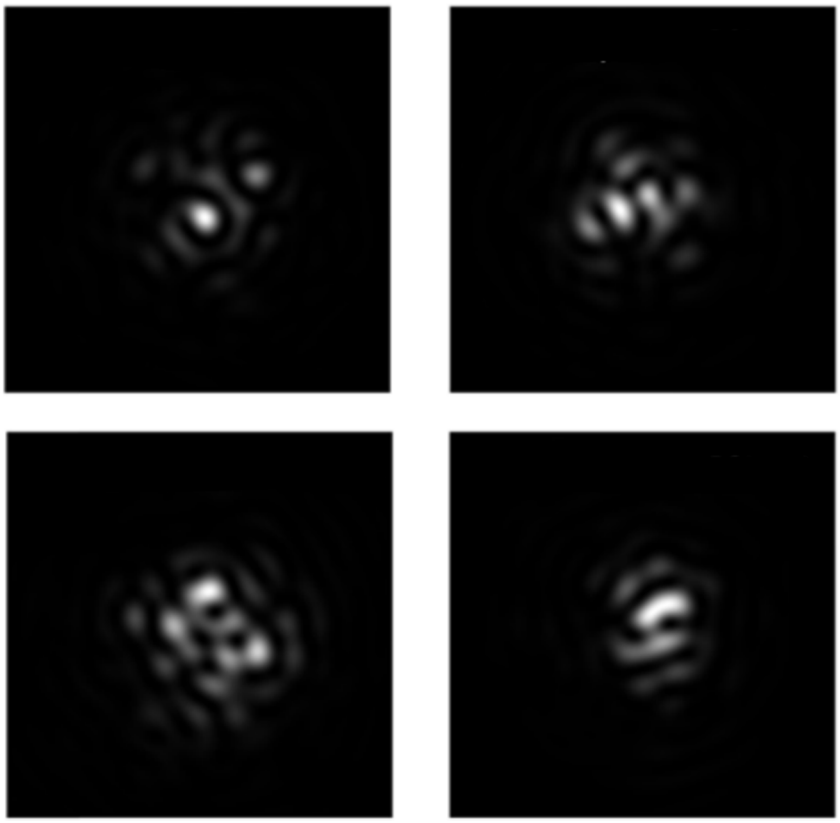

Fig. 4: Short Exposure PSF Examples

When long-exposure images are taken, the PSF tends to take the shape of a two-dimensional Gaussian unless adaptive optics are used to compensate for the PSF in real time [14]. In the specific case of adaptive optics the PSF tends to take on the shape of a two-dimensional Lorentzian distribution [15]. An example of a long-exposure non-adaptive optics PSF is shown in Fig. 5.

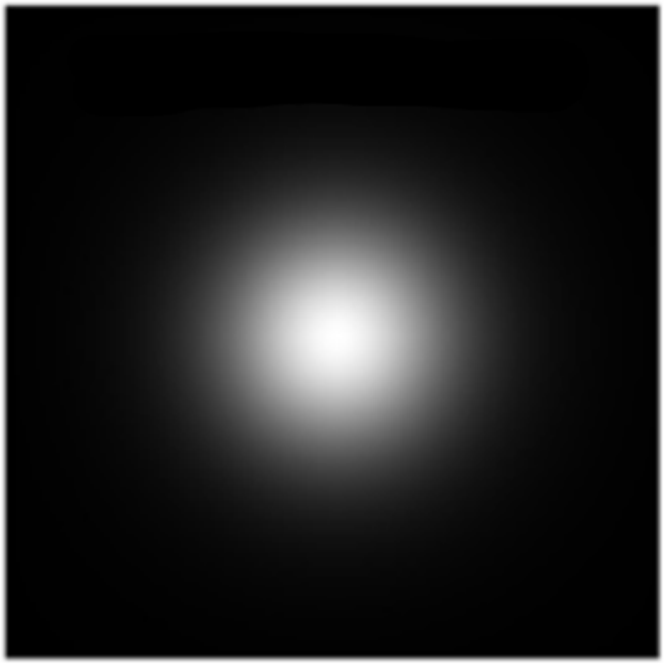

Fig. 5: Long Exposure PSF Example

We can measure the extent of blurring imparted by the atmosphere using Fried's seeing parameter, r_0 [16]. Fried built upon the work of Kolmogorov's work to accurately describe the degree of atmospheric using r_0, which is measured in units of centimeters [12] [17].

As atmospheric distortion increases, r_0 decreases. Examples of generated PSFs with various r_0 values are shown in Fig. 6 for a one-meter diameter telescope imaging at a visible wavelength of 0.5 micrometers.

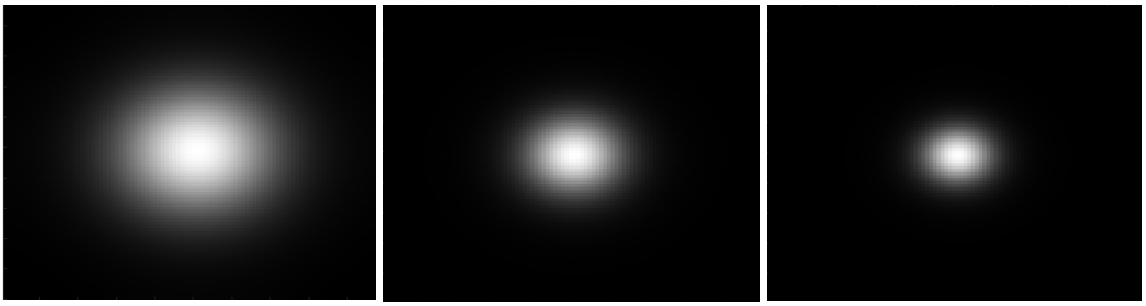

Fig. 6: Long-Exposure PSF Examples with r_0 values of 6cm (left), 10cm (middle), and 14cm (right)

When the imaging system is space-invariant, an image can be described mathematically as the convolution of the PSF and the object [18]. An example of this is shown below in Fig. 7 and Fig. 8.

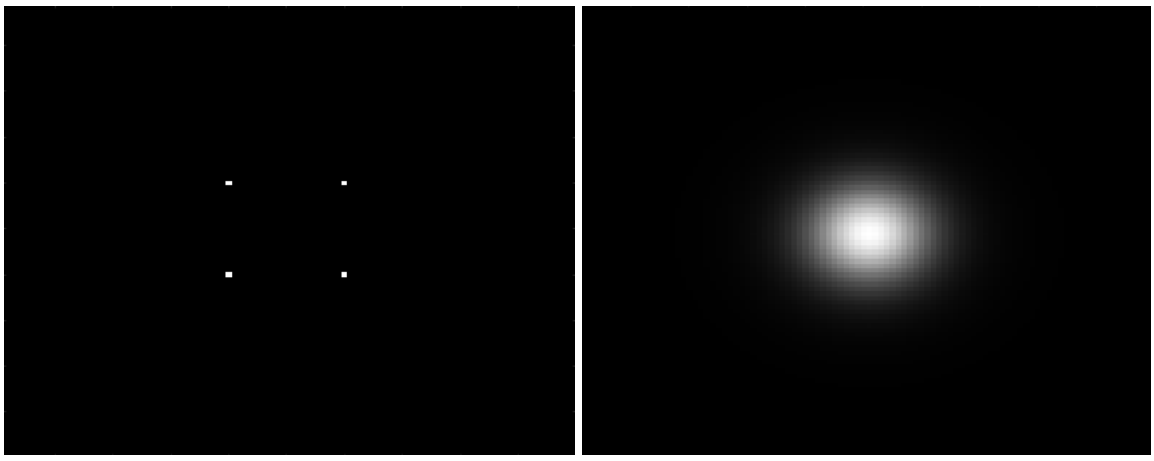

Fig. 7: Object (left) and PSF (right) Example

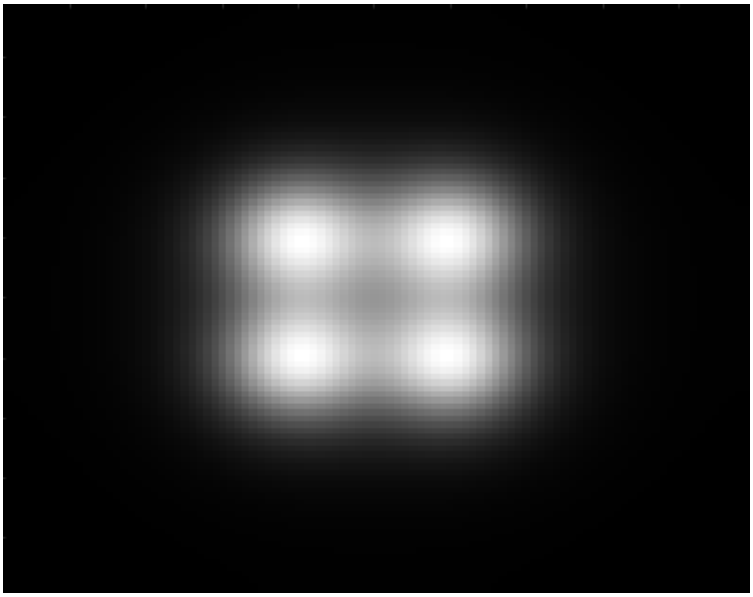

Fig. 8: Convolution of Object and PSF Example

2.3 Blind Deconvolution

Using properties of Fourier transforms, we can define convolution as shown in Eq. (3) where *i* is the two-dimensional image, *z* and *w* describe pixels in the detector plane, where the image is captured, *x* and *y* describe pixels in the source plane, *o* is the object being imaged, *h* is the PSF, *f* is the two-dimensional spatial frequency, *O* is the Fourier transform of the object, o, and *H* is the Fourier transform of the PSF, h.

$$i(z,w) = \sum_{x}\sum_{y} o(x,y)h(z-x,w-y) \Leftrightarrow I(f) = O(f)H(f) \tag{3}$$

Deconvolution is the inverse of convolution. Because an image is the convolution of the object being imaged with PSF, we can use deconvolution to separate the two functions. When we know two of the three functions in Eq. (3) we can directly solve for

the remaining unknown in the Fourier domain, except at regions where one of the functions is equal to zero, or nearly so. A typical example of this would be when the image and either the PSF or object is known, but the other is unknown.

However, when both the PSF and the object are not known a method called blind deconvolution must be performed. Blind deconvolution is the estimation of an object and PSF without knowledge of either. Since 1976 many different techniques have been used to perform blind deconvolution, including maximum a posteriori (MAP) estimation [19]. A direct, non-iterative approach was proposed by Carasso that involves interactive tuning and estimation of the PSF using Fourier analysis [20]. The total variational minimization (TV) method is highly effective in recovering sharp edges [21]. Another method is the Alternating Minimization Algorithm which is a joint minimization model [22]. These are only a few examples of the many different approaches to blind deconvolution and there are many others.

2.4 Spatial Separability

A two-dimensional spatially separable image is one which can be separated into two one-dimensional vectors without loss of information. We can solve for these horizontal and vertical vectors by summing along the y- and x-axis, respectively. This is shown in Eq. (4) and Eq. (5) where *i* is the two-dimensional image, i_y is the vertical component vector, i_x is the horizontal component vector. Once again and throughout this paper *x* and *y* describe pixels in the source plane, *z* and *w* describe pixels in the detector plane.

$$i_y(w) = \sum_z i(z, w) \tag{4}$$

$$i_x(z) = \sum_w i(z, w) \tag{5}$$

If an image is spatially separable, the multiplication of the two component vectors will accurately reproduce the original, two-dimensional image. Fig. 9 gives an example of a spatially separable image before separation and after reconstruction.

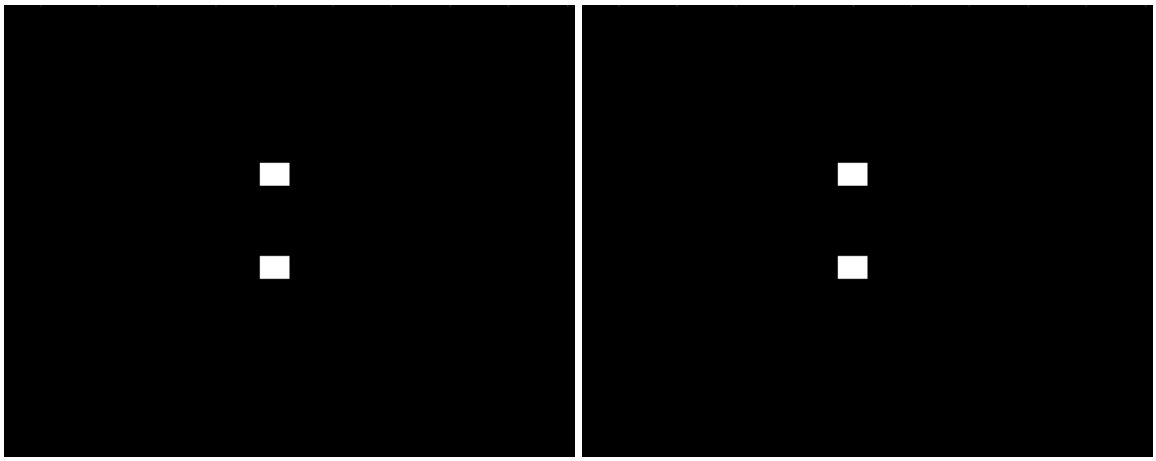

Fig. 9: Separable Example, True Image (left) Reconstructed Image (right)

We observe that the original image was accurately reconstructed from the one-dimensional component vectors shown in Fig. 10.

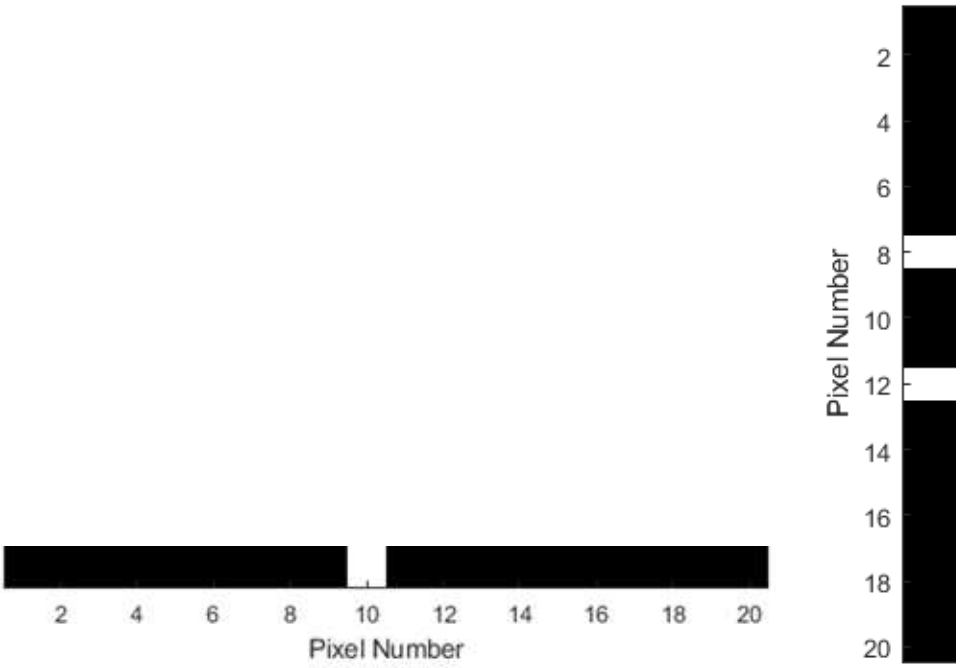

Fig. 10: Separable Ex. #1, Horizontal (left) and Vertical (right) Component Vectors

Other examples of spatially separable objects are shown below in Fig. 11. There are, of course, many other examples of spatially separable images.

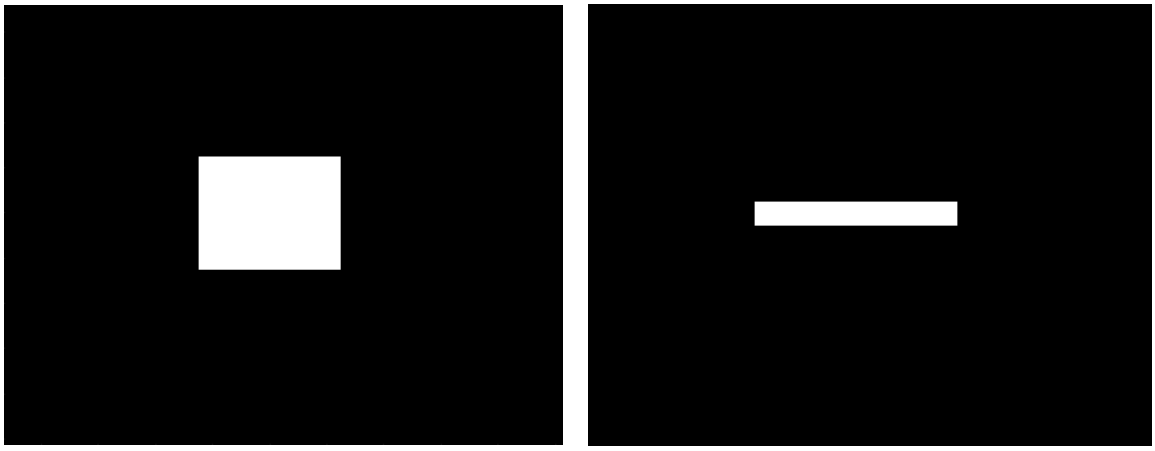

Fig. 11: Spatially Separable Object Examples

However, many objects are not spatially separable. We will explore a non-separable example that is very similar to the example shown in Fig. 9. We take our original, separable example and shift the bottom point source to the right by one pixel. We then separate the image into two orthogonal vector components as described in Eq. (4) and Eq. (5). Finally, we once again multiply these two vectors to form a reconstructed image. The original and reconstructed images are shown in Fig. 12 and the component vectors are shown in Fig. 13.

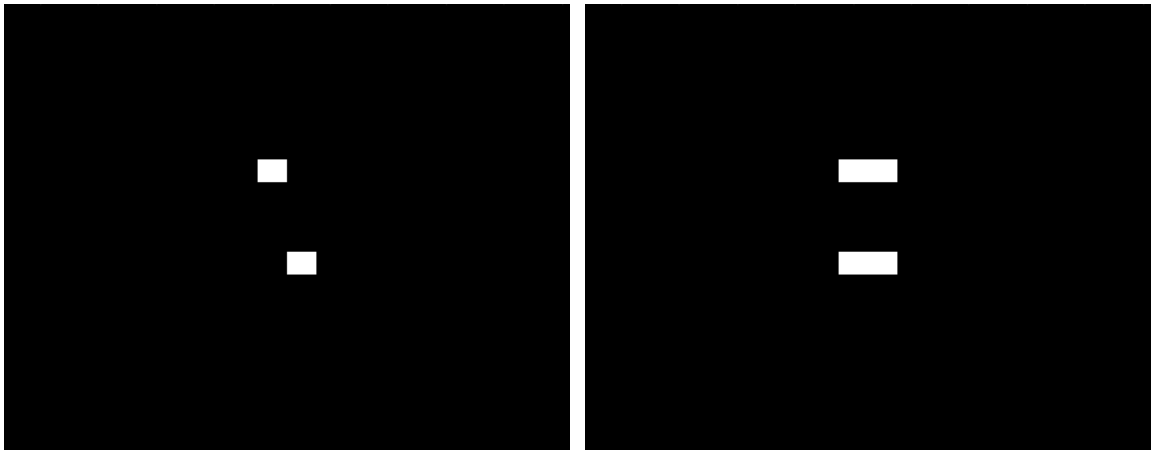

Fig. 12: Non-Separable Ex. #1, Original (left) and Reconstructed (right)

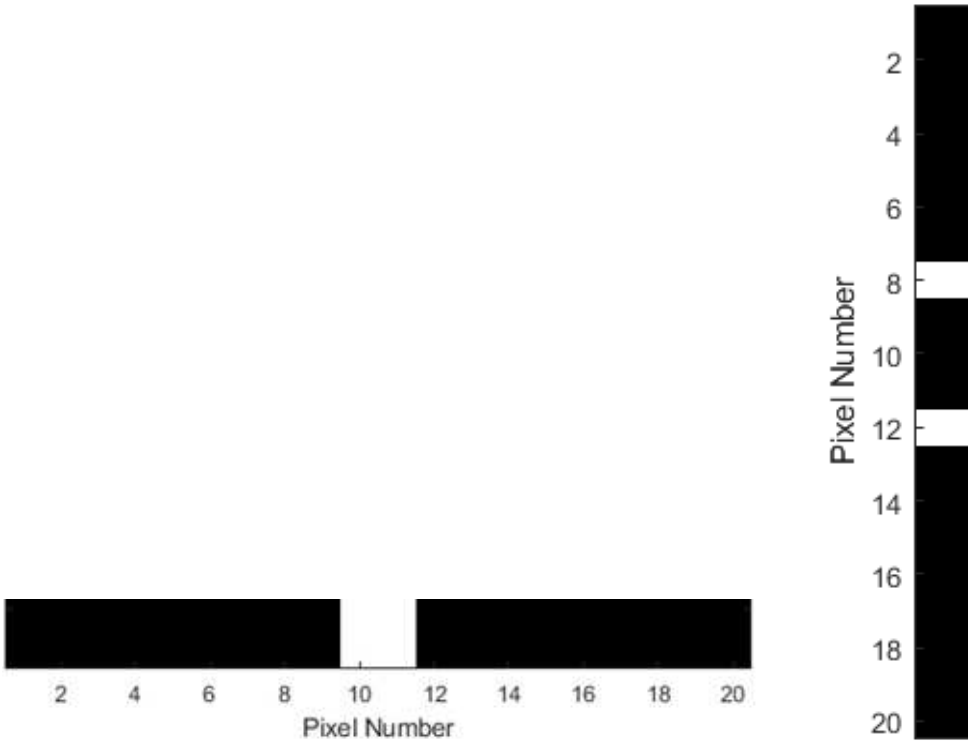

Fig. 13: Non-Separable Ex. #1, Horizontal (left) and Vertical (right) Component Vectors

We observe that the original image was not accurately reconstructed in this case, meaning that this is an example of a non-spatially separable image.

2.5 Expectation Maximization

The EM algorithm is a technique used in maximum-likelihood estimation problems that can estimate parameters of a probability distribution function [23]. The ideas involved in the EM algorithm were independently discovered and utilized by many different researchers until the ideas were brought together, formally proved, and named by Arthur P. Dempster in 1977 [24]. The EM algorithm is useful in situations where the observed data can be viewed as incomplete. The term “incomplete data” implies the existence of two sample spaces. These are both the sample space from which the incomplete data is taken, defined as $\mathscr{A}$, and the sample space from which the complete data is taken, defined as $\mathscr{B}$, with a many-to-one mapping from $\mathscr{A}$ to $\mathscr{B}$[24]. Some examples of fields that meet these criteria, and therefore wherein the EM algorithm proves exceptionally useful, include genetics, mixture distributions, and many types of studies that have unknown factors affecting the outcomes [25], [26], [27], [28], [29].

The EM algorithm is used to produce update equations that can be used to process atmospheric data. These update equations are highly effective in processing atmospheric image data with the incomplete data being the collected image and the complete data being the true object. For further examples of this application see the following references [23], [30], [31], [1].

III. Methodology

3.1 Overview

In this chapter we outline the process used to derive the iterative update equations to be used in the proposed algorithm. First, we explain the various models utilized. Next, we show the spatial separability of both objects and PSFs to be utilized. Finally, we use the steps of the EM algorithm to arrive at the update equations used to estimate the object and PSF at each iteration.

3.2 Models Utilized

In this section we discuss the models utilized in the algorithm derivation and simulation. First, we explore the model used to represent the image taken of the satellite, then the mathematical models describing both the optical and atmospheric transfer functions, and finally the mathematical model describing the data.

3.2.1 Satellite image model

When imaging objects in GEO from the earth's surface, they are typically unresolvable. Satellites at GEO can therefore be modeled as point-sources. Throughout this paper collections of Dirac delta functions will be used to model collections of satellites in GEO.

3.2.2 Transfer Functions

The simulated total transfer function will be modeled as the product of the optical transfer function and the atmospheric transfer function. The inverse Fourier transform will then be taken to produce the total PSF as shown in Equations (6) and (7). The OTF, H_{opt}, and long-exposure transfer function, H_s, are defined in the following sections, where

f_x and f_y are spatial frequencies, n and m are pixel coordinates, and P_{tot} is the point spread function of the system.

$$H_{tot}(f_x, f_y) = H_{opt}(f_x, f_y) \times H_s(f_x, f_y) \quad (6)$$

$$P_{tot}(n, m) = \mathcal{F}^{-1}\{H_{tot}(f_x, f_y)\} \quad (7)$$

3.2.2.1. Optical Transfer Function

The OTF due to the optics , H_{opt}, can be described as the autocorrelation of the pupil function, P. This is shown below in Eq. (8). This model is valid for systems with or without optical aberrations.

$$H_{opt}(f_x, f_y) = \frac{\iint_{-\infty}^{\infty} P\left(w + \frac{f_x}{2}, y + \frac{f_y}{2}\right) P^*\left(w - \frac{f_x}{2}, z - \frac{y_z}{2}\right) dwdz}{\iint_{-\infty}^{\infty} |P(w, z)|^2 \, dwdz} \quad (8)$$

3.2.2.2 Long-Exposure Atmospheric Transfer Function

In long-exposure imaging, the atmospheric PSF can be accurately modeled as the average of many short-exposure PSFs. In this paper, we can assume that the integration time is sufficiently long to allow many instances of atmospheric turbulence to produce PSFs to mimic a long-term average. We will therefore simulate the impulse response of the system using the long exposure OTF computed previously by Fried [17]. This

description of the long exposure OTF is shown below in Eq. (9). The only variable that will be randomized in simulating the PSF is the seeing parameter, represented below as r_0. The spatial frequency in radians^{-1} is represented as $\sqrt{f_x^2 + f_y^2}$, θ_0 is a specific single value of spatial frequency [32]. Also, in Eq. (9) $\bar{\lambda}$ is the average wavelength of the light and α is a parameter related to adaptive optics. Because we are dealing with long exposure PSFs and not utilizing any adaptive optics in this experiment, we can set alpha equal to zero, yielding Eq. (10)

$$H_s(f_x, f_y) = e^{-3.44\left(\frac{\bar{\lambda}\left(\sqrt{f_x^2+f_y^2}\right)}{r_0}\right)^{5/3}\left[1-\alpha\left(\frac{\sqrt{f_x^2+f_y^2}}{\theta_0}\right)^{1/3}\right]} \tag{9}$$

$$H_s(f_x, f_y) = e^{-3.44\left(\frac{\bar{\lambda}\sqrt{f_x^2+f_y^2}}{r_0}\right)^{5/3}} \tag{10}$$

3.2.3 Data

Statistical models for the incomplete (collected) data and the complete data are defined in the following sections. The baseline algorithm is verified with computer simulated data. The MATLAB programming language is used for all coding implementation on a standard desktop computer.

3.2.3.1 Statistical Model for Incomplete Data

The incomplete data is mathematically described as the sum of the complete data as shown in Eq. (11). The incomplete data is defined as d_k and the complete data is described as $\zeta_{k.}$, x and y describe pixels in the source plane, z and w describe pixel coordinates in the detector plane. The data due to the background is defined as d_B and its mean is defined in Eq. (12).

$$d_k(z,w) = \sum_x \sum_y \zeta_k(z,w,x,y) + d_B(z,w) \tag{11}$$

$$B = E[d_B(z,w)] \tag{12}$$

Because the majority of this algorithm will use one-dimensional vectors, we'll first separate the incomplete data simply by summing along each axis of the two-dimensional data as shown below in Equations (13) and (14).

$$d_{k1}(z) = \sum_w d_k(z,w) \tag{13}$$

$$d_{k2}(w) = \sum_z d_k(z,w) \tag{14}$$

Substituting our definition of the incomplete data from Eq. (11) into Equations (13) and (14) we obtain Equations (15) and (16), respectively.

$$d_{k1}(z) = \sum_{w} [\sum_{x} \sum_{y} \zeta_k(z, w, x, y) + d_B(z, w)] \tag{15}$$

$$d_{k2}(w) = \sum_{z} [\sum_{x} \sum_{y} \zeta_k(z, w, x, y) + d_B(z, w)] \tag{16}$$

We define each of the component vectors containing the background as shown below in Eq. (17) and Eq. (18). We also define B_1 and B_2 as the mean value of the background vectors as shown in Eq. (19) and Eq. (20).

$$d_{B1}(z) = \sum_{w} d_B(z, w) \tag{17}$$

$$d_{B2}(w) = \sum_{z} d_B(z, w) \tag{18}$$

$$B_1 = E[d_{B1}(z)] \tag{19}$$

$$B_2 = E[d_{B2}(w)] \tag{20}$$

Using Eq. (17) and Eq. (18) we can then define d_{k1} and d_{k2} in Eq. (21) and Eq. (22), respectively.

$$d_{k1}(z) = \sum_{w}\sum_{x}\sum_{y} \zeta_k(z,w,x,y) + d_{B1}(z) \tag{21}$$

$$d_{k2}(w) = \sum_{z}\sum_{x}\sum_{y} \zeta_k(z,w,x,y) + d_{B2}(w) \tag{22}$$

We can also describe the statistical expected value of the incomplete data in Eq. (23) as the image intensity detected for a specific frame of data, where i_k is the image intensity, o_1 is the vertical vector component that describes the object when multiplied with o_2, the horizonal vector component of the object. The PSF is also separated into vertical, h_1, and horizontal, h_2, components. The image intensity is the convolution of the spatially separated object and PSF as shown in Eq. (24).

$$E[d_k(z,w)] = i_k(z,w,o_1,o_2,h_1,h_2) \tag{23}$$

$$i_k(z,w,o_1,o_2,h_1,h_2) = \sum_{x}\sum_{y} o_1(x)o_2(y)h_1(z-x)h_2(w-y) + B \tag{24}$$

We now separate the expectation of $d_k(z,w)$ shown in Equations (23) and (24) into one-dimensional vectors, starting with $d_{k1}(z)$ as shown below in Equations (25)-(33).

$$E[d_{k1}(z)] = i_{k1}(z, o_1, h_1) \tag{25}$$

$$i_{k1}(z, o_1, h_1) = \sum_w \sum_x \sum_y o_1(x) o_2(y) h_1(z - x) h_2(w - y) + B_1 \tag{26}$$

$$i_{k1}(z, o_1, h_1) = \sum_x \sum_y [\sum_w h_2(w - y)] o_1(x) o_2(y) h_1(z - x) + B_1 \tag{27}$$

By definition, a two-dimensional PSF sums to one. Therefore, because they are formed by summing along one of the axis, the horizontal or vertical component vectors of the PSF also sum as shown in Eq. (28). Using this property, we can simplify Eq. (27) yielding Eq. (29).

$$\sum_w h_2(w - y) = 1 \tag{28}$$

$$i_{k1}(z, o_1, h_1) = \sum_x \sum_y o_1(x) o_2(y) h_1(z - x) + B_1 \tag{29}$$

For this algorithm to work correctly, o_2 must sum to one as shown in Eq. (30). To ensure that this happens, we will utilize a Lagrange multiplier which will be discussed further in the next section and is expounded upon in Equations (65) - (67).

$$\sum_{y} o_2(y) = 1 \tag{30}$$

$$i_{k1}(z, o_1, h_1) = \sum_{x} \left[\sum_{y} o_2(y) \right] o_1(x) h_1(z - x) + B_1 \tag{31}$$

$$i_{k1}(z, o_1, h_1) = \sum_{x} o_1(x) h_1(z - x) + B_1 \tag{32}$$

Similarly, for d_{k2}(w), we now separate the expectation of d_k(z,w) shown in Equations (23) and (24) into the other one-dimensional vector.

$$E[d_{k2}(w)] = i_{k2}(w, o_2, h_2) \tag{33}$$

$$i_{k2}(w, o_2, h_2) = \sum_{z} \sum_{x} \sum_{y} o_1(x) o_2(y) h_1(z - x) h_2(w - y) + B_2 \tag{34}$$

$$i_{k2}(w, o_2, h_2) = \sum_{x} \sum_{y} [\sum_{z} h_1(z - x)] o_1(x) o_2(y) h_2(w - y) + B_2 \tag{35}$$

Again, a PSF sums to one as shown above in Eq. (28). Using this property, we can simplify Eq. (35) yielding Eq. (36).

$$i_{k2}(w, o_2, h_2) = \sum_x \sum_y o_1(x) o_2(y) h_2(w - y) + B_2 \quad (36)$$

In this case, we can treat the sum of o_1 as a known constant value. This is because our algorithm will first estimate o_1 independently of o_2. We then can then solve for the sum of our o_1 estimate before estimating o_2 which becomes the constant value termed as o_1^{sum}. We can then simply multiply by o_1^{sum} as shown in Eq. (38).

$$i_{k2}(w, o_2, h_2) = \sum_y [\sum_x o_1(x)] o_2(y) h_2(w - y) + B_2 \quad (37)$$

$$i_{k2}(w, o_2, h_2) = \sum_y o_1^{sum} o_2(y) h_2(w - y) + B_2 \quad (38)$$

3.2.3.2. Statistical Model for Complete Data

The complete data is observed indirectly through the incomplete data. The complete data is described as many Poisson-distributed random variables. Their mean is shown below in Eq. (39).

$$E[\zeta_k(z, w, x, y)] = o_1(x) o_2(y) h_1(z - x) h_2(w - y) \quad (39)$$

We can also separate this expectation of the complete data as shown below in Equations (40) and (41).

$$E[\zeta_{k1}(z,x)] = o_1(x)h_1(z-x) \tag{40}$$

$$E[\zeta_{k2}(w,y)] = o_2(y)h_2(w-y) \tag{41}$$

3.3 Spatial Separability

In this section we will demonstrate the spatial separability of both the object being imaged and the long-exposure PSF. It will be shown that the types of functions to be utilized with this algorithm are indeed spatially separable.

3.3.1 Object Separability

Any of the functions shown to be separable in section 2.4 Spatial Separability, can be utilized. The specific object utilized throughout this paper will be a horizontally spaced binary star system as shown below in Fig. 14.

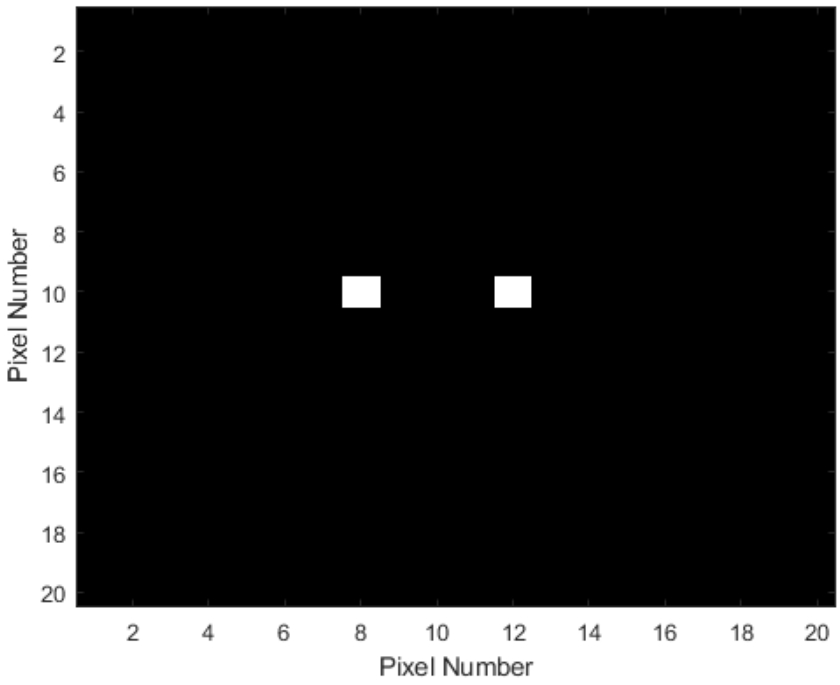

Fig. 14: Horizontal Binary Star System, True Object

We obtain the horizontal and vertical componant vectors as described in previous sections, shown in Fig. 15, and multiply them together to obtain the reconstructed object shown in Fig. 16.

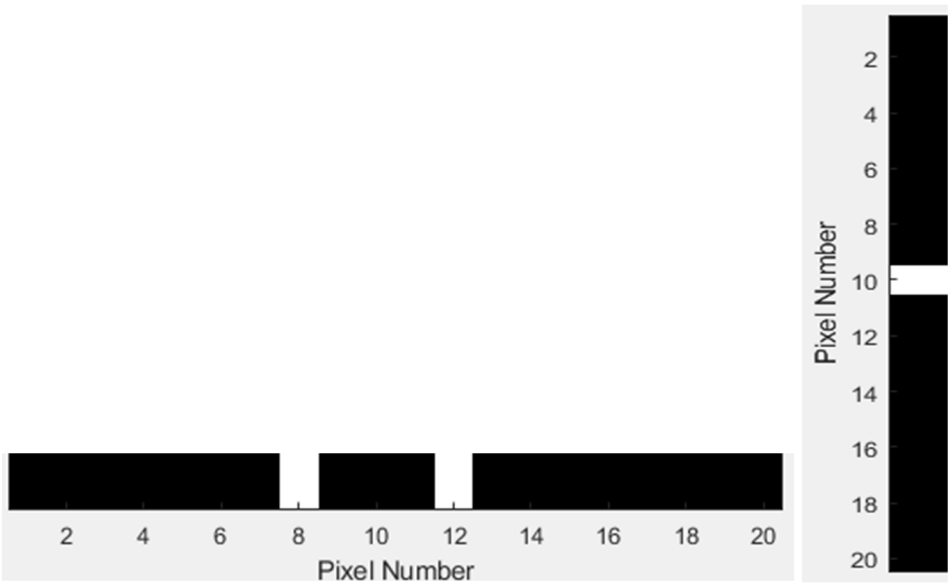

Fig. 15: Horizontal Binary Star System, Horizontal (left) and Vertical (right) Component Vectors

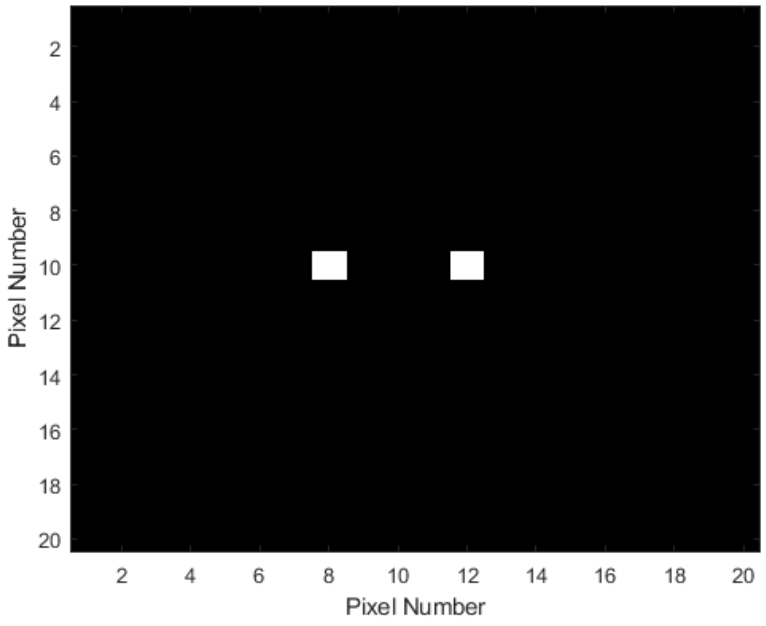

Fig. 16: Horizontal Binary Star System, Reconstructed Object

Upon observation, we see that the image reconstruction was successful. Therefore, this horizontal binary star system can be considered spatially separable.

3.3.2 PSF Separability

In this section we will demonstrate that a long-exposure atmospheric PSF is spatially separable. In addition to visual comparison of the true image against the reconstructed image we can calculate the Mean-Squared Error (MSE) between the true PSF and reconstructed PSF. If the MSE is equal to zero, or very nearly so, we assume that the function is spatially separable.

The MSE is calculated using Eq. (42), where m is the number of pixels on the x-axis, n is the number of pixels on the y-axis, $T(x,y)$ is the true intensity value of the PSF at pixel (x,y), and $h_1(y)$ and $h_2(x)$ are the estimated values of the spatially separated vector components of the PSF at pixels x and y.

$$MSE = \frac{1}{mn}\sum_{x=1}^{m}\sum_{y=1}^{n}[T(x,y) - h_1(y)h_2(x)]^2 \tag{42}$$

Using Eq. (8) we generate a simulated Optical PSF and using Eq. (10) we generate an atmospheric transfer function. Then using Eq. (6) and Eq. (7) we obtain the combined PSF. For the purpose of these tests, the r_0 value chosen can be arbitrary as the shape of the PSF is consistent regardless of the r_0 value chosen [17]. For this example, we will use an r_0 value of 10cm. Multiplication of the horizonal and vertical vector components of the PSF produces the reconstructed PSF. Our example long-exposure PSF

and the reconstruction are shown in Fig. 17. The spatially separated horizontal and vertical vector components are shown in Fig. 18.

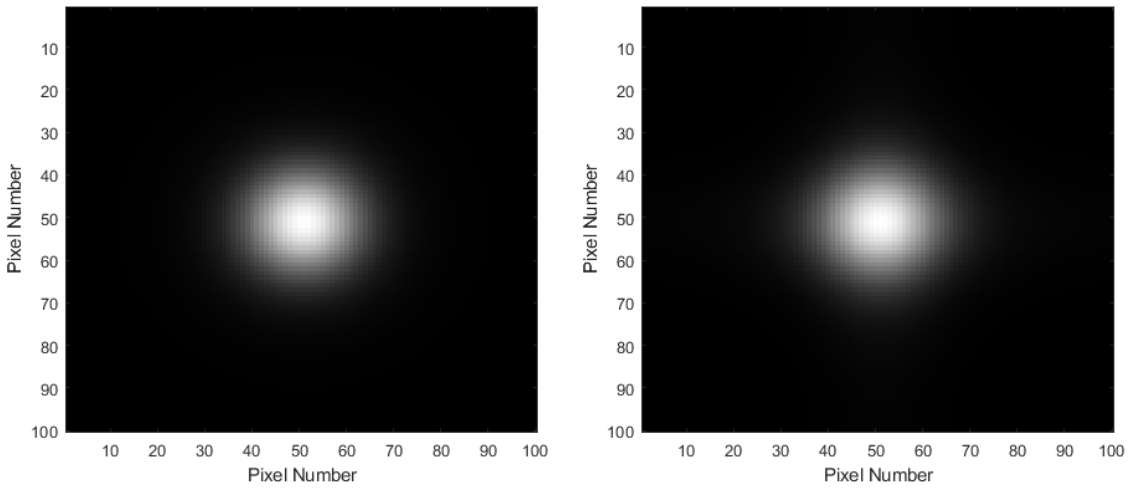

Fig. 17: Original (Left) and Reconstructed (Right) PSFs (r_0 = 10cm)

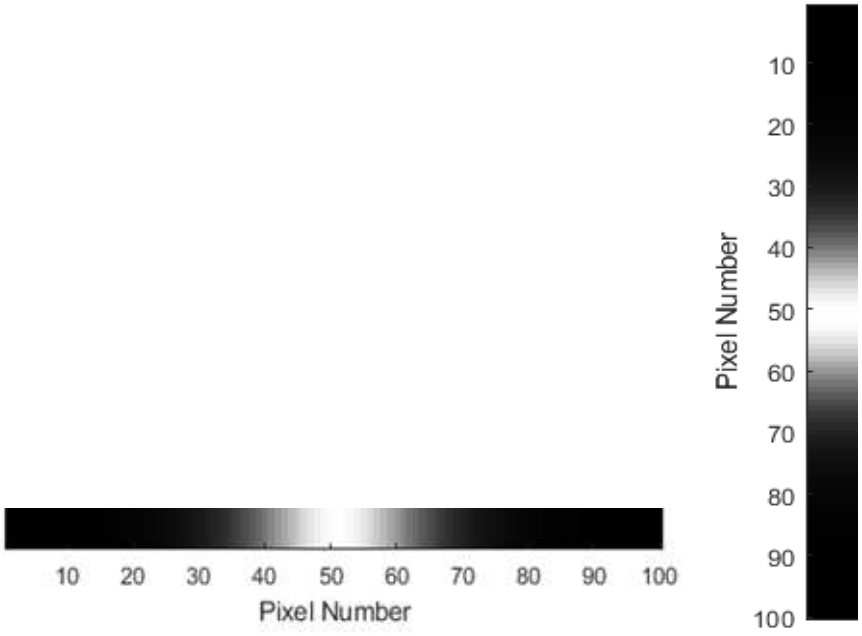

Fig. 18: PSF (r_0 = 10cm), Horizontal (left) and Vertical (right) Component Vectors

Visually we see that the reconstructed PSF appears be an accurate representation of the original PSF. To confirm we calculate the MSE between the original and reconstructed PSF using Eq. (42). The MSE is found to be 4.03×10^{-10}, which leads us to believe that a long-exposure PSF is spatially separable, or at least very nearly so. It will be further shown in Chapter IV that the long-exposure PSF is separable enough for a one-dimensional algorithm to produce accurate estimates and reconstructions.

3.4 Expectation-Maximization Algorithm

This section gives an overview of the EM algorithm, and then details the EM algorithm steps performed for this research.

3.4.1 EM Overview

The EM algorithm is an effective method of solving for an unknown object and an unknown PSF introduced from the atmosphere and telescope. Each iteration establishes an increasingly accurate estimate of the object and PSF. The two major steps of the EM algorithm are the expectation step and the maximization step, with each having various sub-steps [24]. The steps and derivation found below show the underlying mathematical models and operations that make this blind deconvolution possible. The steps of the EM algorithm used in this paper are found below.

1. Generate Probability Function for Incomplete Data
2. Generate Log-Likelihood Function
3. Derive Expectation
4. Maximize Expectation

5. Solve for Expected Value of Complete Data Given Incomplete Data

6. Solve for Update Equations

3.4.2 Generate Probability Function for Incomplete Data

The incomplete data is assumed to be Poisson-distributed at each point on the detector plane. Measurements in each pixel and frame are also assumed to be statistically independent from one another. We can, therefore, define a probability function for the incomplete data in one pixel of one frame below in Eq. (43).

$$P[d_k(z,w)] = \frac{i_k(z,w,o_1,o_2,h_1,h_2)^{d_k(z,w)} e^{-i_k(z,w,o_1,o_2,h_1,h_2)}}{d_k(z,w)!} \tag{43}$$

We will use the EM algorithm to obtain two separate update equations for o_1 and o_2. At each step of the EM algorithm, we will calculate separately for o_1 and o_2. The incomplete data functions for each component of the incomplete data are expressed below in Eqs. (44) and (45). The vertical component of the image intensity is represented by i_{k1} and the vertical component of the incomplete data is represented by d_{k1}. The horizontal vector components of intensity and incomplete data are likewise represented by i_{k1} and d_{k1}.

$$P[d_{k1}(z)] = \frac{i_{k1}(z,o_1,h_1)^{d_{k1}(z)} e^{-i_{k1}(z,o_1,h_1)}}{d_{k1}(z)!} \tag{44}$$

$$P[d_{k2}(w)] = \frac{i_{k2}(w, o_2, h_2)^{d_{k2}(w)} e^{-i_{k2}(w,o_2,h_2)}}{d_{k2}(w)!} \tag{45}$$

Because each pixel and frame are independent of each other, we can calculate the total probability of the incomplete data simply by multiplying the individual probabilities. This is shown in Eqs. (46) and (47) where *k* is defined the frame number.

$$P[d_{k1}(z) \forall (z,k) \in [1,N],[1,K]] \tag{46}$$
$$= \prod_k \prod_z \frac{i_{k1}(z, o_1, h_1)^{d_{k1}(z)} e^{-i_{k1}(z,o_1,h_1)}}{d_{k1}(z)!}$$

$$P[d_{k2}(w) \forall (w,k) \in [1,N],[1,K]] \tag{47}$$
$$= \prod_k \prod_w \frac{i_{k2}(w, o_2, h_2)^{d_{k2}(w)} e^{-i_{k2}(w,o_2,h_2)}}{d_{k2}(w)!}$$

3.4.3 Generate Log-Likelihood Function

To enable much simpler maximization, the log-likelihood function is found to replace products with sums. To find the log-likelihood function we simply take the natural log of our previously calculated total probability function. This is shown in Eqs. (48) and (49) where L is the log-likelihood function.

$$L(o_1, h_1) = \sum_k \sum_z d_{k1}(z) \ln[i_{k1}(z, o_1, h_1)] - i_{k1}(z, o_1, h_1) \quad (48)$$
$$- \ln[d_{k1}(z)!]$$

$$L(o_2, h_2) = \sum_k \sum_w d_{k2}(z) \ln[i_{k2}(w, o_2, h_2)] - i_{k2}(w, o_2, h_2) \quad (49)$$
$$- \ln[d_{k2}(w)!]$$

The final term of the log-likelihood function, the factorial of the incomplete data, does not impact the maximization step because it is not dependent on the object or the PSF. Therefore, in Eqs. (50) and (51) the term is removed and will be disregarded in the remaining steps.

$$L(o_1, h_1) = \sum_k \sum_z d_{k1}(z) \ln[i_{k1}(z, o_1, h_1)] - i_{k1}(z, o_1, h_1) \quad (50)$$

$$L(o_2, h_2) = \sum_k \sum_w d_{k2}(w) \ln[i_{k2}(w, o_2, h_2)] - i_{k2}(w, o_2, h_2) \quad (51)$$

Repeating the procedure used to derive Eq. (50) and Eq. (51) using the complete data yields the results shown in Eq. (52) and Eq. (53). L_{1CD} and L_{2CD} are defined as the horizontal and vertical components, respectively, of the log-likelihood function of the complete data.

$$L_{1CD}(o_1, h_1) = \sum_k \sum_z \sum_x \zeta_{k1}(z, x)\{\ln[o_1(x)] + \ln[h_1(z-x)]\} - o_1(x)h_1(z-x) \quad (52)$$

$$L_{2CD}(o_2, h_2) = \sum_k \sum_w \sum_y \zeta_{k2}(w, y)\{\ln[o_2(y)] + \ln[h_2(w-y)]\} - o_1^{sum} o_2(y)h_1(z-x) \quad (53)$$

3.4.4 Conditional Expectation Step

Now, the expected value of the log-likelihood function has been calculated and we need to find the expectation of the complete data log-likelihood function, defined here as $Q_1(o_1,h_1)$ and $Q_2(o_2,h_2)$. This is shown in Equations (54) and (55). We can also put these equations in terms of the complete data as shown in Equations (56) and (57).

$$Q_1(o_1, h_1) = E[L_{1CD}(o_1, h_1)|d_{k1}(z)] \quad (54)$$

$$Q_2(o_2, h_2) = E[L_{2CD}(o_2, h_2)|d_{k2}(w)] \quad (55)$$

$$Q_1(o_1, h_1) = \sum_k \sum_z \sum_x E[\zeta_{k1}(z, x)|d_{k1}(z)]\{\ln[o_1(x)] + \ln[h_1(z-x)]\} - o_1(x)h_1(z-x) \quad (56)$$

(57)

$$Q_2(o_2, h_2) = \sum_k \sum_w \sum_y E[\zeta_{k2}(w,y)|d_{k2}(w)]\{\ln[o_1^{sum}] + \ln[o_2(y)] + \ln[h_2(w-y)]\} - o_1^{sum} o_2(y) h_2(w-y)$$

3.4.5 Maximize Expectation

We now need to maximize the expectation shown in Eq. (56) by taking the derivative with respect to o_1, setting them equal to zero, and then solving for o_1. This is shown in Eq. (58).

$$\frac{dQ_1(o_1, h_1)}{do_1(x_0)} = \sum_k \sum_z \frac{E[\zeta_{k1}(z,x)|d_{k1}(z)]}{o_1(x_0)} - h_1(z - x_0) = 0 \quad (58)$$

Similarly, we will then maximize Eq. (57) with respect to o_2 by setting the derivative of Eq (57) equal to the LaGrange multiplier, l, times the derivative of the constraint function with respect to $o_2(y_o)$ consistent with the method of Lagrange [23]. This is shown in Eq. (59). In this step we focus on maximizing and solving for only one point at a time. We will use x_0 to designate a single point on the x-axis and similarly y_0 will designate a single point on the y-axis.

$$\frac{dQ_2(o_2, h_2)}{do_2(y_0)} = \sum_k \sum_w \frac{E[\zeta_{k2}(w,y)|d_{k2}(w)]}{o_2(y_0)} - o_1^{sum} h_2(w - y_0) = l \frac{d}{do_2(y_0)} \sum_y o_2(y) \quad (59)$$

We know that, by definition, a PSF must sum to one, so h_1 and h_2 will both sum to one in their respective axis. To simplify the maximization step and for this algorithm to work correctly, o_2 must also sum to one. To ensure that this happens, we utilize a Lagrange multiplier which will be expounded upon below in Equations (65) - (67). With these assumptions, we easily solve for o_1 from Eq. (58) as shown below in Eq. (60) where K is defined as the total number of frames. The superscript notation of "new" will signify the updated estimate to be used in the next iteration of the algorithm.

$$o_1^{new}(x_0) = \frac{\sum_k \sum_z \sum_w \sum_y E[\zeta_k(z,w,x,y)|d_k(z,w)]}{K} \tag{60}$$

To ensure that this happens, we utilize a Lagrange multiplier, represented by the variable l. Under the same assumptions employed in solving for o_1, we can now easily solve for o_2, as shown in Equations (61) and (62), yielding our second update equation.

$$\frac{dQ(o_1,o_2,h_1,h_2)}{do_2(y_0)} = \sum_k \sum_z \sum_w \sum_x \frac{E[\zeta_k(z,w,x,y)|d_k(z,w)]}{o_2(y_0)} - (l + o_1^{sum}) \tag{61}$$

$$= 0$$

$$o_2^{new}(y_0) = \frac{\sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)]}{l + o_1^{sum}} \tag{62}$$

To simplify, we can rename the sum of l and o_1^{sum} as L as shown in Eq. (63). Making this substitution gives the update equation for o_2 shown in Eq. (64).

$$L = l + o_1^{sum} \tag{63}$$

$$o_2^{new}(y_0) = \frac{\sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)]}{L} \tag{64}$$

Now, using Eq. (61), we can solve for *L*. This is outlined in Equations (65) - (67)

$$o_2(y)L = \sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)] \tag{65}$$

$$L \sum_y o_2(y) = \sum_y \sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)] \tag{66}$$

$$L = \sum_y \sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)] \tag{67}$$

These update equations for o_1 and o_2, given in Equations (60) and (64), will enable us to calculate improving estimates for the object in the blind deconvolution algorithm at each iteration.

3.5.6 Solve for Expected Value of Complete Data Given Incomplete Data

We must now find the expectation of the complete data given the incomplete data. The first step toward doing this is find the probability of the complete data given the incomplete data, which will be shown to be distributed binomially. Therefore, we know

that the expected value will be equal to, by definition, *Np,* with *N* being the number of trials and *p* being the probability success of any given trial. In Eq. (68), using Bayes Theorem, the probability of the complete data given the incomplete data is given, where *P* is the probability notation.

$$P\left(\zeta_{k1}(z,x)\middle|d_{k1}(z)\right) = \frac{P\left(\zeta_{k1}(z,x) \cap d_{k1}(z)\right)}{P\left(d_{k1}(z)\right)} \tag{68}$$

$$P\left(\zeta_{k2}(w,y)\middle|d_{k2}(w)\right) = \frac{P\left(\zeta_{k2}(w,y) \cap d_{k2}(w)\right)}{P\left(d_{k2}(w)\right)} \tag{69}$$

We can first solve for the probability of the incomplete data. It is known that the expectation of the incomplete data is the image intensity, or the actual image detected in the receiving plane by the photodetector. We know that photons arrive at a photodetector in a Poisson distribution, so we can express the probability of the incomplete data below in Eq. (70).

$$P[d_{k1}(z)] = \frac{i_{k1}(z,o_1,h_1)^{d_{k1}(z)}e^{-i_{k1}(z,o_1,h_1)}}{d_{k1}(z)!} \tag{70}$$

$$P[d_{k2}(w)] = \frac{i_{k2}(w,o_2,h_2)^{d_{k2}(w)}e^{-i_{k2}(w,o_2,h_2)}}{d_{k2}(w)!} \tag{71}$$

Using the relationship between the complete and incomplete data from Eq. (11), the incomplete data is split into the sum of two variables, d_1 and d_2, to facilitate future calculations. This is shown in Equations (72)-(74). We will solve for the probability of vertical component, P[$d_{k1}(z)$] and apply the form of result to P[$d_{k2}(w)$].

$$d_1 = \zeta_{k1}(z, x_0) \tag{72}$$

$$d_2 = \sum_{x \neq x_0} \zeta_{k1}(z, x_0) + d_{B1}(z) \tag{73}$$

$$d_{k1}(z) = d_1 + d_2 \tag{74}$$

The expected value for d_1 and d_2 are shown below in Equations (75) and (76).

$$E[d_1] = m_1 = o_1(x_0)h_1(z - x_0) + B_1 \tag{75}$$

$$E[d_2] = m_2 = \sum_{x \neq x_0} o_1(x_0)h_1(z - x_0) + B_2 \tag{76}$$

Because d_1 and d_2 are statistically independent we can express the probability of the intersection of d_1 and d_2 as the product of their respective probabilities as shown in Eq. (77). In Eq. (78), it can then be expressed in terms of d_{k1} and d_1 due to their relationship given in Eq. (74).

$$P(d_1 \cap d_2) = \frac{{m_1}^{d_1} e^{-m_1}}{d_1!} \frac{{m_2}^{d_2} e^{-m_2}}{d_2!} \tag{77}$$

$$P(d_1 \cap d_{k1} - d_1) = P(d_1 \cap d_k) = \frac{{m_1}^{d_1} e^{-m_1}}{d_1!} \frac{{m_2}^{d_k - d_1} e^{-m_2}}{(d_{k1} - d_1)!} \tag{78}$$

The probability of the incomplete data, from Eq. (70), can now be written in terms of m_1 and m_2 yielding Eq. (44).

$$P[d_k(z,w)] = \frac{(m_1 + m_2)^{d_k(z,w)} e^{-(m_1+m_2)}}{d_k(z,w)!} \tag{79}$$

Finally, we solve for the conditional probability, shown in Eq. (80), which is clearly a binomial distribution.

$$\frac{P(d_1 \cap d_{k1})}{P(d_{k1})} = \frac{d_{k1}!}{d_1(d_{k1} - d_1)} \left(\frac{m_1}{m_1 + m_2}\right)^{d_1} \left(\frac{m_2}{m_1 + m_2}\right)^{d_{k1} - d_1} \tag{80}$$

In order to solve for the expectation of the complete data given the incomplete data, N and p are defined in Equations (81) and (82), respectively.

$$N = d_{k1}(z) \tag{81}$$

$$p = \frac{m_1}{m_1 + m_2} \tag{82}$$

Now that it has been shown that the probability of the complete data given the incomplete data is distributed binomially, its expectation can be expressed in terms of o_1, h_1, and d_{k1}, as shown in Eq. (83).

$$E[\zeta_{k1}(z,x)|d_{k1}(z)] = Np = d_{k1}(z)\frac{o_1(x_0)h_1(z-x_0)}{\sum_x[o_1(x)h_1(z-x)] + B_1} \tag{83}$$

We can also use the form of this result to produce the conditional probability of horizontal component expressed in terms of o_2, h_2, and $d_{k2,}$ as shown below in Eq. (84).

$$E[\zeta_{k2}(w,y)|d_{k2}(w)] = Np = d_{k2}(w)\frac{o_2(y_0)h_2(w-y_0)}{\sum_y[o_2(y)h_2(w-y)] + B_2} \tag{84}$$

3.5.7 Solve for Update Equations

The final step to arrive at an update equation for o_1 is to take Eq. (60) and replace the expectation of the complete data given the incomplete data with the result from Eq. (83) substituting old values for the estimates of o_1 since the expectation is conditional on the incomplete data and old estimates from the previous iteration. This produces the update equation that will be used in the blind deconvolution algorithm and is shown in Eq. (85). Similarly, for o_2 we take Eq. (64) and again replace the expectation of the complete data given the incomplete data with the result from Eq. (83) doing the same substitution using old estimates from the previous iteration. This yields the update equation for o_2 and is shown in Eq. (86).

$$o_1^{new}(x_0) = \frac{\sum_k \sum_z \frac{d_{k1}(z) o_1^{old}(x_o) h_1(z - x_0)}{\sum_x [o_1^{old}(x) h_1(z - x)] + B_1}}{K} \tag{85}$$

$$o_2^{new}(y_0) = \frac{\sum_k \sum_w \frac{d_{k2}(w) o_2^{old}(y_o) h_2(w - y_0)}{\sum_y [o_2^{old}(y) h_2(w - y)] + B_2}}{L} \tag{86}$$

3.5.8 Solve for the Seeing Parameter

To solve for the correct PSF, the algorithm is run multiple times using various r_0 values each one yielding a different PSF. A log-likelihood is calculated for each solution using Eq. (50). The r_0 value corresponding with the first peak value is chosen to be the estimated r_0. This approach produces an r_0 estimate within one centimeter of the correct value in all cases tested thus far.

IV. Results

This Chapter outlines the results achieved when applying the described one-dimensional algorithm to both simulated and laboratory-generated data.

4.1 Trials using Simulated Data

This section explores the two identified performance metrics. The simulated object used will be a binary star system. The OTF and the long-exposure transfer function are simulated using Equations (8) and (10), respectively. The total PSF is found by taking

the inverse Fourier transform of the two OTFs multiplied together. In this case, a high signal to noise ratio (SNR) is assumed and simulated, mimicking imaging of distant stars.

The true seeing parameters used in these trials will be 14cm, 12cm, and 10cm. Because the same method of solving for r_0 is employed in both the two-dimensional and one-dimensional algorithm, the true r_0 value is given in the trials producing the results below.

The PSF is generated as described in the paragraph above and in section 2.2. The specifications of the computer performing the calculations are given for reference below in Table 1.

Table 1: Computer Specifications

Specification	**Value**
Central Processing Unit (CPU)	Intel® Core™ i7-7700HQ
CPU Base Frequency	2.8 GHz
Motherboard	Intel® HM175
Installed Random Access Memory (RAM)	12GB
Graphics Processing Unit	NVIDIA® GeForce® GTX 1050 Ti
GPU Dedicated Memory	4GB
Hard Drive	Intel 660p Series M.2 2280
Hard Drive Capacity	1TB
Hard Drive Format	Solid State Drive

4.1.1 Equal Intensity Binary Star System

The binary system is modeled using two pixels of equal intensity separated by ten pixels as shown in Fig. 19. In all results going forward the PSF is generated as described in the paragraph above and in section 2.2. An example is shown below in Fig. 20.

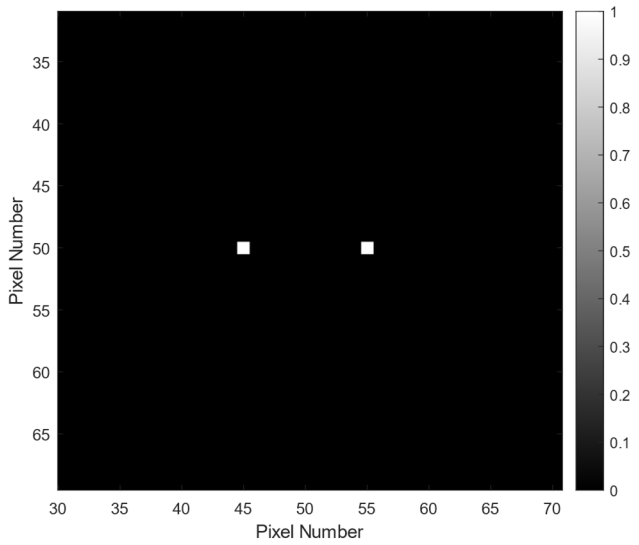

Fig. 19. Binary Star System Model

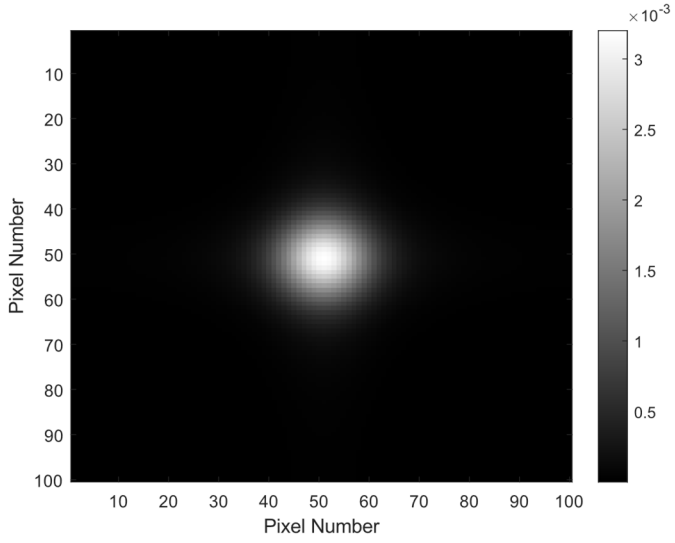

Fig. 20. Simulated PSF

To achieve an image to be deconvolved, the binary star system is convolved with the generated PSF. An example of generated data with Poisson noise included, is shown below in Fig. 21.

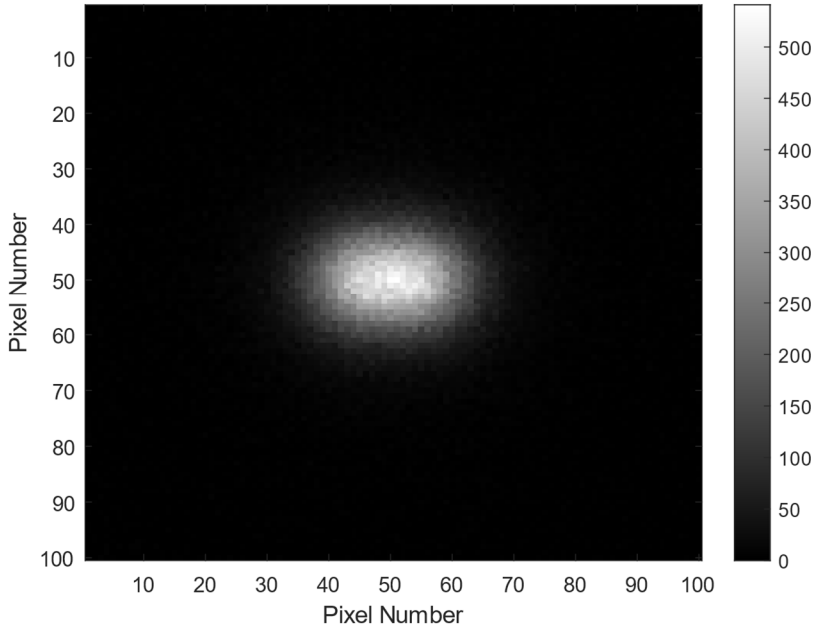

Fig. 21. Simulated data

Running the frame of simulated data through the two-dimensional blind deconvolution algorithm generates an estimate for the object. In these trials we run 200,000 iterations of each algorithm to produce an object estimate. This is done three times using three different r_0 values. These results will be compared to the true object and PSFs and then used for future comparison to be discussed in the following sections. The object estimate achieved using this two-dimensional algorithm with an r_0 value of 14cm is magnified for detail and is shown in Fig. 22 and the result when using an r_0 value of 10cm is shown in Fig. 23.

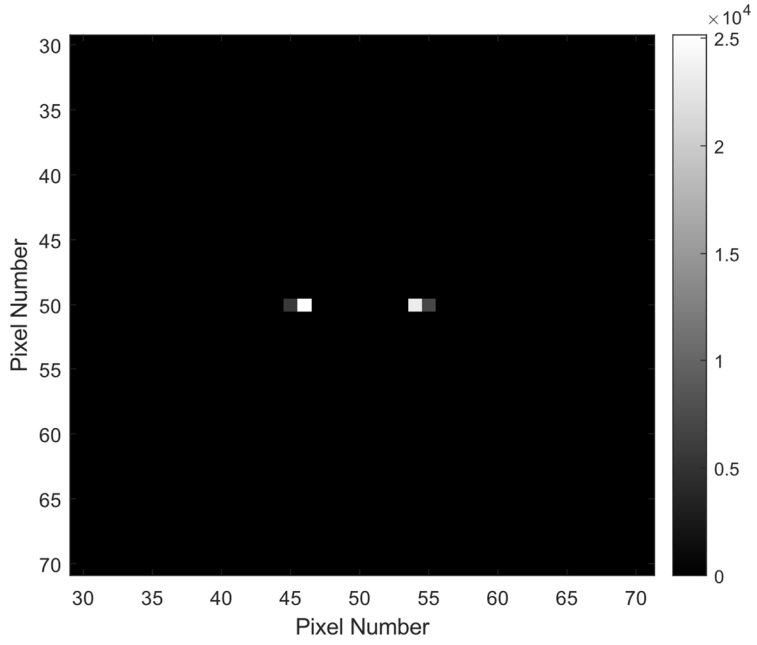

Fig. 22. Two-Dimensional Object Deconvolution Result (r_0=14cm)

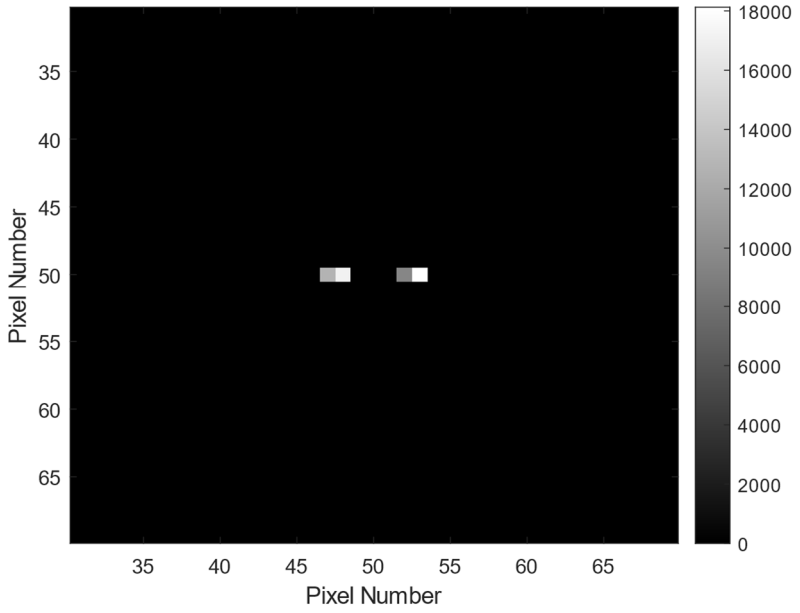

Fig. 23. Two-Dimensional Object Deconvolution Result (r_0=10cm)

Running the data through the one-dimensional blind deconvolution algorithm generates estimates for the object and the PSF. Once again, the object estimates achieved using this one-dimensional algorithm with r_0 values of 14cm and 10cm are magnified for detail and are shown in Fig. 24 and Fig. 25.

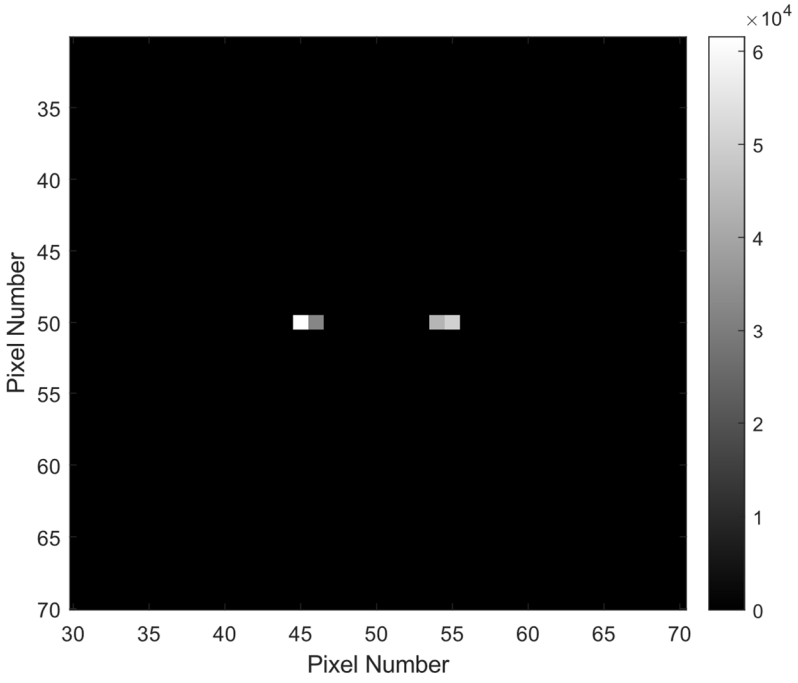

Fig. 24. One-Dimensional Object Deconvolution Result (r_0=14cm)

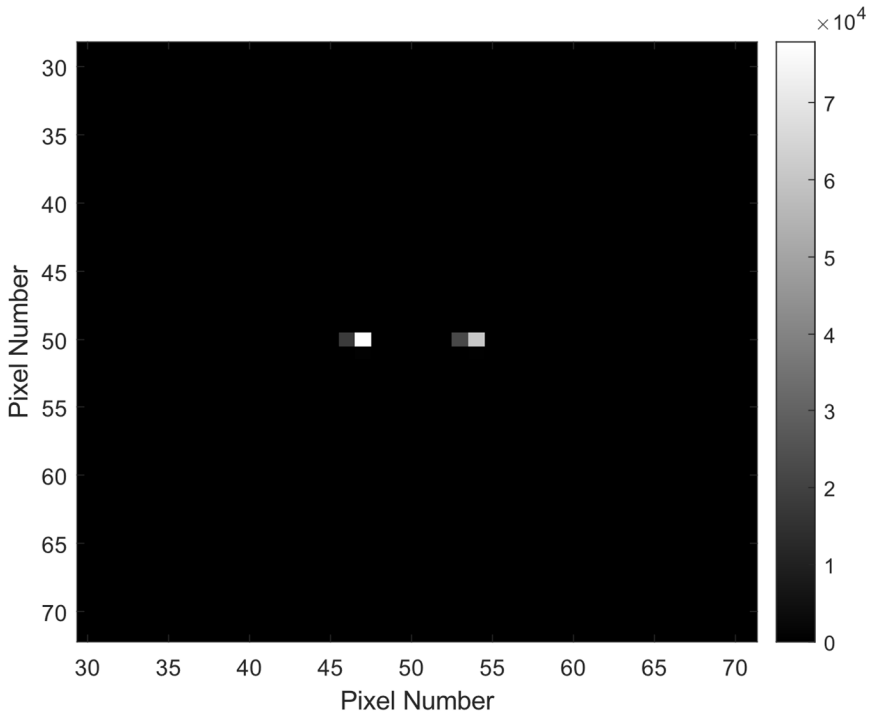

Fig. 25. One-Dimensional Object Deconvolution Result (r_0=10cm)

Upon inspection, we see that both algorithms generated relatively accurate results in comparison to the true object, especially with r_0 values of 14cm. The intensity of the stars in the binary system are not exactly equal as was the case for the true object, but a basic reconstruction appears to be successful in both cases. In the following sub-sections, we will compare the accuracy and speed of the two algorithms.

4.1.1.1 Speed Comparison

To measure how quickly each algorithm can perform the required deconvolution, we simply use MATLAB's built-in functions to record the time needed to complete each trial. Fig. 26Fig. 26. Time Required Comparison shows the time required to complete

200,000 iterations for each trial and Table 2 shows the average results of our mean-squared error calculation.

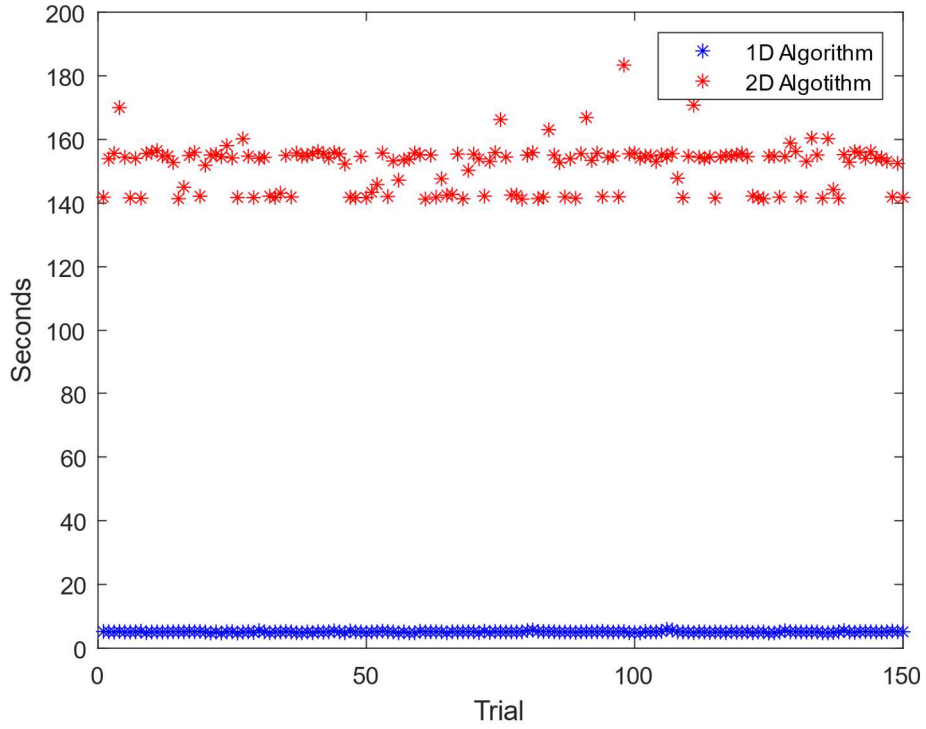

Fig. 26. Time Required Comparison (All r_0 Values)

Table 2. Average Time Required

Algorithm Used	**Average Time Required (Seconds)**
One-Dimensional	5.1183
Two-Dimensional	151.53

We observe that, on average, the one-dimensional algorithm performs the blind deconvolution approximately 30 times faster than the two-dimensional algorithm.

4.1.1.2 Accuracy Comparison

We will use three separate metrics to compare the accuracy of the algorithms. Using the mean-squared error technique, we can compare the total accuracy at each pixel of the two-dimensional algorithm with that of the one-dimensional algorithm. The MSE is calculated using Eq. (87), where m is the number of pixels on the x-axis, n is the number of pixels on the y-axis, $T(x,y)$ is the true intensity value of the object/PSF at pixel (x,y), and $o_1(y)$ and $o_2(x)$ are the estimated values of the spatially separated object at pixels x and y.

$$MSE = \frac{1}{n}\sum_{x=1}^{m}\sum_{y=1}^{n}[T(x,y) - o_1(y)o_2(x)]^2 \tag{87}$$

We will also use a more targeted approach in measuring accuracy with two other calculations. First, we calculate the estimated brightness ratio and its associated error. The estimated brightness ratio is found simply by dividing the estimated intensity value of the bright source, on the left, by the estimated intensity value of the dim source, on the right. The brightness ratio error, $E_{br,}$ was calculated using Eq. (88) where β_{true} is the true brightness ratio, β_{ave} is the average estimated brightness ratio.

$$E_{br} = \frac{|\beta_{true} - \beta_{ave}|}{\beta_{true}} \tag{88}$$

The pixel spacing was calculated by find the pixel separation between the two brightest points on each side of the estimate. The spacing error between the two sources was calculated using Eq. (89) where σ_{true} is the true pixel spacing between sources and σ_{ave} is the average estimated pixel spacing between sources.

$$E_{sp} = \frac{|\sigma_{true} - \sigma_{ave}|}{\sigma_{true}} \tag{89}$$

Each algorithm performed the deconvolution fifty times at each value of r_0 with Poisson noise randomly generated for each trial. 200,000 iterations of the algorithm were performed in each trial.

4.1.1.2.1 Mean-Squared Error

Fig. 27, Fig. 28 and Fig. 29 show the calculated MSE for each trial with r_0 values of 14cm, 10cm, and 6cm, respectively.

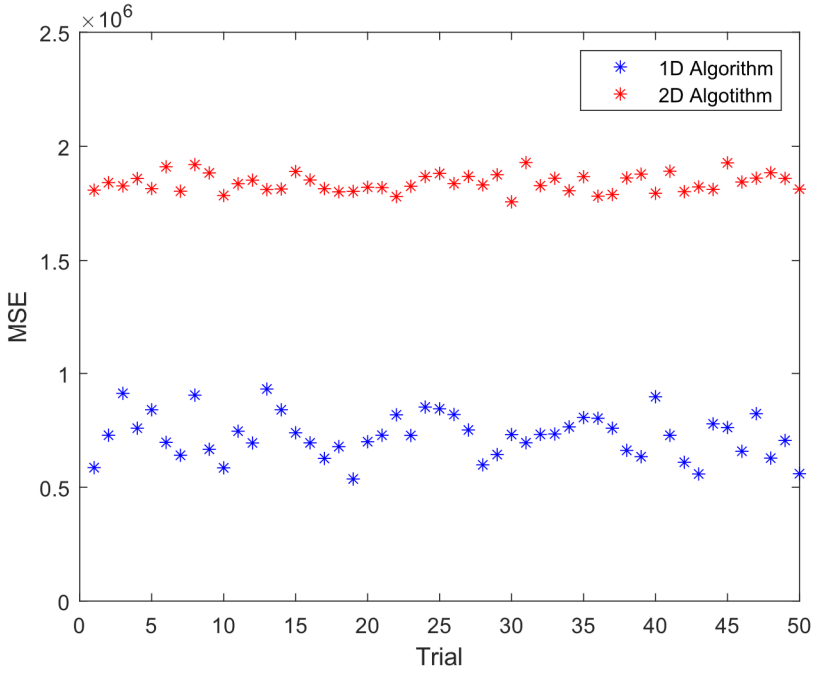

Fig. 27. Mean-Squared Error Comparison (r_0=14cm)

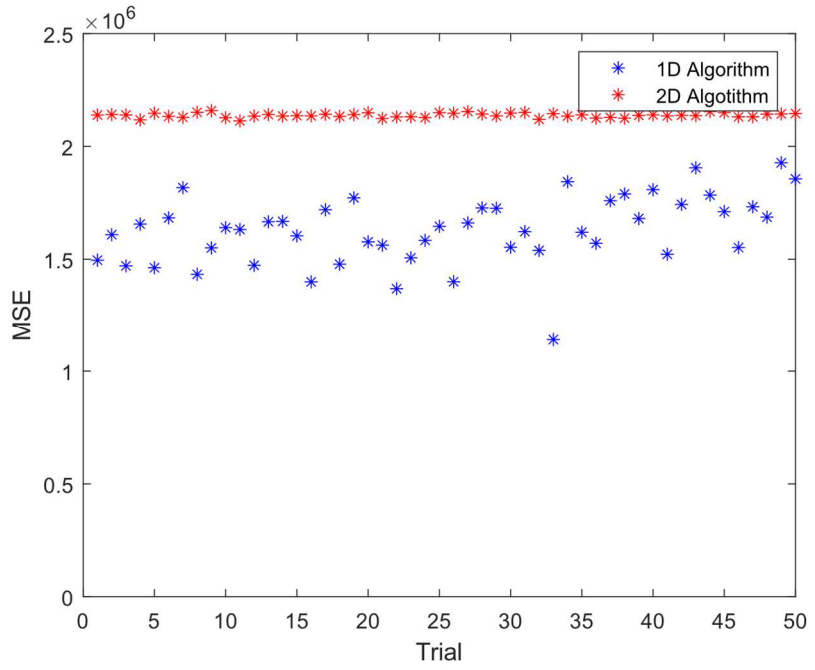

Fig. 28. Mean-Squared Error Comparison (r_0=12cm)

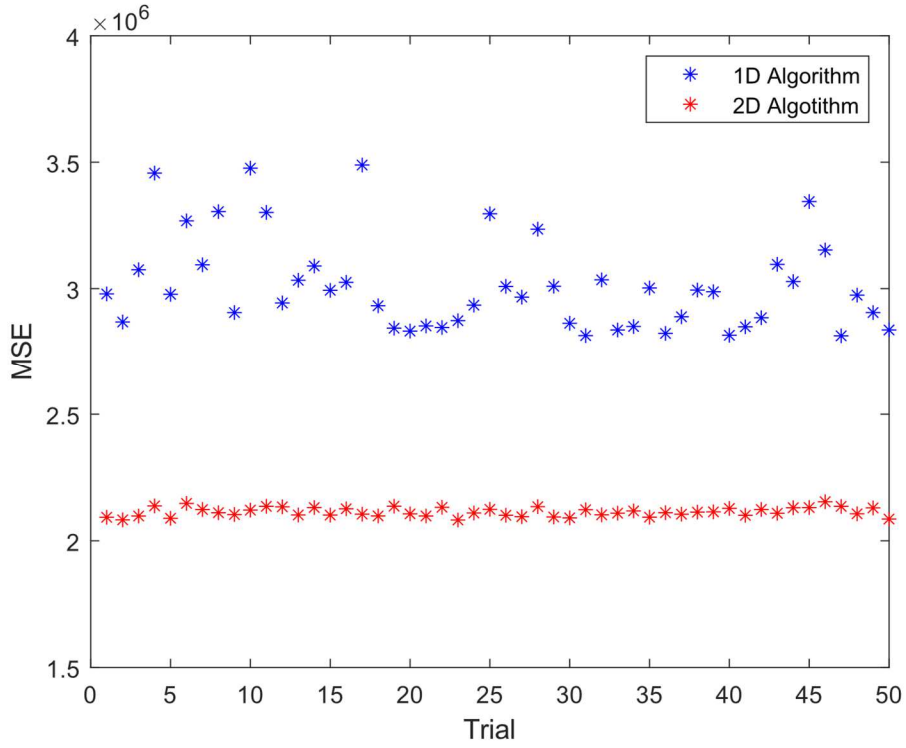

Fig. 29. Mean-Squared Error Comparison (r_0=10cm)

We observe that the one-dimensional algorithm achieves a lower MSE at higher r_0 values, but that the two-dimensional algorithm achieves a lower MSE at the lowest r_0 value of 10cm.

4.1.1.2.2 Brightness Ratio

The estimated brightness ratio of each trial is shown for r_0 values of 14cm, 10cm, and 6cm in Fig. 30, Fig. 31, and Fig. 32, respectively. The true brightness ratio value, in this case one, is also shown.

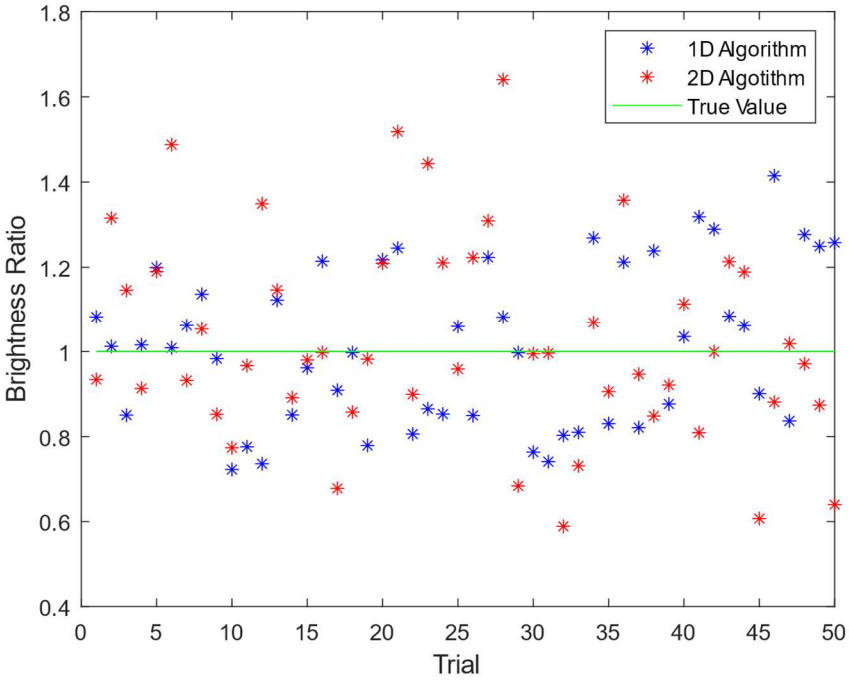

Fig. 30. Estimated Brightness Ratio (r_0=14cm)

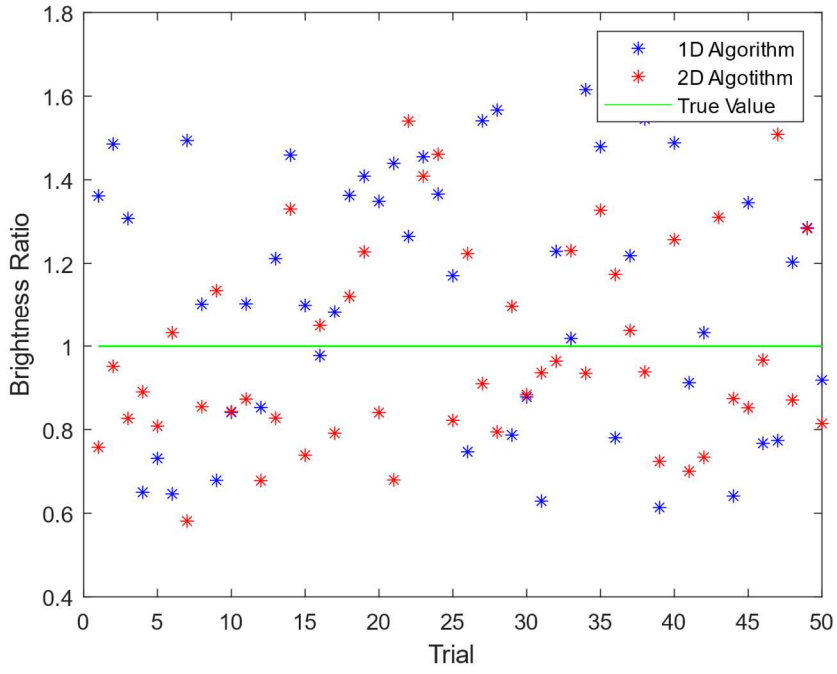

Fig. 31. Estimated Brightness Ratio (r_0=12cm)

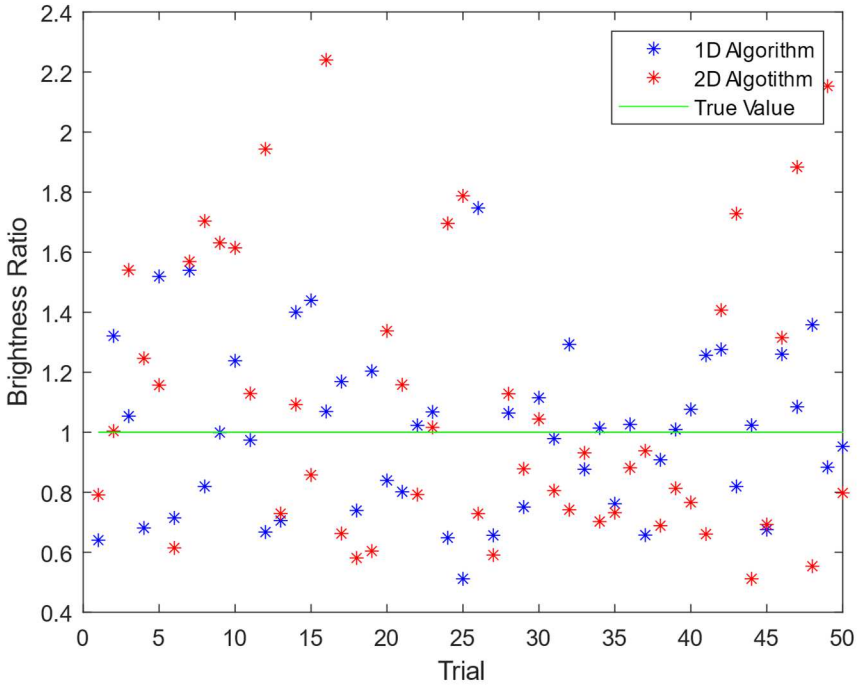

Fig. 32. Estimated Brightness Ratio (r_0=10cm)

For all three r_0 values both algorithms appear to produce similar brightness ratio estimates, with the two-dimensional algorithm appearing to have slightly more variance and a higher tendency to overestimate the ratio.

4.1.1.2.3 Source Pixel Spacing

The number of pixels estimated between the two sources is shown for r_0 values of 14cm, 10cm, and 6cm in Fig. 33, Fig. 34, and Fig. 35, respectively.

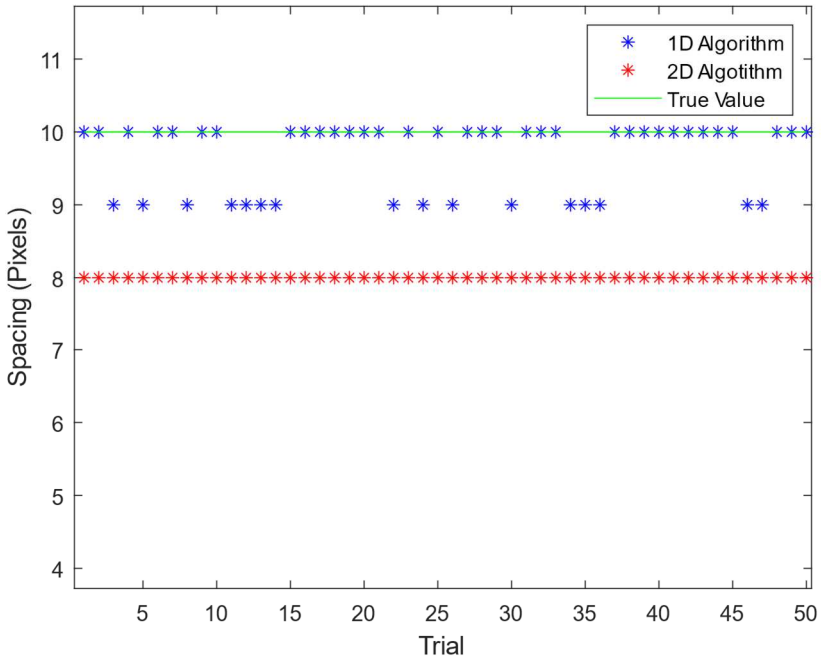

Fig. 33. Estimated Pixel Spacing (r_0=14cm)

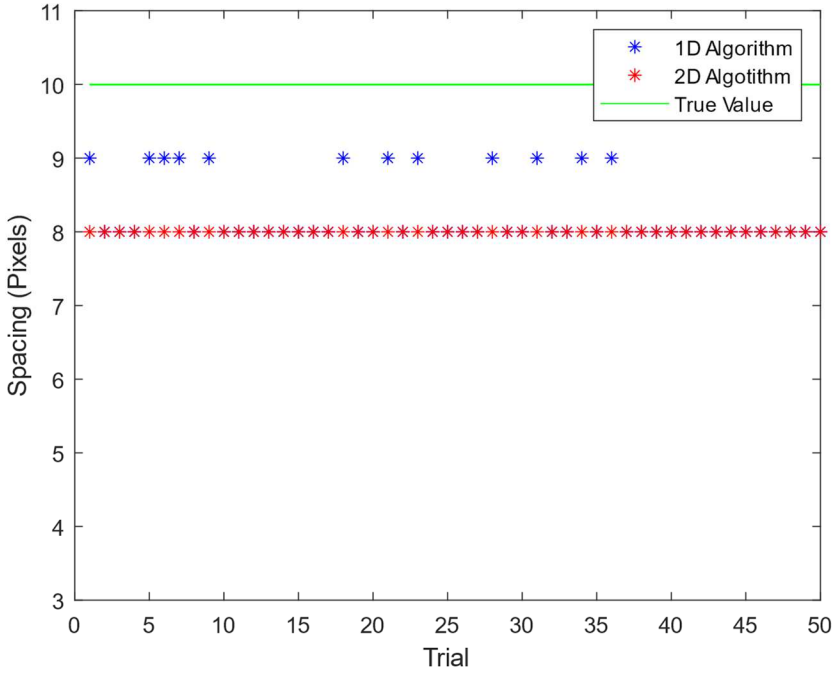

Fig. 34. Estimated Pixel Spacing (r_0=12cm)

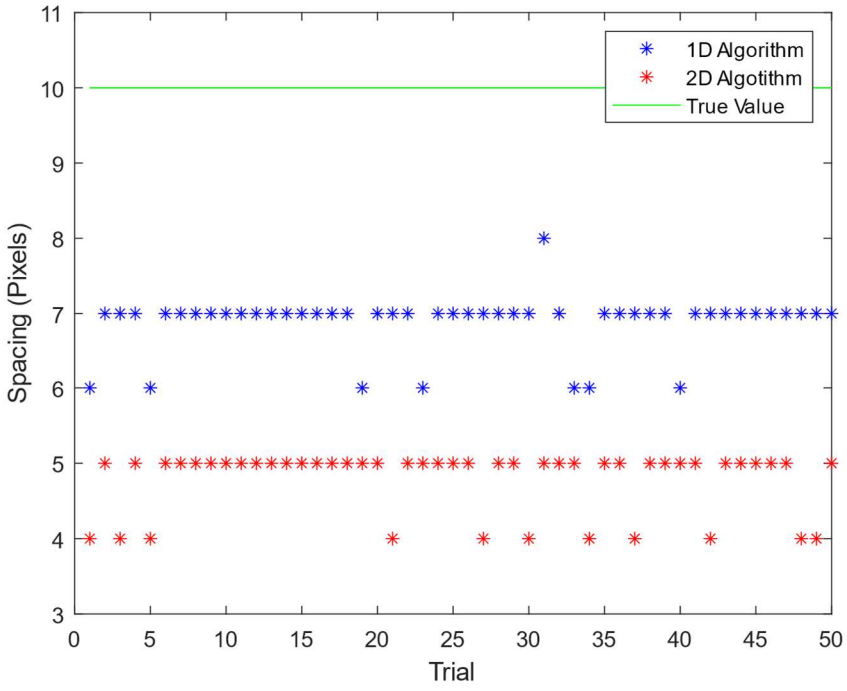

Fig. 35. Estimated Pixel Spacing (r_0=10cm)

We observe that the one-dimensional algorithm estimates the pixel spacing between sources more accurately in almost every case, though performance decreases as r_0 decreases for both.

4.1.1.2.4 Average Error Values

The averages values for MSE, brightness ratio and pixel spacing for each value of r_0 are all shown together in Table 3, Table 4, and Table 5, below.

Table 3. Accuracy (r_0=14cm)

Algorithm Used	Average MSE	Average Brightness Ratio Error	Average Spacing Error
1-D	7.27×10^5	1.4%	3.2%
2-D	1.84×10^6	2.4%	20%

Table 4. Accuracy (r_0=12cm)

Algorithm Used	Average MSE	Average Brightness Ratio Error	Average Spacing Error
1-D	1.63×10^6	1.3%	10.6%
2-D	2.14×10^6	1.2%	20%

Table 5. Accuracy (r0=10cm)

Algorithm Used	Average MSE	Average Brightness Ratio Error	Average Spacing Error
1-D	3.01×10^6	0.6%	31.2%
2-D	2.11×10^6	9.1%	52%

We see that when r_0 is set to 14cm and 12cm, the one-dimensional algorithm produces a lower MSE error. However, when the r_0 value of 10cm is used the two-dimensional algorithm produces a lower MSE.

Both algorithms produce relatively accurate approximations of the brightness ratio at r_0 values of 14cm and 12cm, with the one-dimensional algorithm having a slightly lower average error in both cases. With an r_0 value of 10cm, the one-dimensional algorithm maintains a low average error, while the two-dimensional algorithm's error increases to nearly 10%.

We see from this data that the one-dimensional algorithm clearly estimates the spacing between the two sources better than the two-dimensional algorithm overall.

It may seem counterintuitive that when r_0 is set to 10cm the one-dimensional algorithm produces a higher MSE error while at the same time producing significantly lower spacing and brightness errors. This is likely due to the way that MSE error is calculated and provides an example of some of the issues with MSE. Because it involves summing the square of each pixel value, a recreated image that estimated two distinct, bright point sources whose location was off by only one pixel could have a higher MSE that than two dimmer, slightly blurred source in the same location. MSE error does not typically correspond well with human visual perception.[33] We see in the following sections that this pattern continues when altering the true brightness ratio.

4.1.2 Dim Star in Proximity to Bright Star

The binary system is modeled using two pixels separated by ten pixels just like in section 4.1.1 Equal Intensity Binary Star System but in this case we decrease the intensity of the pixel on the right. This is shown in Fig. 36 with a brightness ratio of 4:1 and in Fig. 38 with a brightness ratio of 10:1. To achieve an image to be deconvolved, the binary star system is once again convolved with the generated PSF. An example of generated data with a brightness ratio of 4:1 and Poisson noise included, is shown below in Fig. 37. Fig. 39 generated data with a source brightness ratio of 10:1.

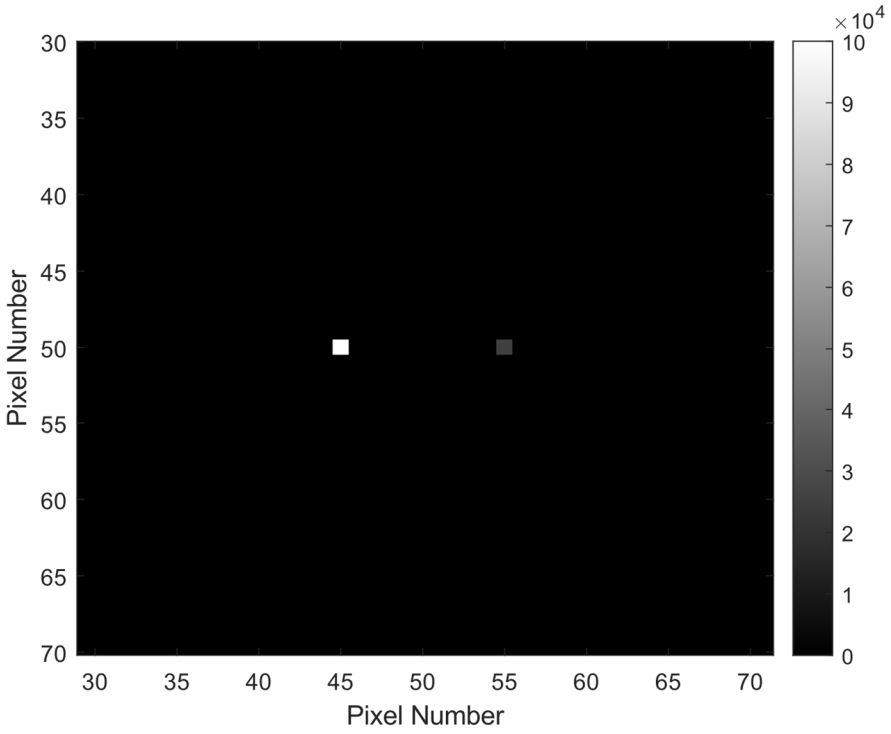

Fig. 36. Dim Star Object (4:1 Source Brightness Ratio)

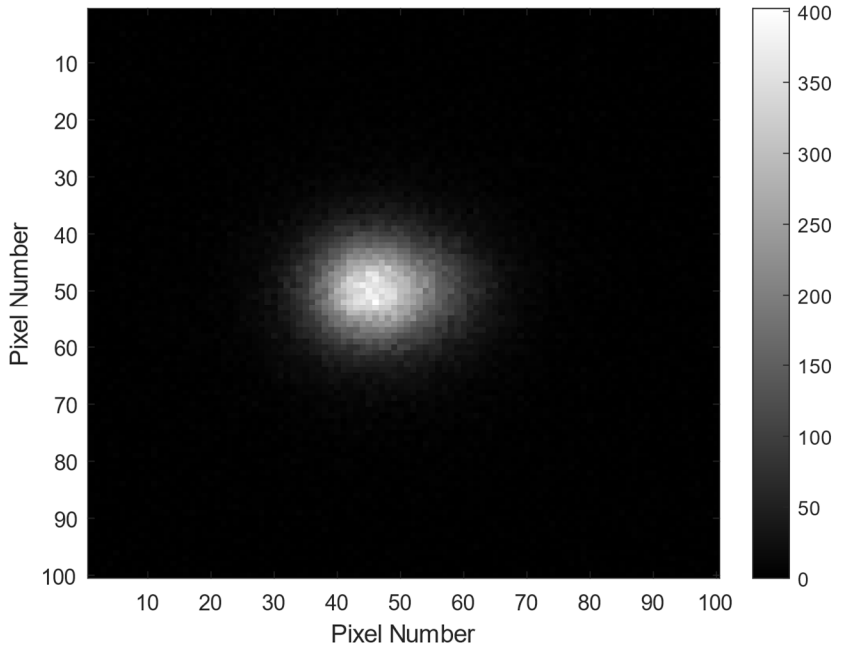

Fig. 37. Dim Star Simulated Data (4:1 Source Brightness Ratio)

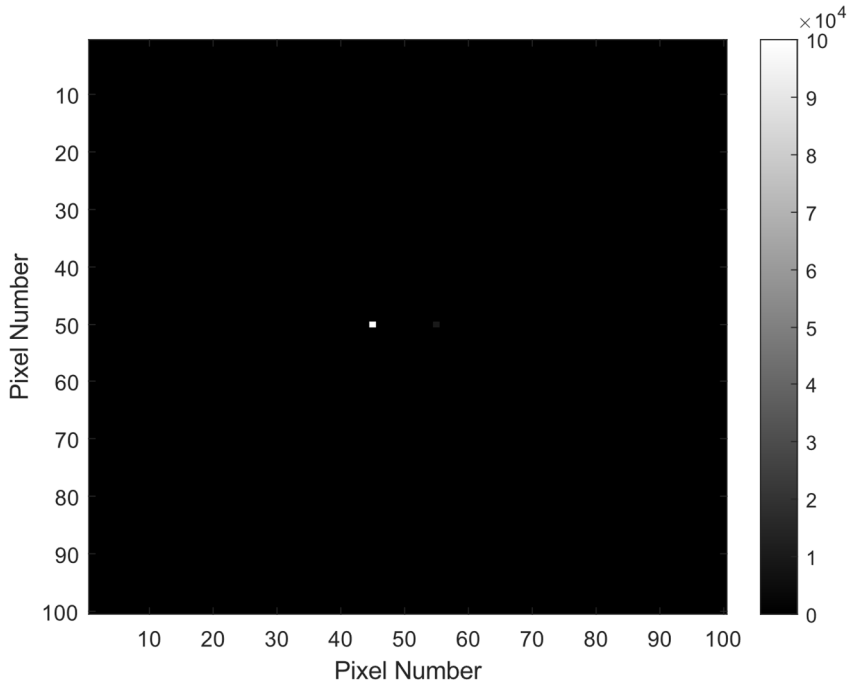

Fig. 38. Dim Star Object (10:1 Source Brightness Ratio)

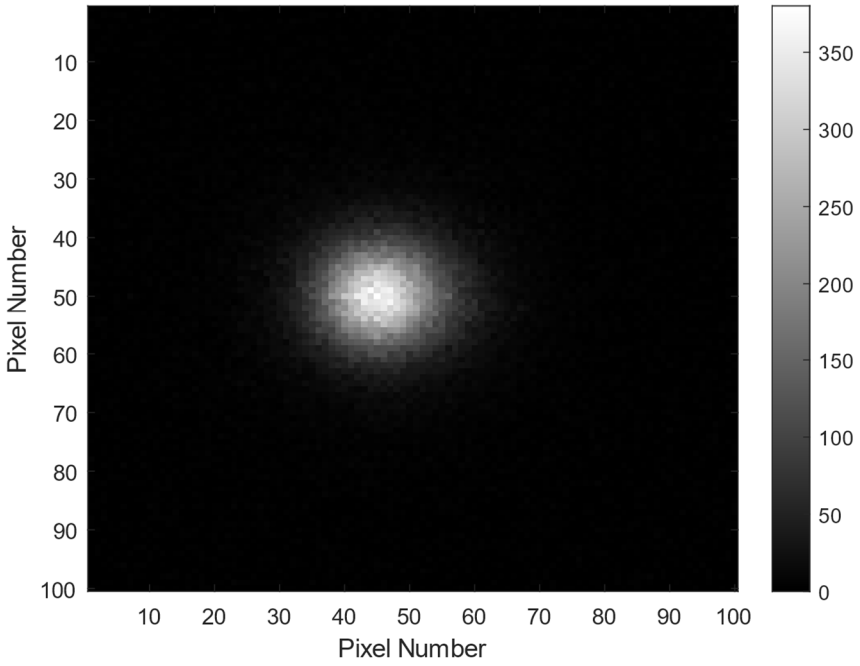

Fig. 39. Dim Star Simulated Data (10:1 Source Brightness Ratio)

Using our two different algorithms we once again process the simulated data. In these trials we continue to run 200,000 iterations of each algorithm to produce an object estimate. These results will be compared to the true object and PSFs and then used for future comparison to be discussed in the following sections. The object estimate achieved using this two-dimensional algorithm is magnified for detail and is shown in Fig. 22. Running this frame of simulated data through the two-dimensional blind deconvolution algorithm generates an estimate for the object. In these trials we run 200,000 iterations of each algorithm to produce an object estimate. This is done three times using three different r_0 values. These results will be compared to the true object and PSFs and then used for future comparison to be discussed in the following sections. The object estimate achieved using this two-dimensional algorithm with an r_0 value of 14cm is magnified for detail and is shown in Fig. 40 and the result when using an r_0 value of 10cm is shown in Fig. 41.

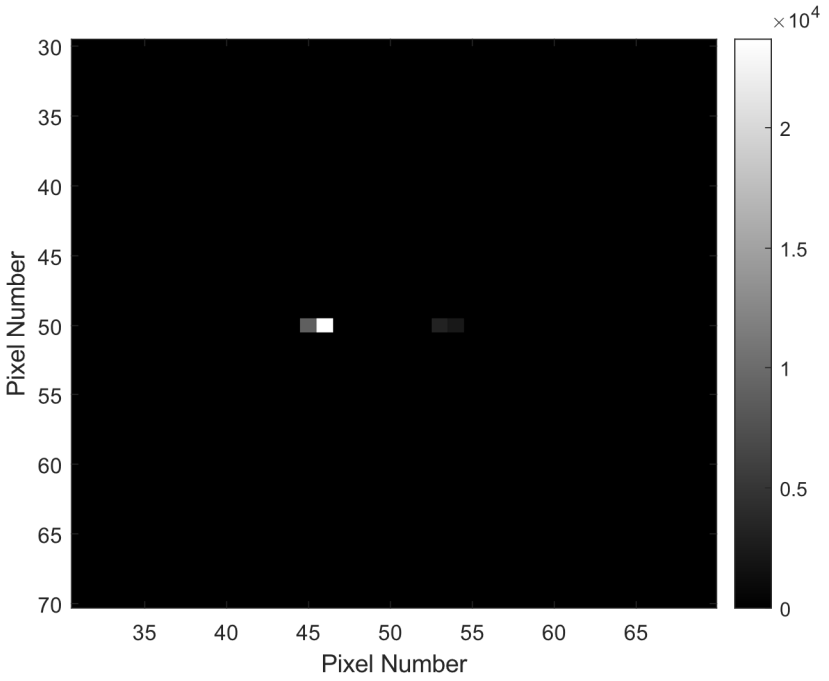

Fig. 40. Two-Dimensional Object Deconvolution Result (r_0=14cm)

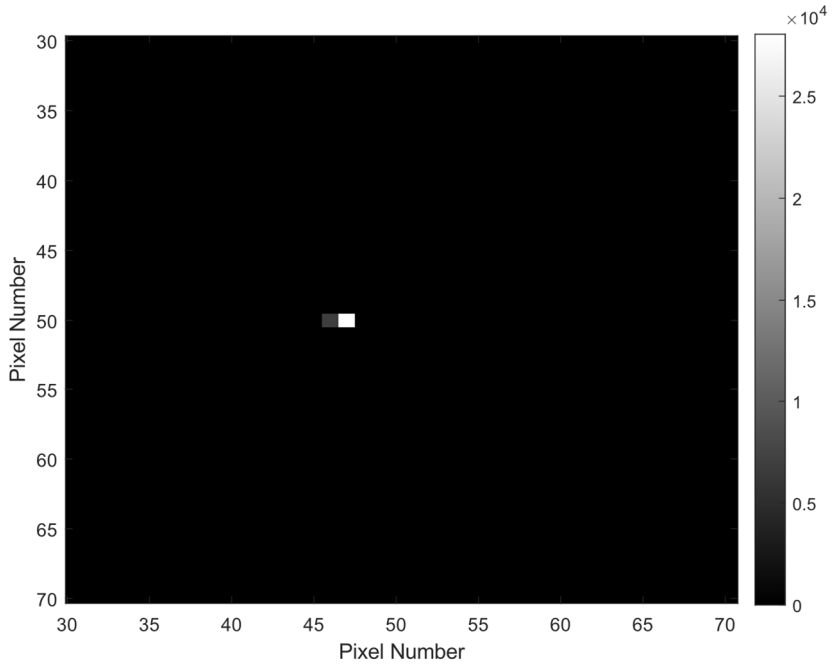

Fig. 41. Two-Dimensional Object Deconvolution Result (r_0=10cm)

Running the data through the one-dimensional blind deconvolution algorithm generates estimates for the object and the PSF. Once again, the object estimates achieved using this one-dimensional algorithm with r_0 values of 14cm and 10cm are magnified for detail and are shown in Fig. 42 and Fig. 43.

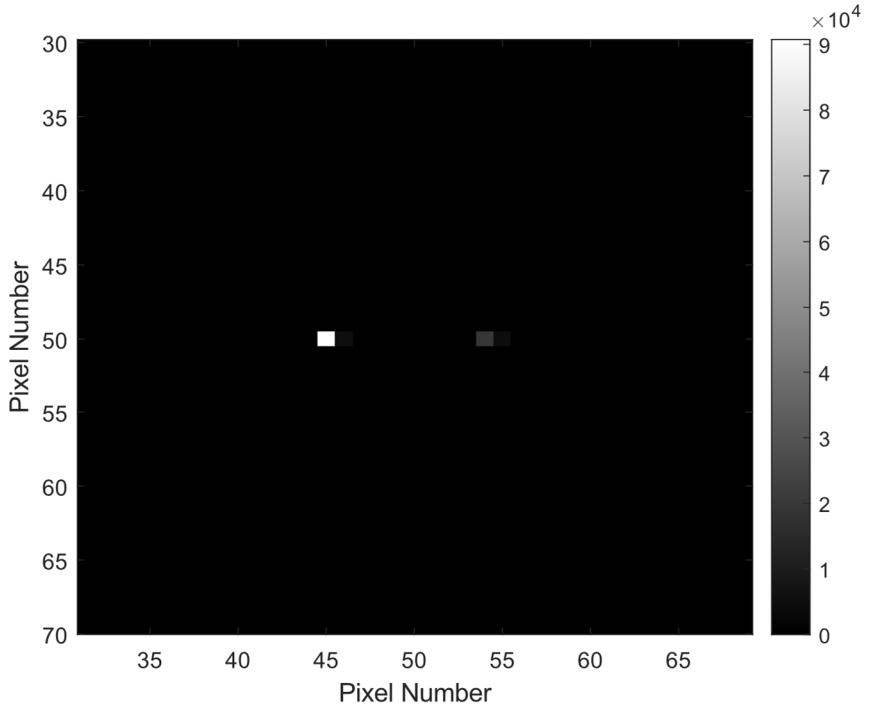

Fig. 42. One-Dimensional Deconvolution Result (r_0=14cm, 4:1 Brightness Ratio)

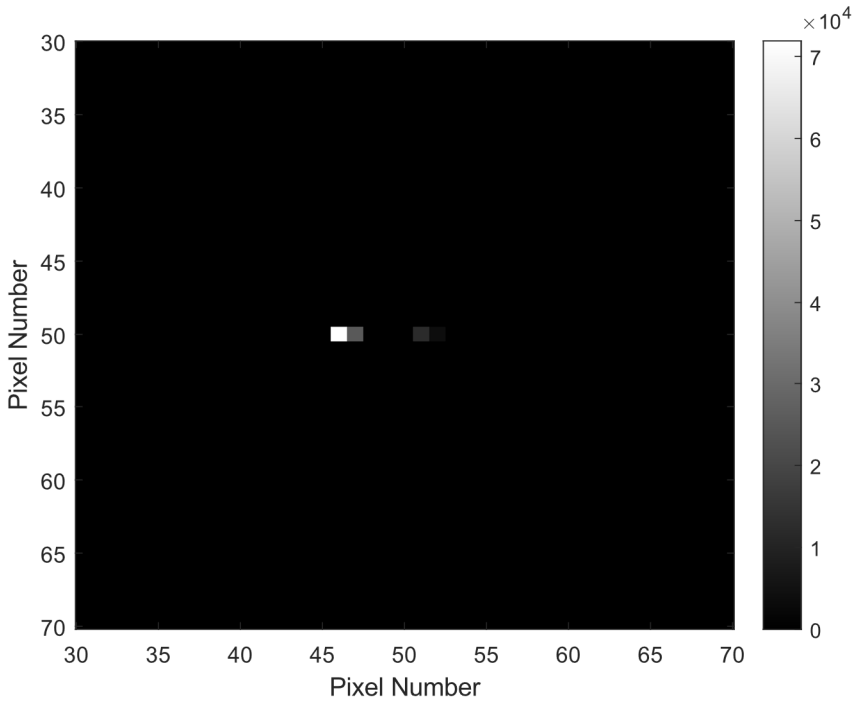

Fig. 43. One-Dimensional Deconvolution Result (r_0=10cm, 4:1 Brightness Ratio)

Upon inspection, we see that both algorithms generated relatively accurate results in comparison to the true object when using r_0 values of 14cm. The intensity of the stars in the binary system are not exactly equal as was the case for the true object, but a basic reconstruction appears to be successful in both cases. However, we notice that the two-dimensional algorithm was unable to reproduce the second, dim object when an r_0 value of 10cm was used and the one-dimensional algorithm estimate consisted of much more closely spaced sources. In the following sub-sections, we will compare the accuracy and speed of the two algorithms.

4.1.2.1 Speed Comparison

As in the previous section, Fig. 44 shows the time required to complete 200,000 iterations for each trial and Table 6 shows the average results of our MSE calculation.

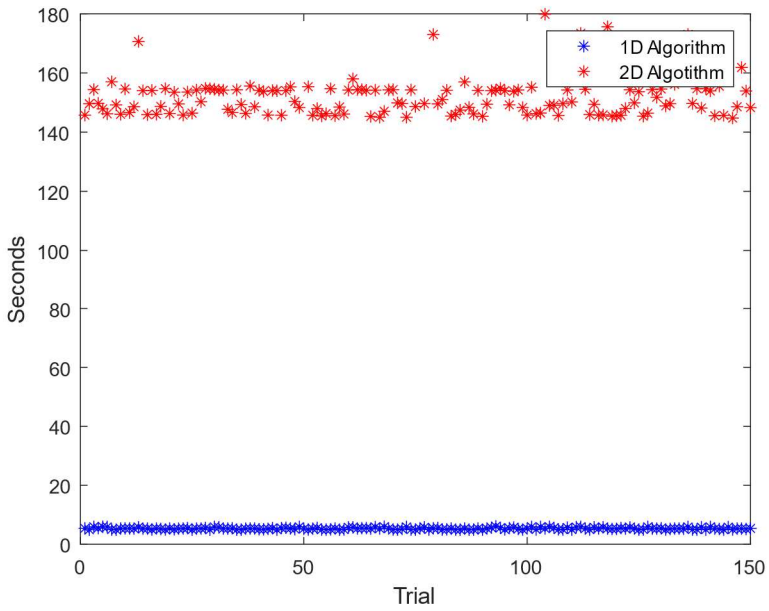

Fig. 44. Time Required Comparison (All r_0 Values)

Table 6. Average Time Required

Algorithm Used	**Average Time Required (Seconds)**
One-Dimensional	5.336
Two-Dimensional	153.3

In this case, we observe that, on average, the one-dimensional algorithm performs the blind deconvolution approximately 28 times faster than the two-dimensional algorithm. Once again, we see that the proposed one-dimensional algorithm is significantly faster than the two-dimensional algorithm.

4.1.2.2 Accuracy Comparison

We use the same accuracy metrics to compare the accuracy of the algorithms in this section including MSE, pixel spacing, and brightness ratio. Once again, each algorithm performed the deconvolution fifty times at each value of r_0, 14cm, 12cm, and 10cm, with Poisson noise randomly generated for each trial. 200,000 iterations of the algorithm were performed in each trial. However, in this section we also vary the brightness ratio between sources. The ratio values of 2:1, 4:1, and 10:1 were all tested at each value of r_0 and their average accuracy comparison results are shown in Table 7 through Table 9. The plots showing individual accuracy test results at only the 4:1 brightness ratio are shown in the following subsections.

4.1.2.2.1 Mean-Squared Error

The MSE plots for a true source brightness ratio of 4:1 are shown below in Fig. 45, Fig. 46, and Fig. 47 with r_0 values of 14cm, 12cm, and 10cm, respectively.

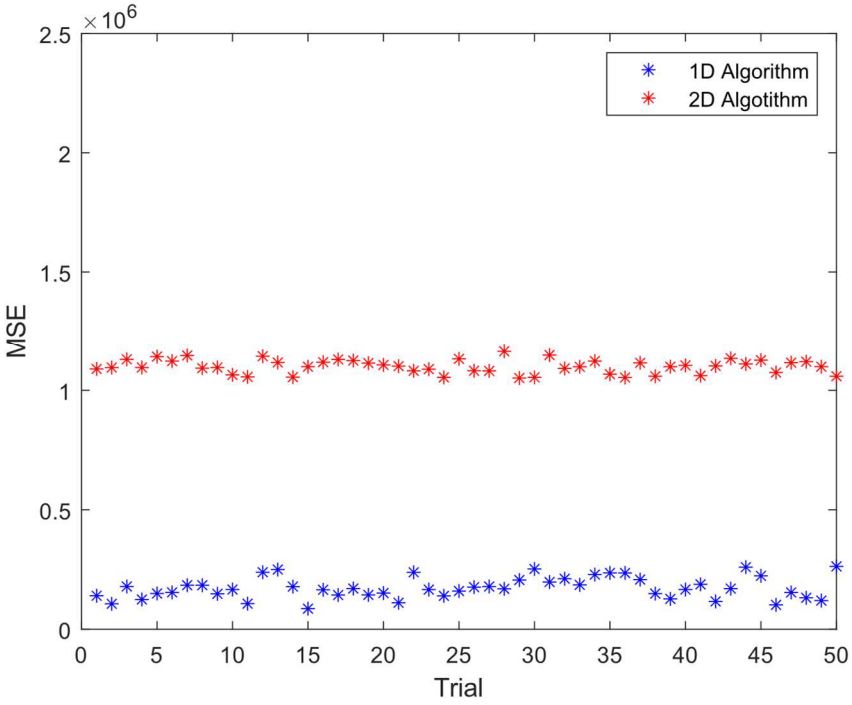

Fig. 45. Mean-Squared Error Comparison (r_0=14cm)

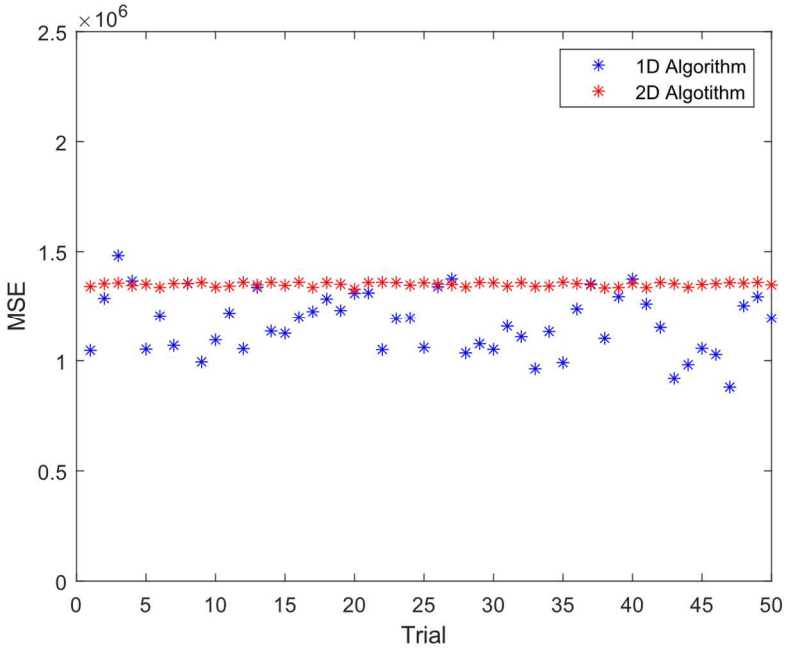

Fig. 46. Mean-Squared Error Comparison (r_0=12cm)

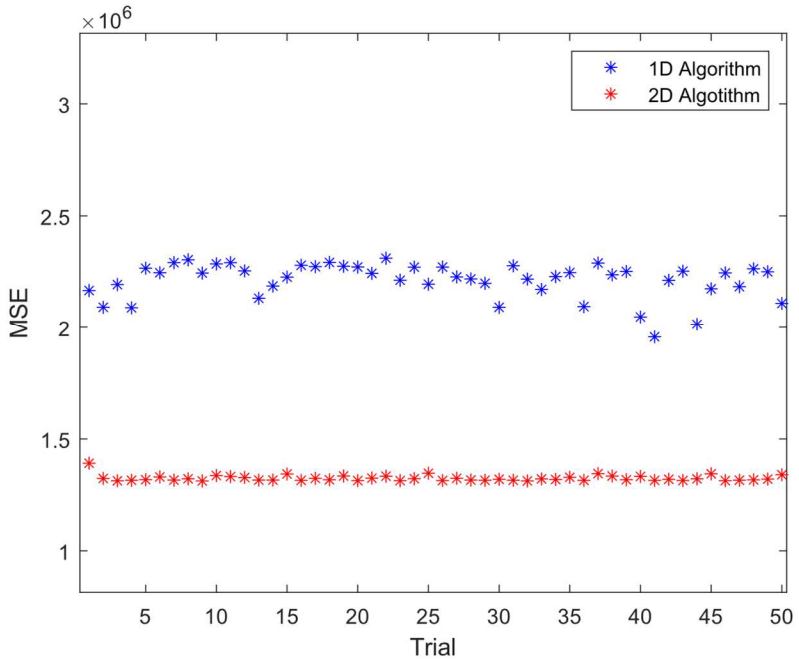

Fig. 47. Mean-Squared Error Comparison (r_0=10cm)

We again observe that the one-dimensional algorithm achieves a lower MSE at higher r_0 values, but that the two-dimensional algorithm achieves a lower MSE at the lowest r_0 value of 10cm much like the 1-1 brightness ratio comparison in the previous section.

4.1.2.2.2 Brightness Ratio

The estimated brightness ratio of each trial is shown for r_0 values of 14cm, 10cm, and 6cm in Fig. 30, Fig. 31, and Fig. 32, respectively.

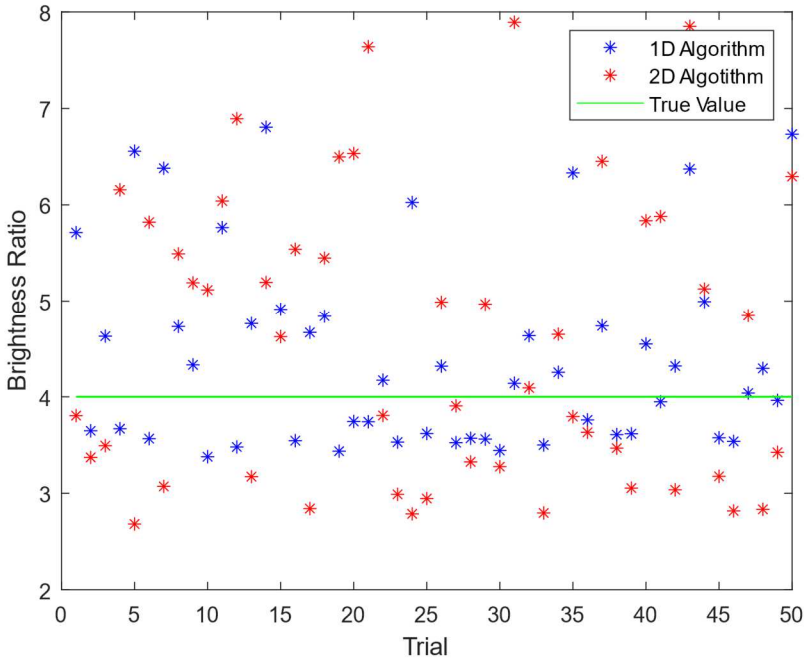

Fig. 48. Estimated Brightness Ratio (r_0=14cm)

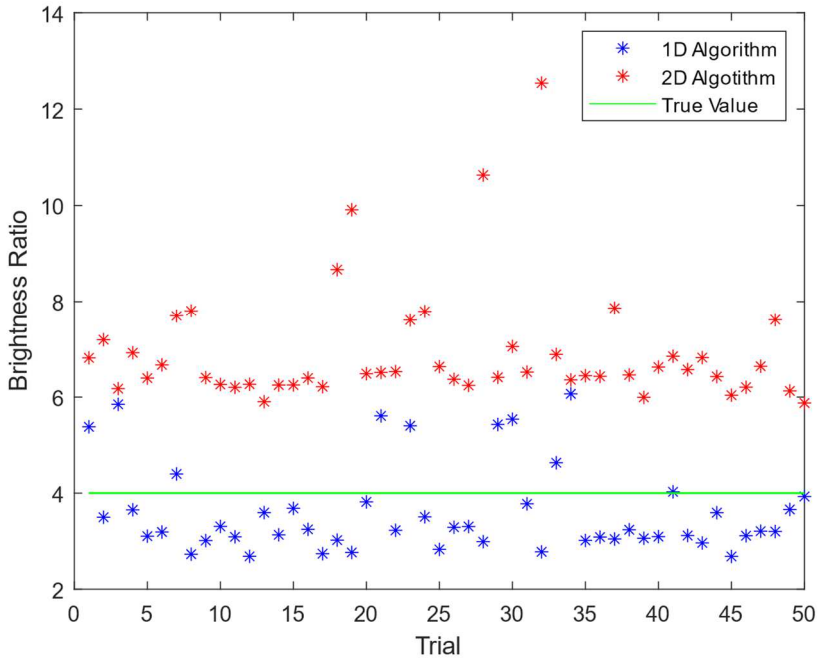

Fig. 49. Estimated Brightness Ratio (r_0=12cm)

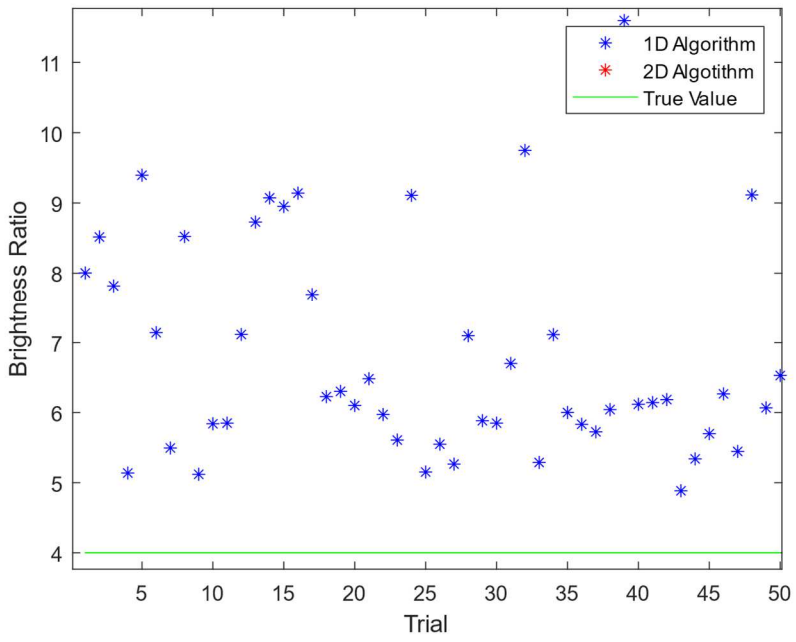

Fig. 50. Estimated Brightness Ratio (r_0=10cm)

When r_0 is equal to 14cm both algorithms appear to produce similar brightness ratio estimates, like the sections above, with the two-dimensional algorithm appearing to have slightly more variance and a higher tendency to overestimate the ratio. However, as r_0 decreases, the two-dimensional algorithm increasingly overestimates the ratio as it struggles to detect the dim source at all (See Fig. 41). Because of this, when r_0 is set to 10cm, the two-dimensional algorithm estimates an infinite brightness ratio.

4.1.2.2.3 Source Pixel Spacing

The number of pixels estimated between the two sources is shown for r_0 values of 14cm, 10cm, and 6cm in Fig. 33, Fig. 34, and Fig. 35, respectively.

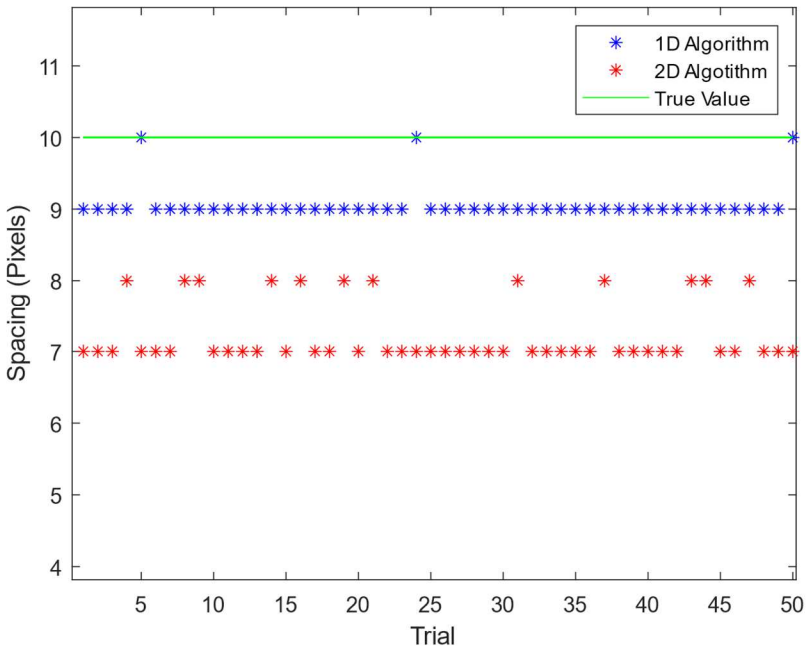

Fig. 51. Estimated Pixel Spacing (r_0=14cm)

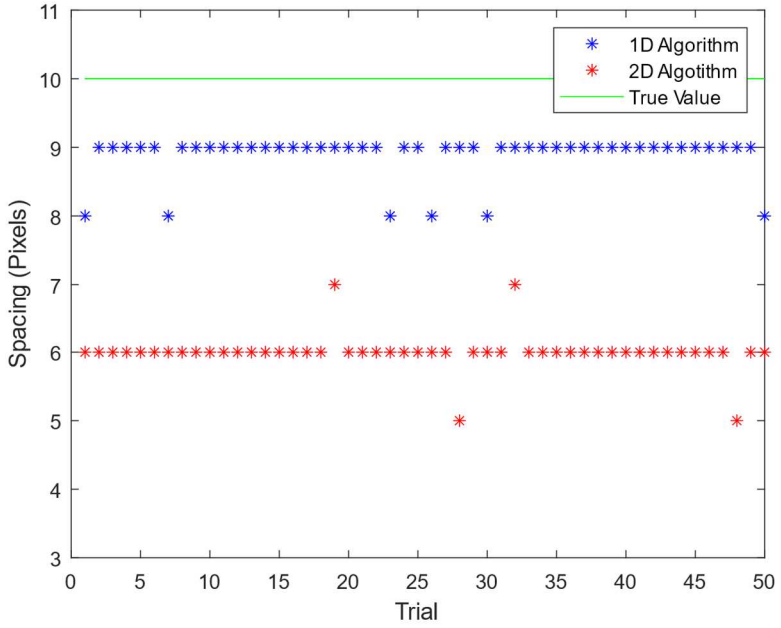

Fig. 52. Estimated Pixel Spacing (r_0=12cm)

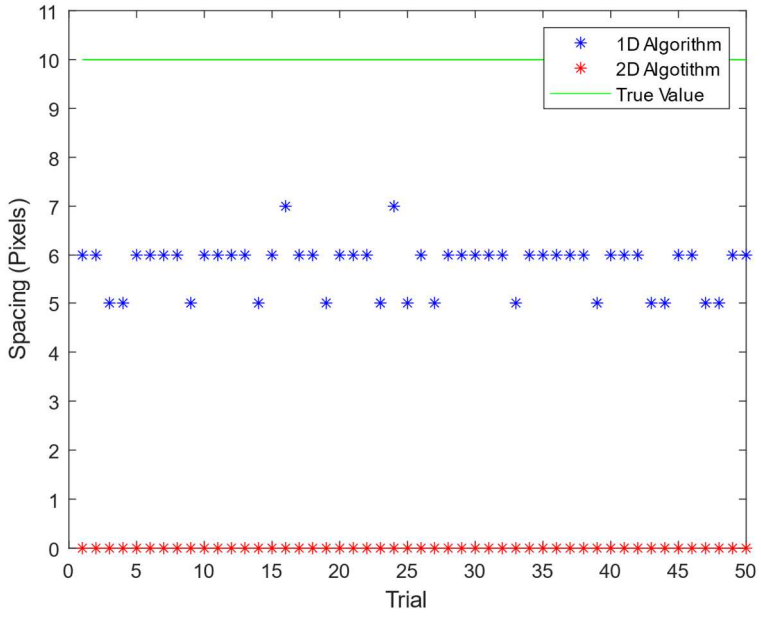

Fig. 53. Estimated Pixel Spacing (r_0=10cm)

We again observe that the one-dimensional algorithm estimates the pixel spacing between sources more accurately in every case, though as r_0 decreases, the accuracy of both algorithm's estimate decreases as well. We see that the two-dimensional algorithm was unable to detect the dim source at all when r_0 was set to 10cm.

4.1.2.2.4 Average Error Values

The averages values for MSE, brightness ratio and pixel spacing for each value of r_0 and at each brightness ratio are all shown together in Table 7, Table 8, and Table 9, below.

Table 7. Error Comparison (r0=14cm)

True Brightness Ratio	Algorithm Used	Average MSE	Average Brightness Ratio Error	Average Spacing Error
2:1	**1-D**	1.73×10^{5}	37%	2.4%
	2-D	1.10×10^{6}	28%	20%
4:1	**1-D**	1.34×10^{5}	10%	9.4%
	2-D	9.08×10^{5}	14%	28%
10:1	**1-D**	8.64×10^{4}	11%	17%
	2-D	8.98×10^{5}	Infinite	Infinite

With an r_0 value of 14cm, we observe that, on average, the one-dimensional algorithm produces lower error for most of the brightness ratios and metrics. A notable exception to this is the brightness ratio estimation where the true brightness ratio was 2:1. In this case the one-dimensional algorithm's brightness error was higher than that of the two-dimensional algorithm. It did estimate the pixel spacing correctly nearly every time in this case, producing a very low spacing error.

Table 8. Error Comparison (r_0=12cm)

True Brightness Ratio	Algorithm Used	Average MSE	Average Brightness Ratio Error	Average Spacing Error
2:1	**1-D**	1.17×10^6	10%	20%
	2-D	1.35×10^6	5.8%	30%
4:1	**1-D**	2.14×10^6	9.8%	20%
	2-D	1.16×10^6	73%	40%
10:1	**1-D**	2.59×10^5	14%	22%
	2-D	9.37×10^5	Infinite	Infinite

With an r_0 value of 12cm, we again see that, on average, the one-dimensional algorithm produces lower error for most of the brightness ratios and metrics. However, once again, the brightness ratio estimation of the one-dimensional algorithm where the true brightness ratio was 2:1 was higher than the estimation produced by the two-dimensional algorithm. It is unclear as to why exactly this tends to occur when a brightness ratio of 2:1 is used.

Table 9. Error Comparison (r0=10cm)

True Brightness Ratio	Algorithm Used	Average MSE	Average Brightness Ratio Error	Average Spacing Error
2:1	**1-D**	2.21×10^{6}	19%	30%
	2-D	1.32×10^{6}	50%	65%
4:1	**1-D**	1.74×10^{6}	70%	42%
	2-D	1.56×10^{6}	Infinite	Infinite
10:1	**1-D**	1.40×10^{6}	Infinite	Infinite
	2-D	9.89×10^{5}	Infinite	Infinite

Finally, with an r_0 value of 10cm, we observe that the one-dimensional algorithm produces a higher MSE than the two-dimensional algorithm. This is consistent with the results from the 1-1 brightness ratio and, again, is likely due to the way MSE is calculated. At this r_0 value we see that the one-dimensional algorithm produces a lower brightness ratio and spacing error for all brightness ratios except 10:1, where both algorithms are unable to detect the presence of the dim object.

4.2 Trials Using Lab-Collected Data

In this section we apply both the one-dimensional and two-dimensional algorithms to lab-collected data to compare their performance. All the tested data used was previously collected due to limited laboratory access at the time of research. This data was collected using a camera, computer, monitor, lens, and a heat source as shown in Fig. 54. The camera was set 1.5 meters from the computer monitor to be imaged. The

computer monitor displayed two-point sources of equal intensity similar to previous sections. The lens was used to focus the image onto the camera's sensor. Atmospheric turbulence was generated using a heat source placed between the lens and the monitor. This simulated an atmospheric boiling effect and created measurable atmospheric turbulence, though the amount of turbulence added was somewhat limited by the environment and equipment.

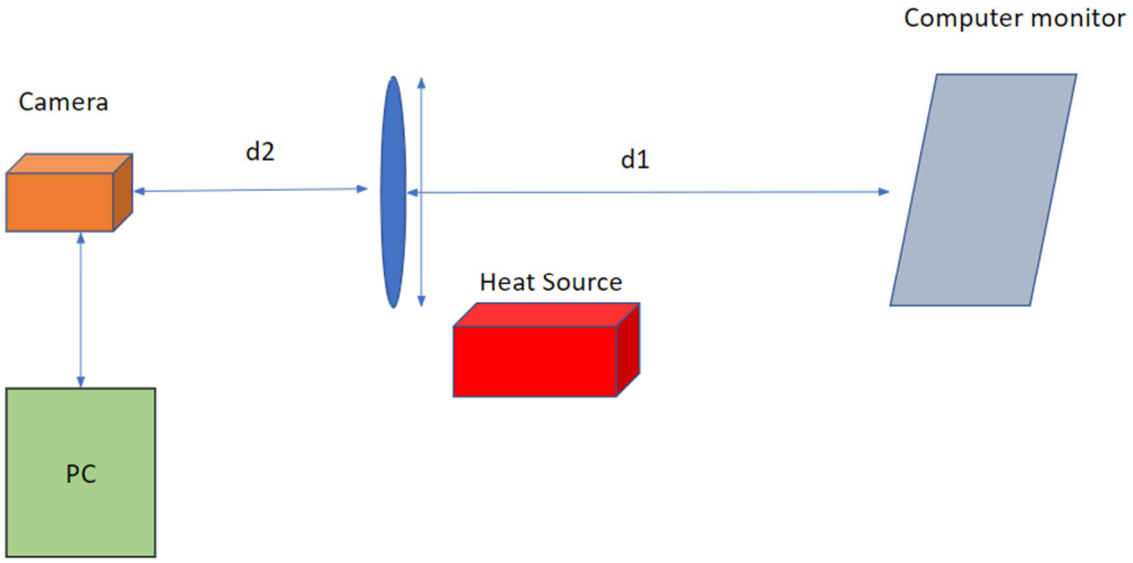

Fig. 54: Laboratory Setup

The specifications and parameters of the devices used in the collection of this data are shown below in Table 10.

Table 10: Data Collection Parameters and Specifications

Device		
Camera	Sensor Resolution (pixels)	3296 × 2472
	Pixel Size (µm)	5.5
	Exposure Time (ms)	10
	ADC Gain	1
	Aperture Diameter (mm)	$25.4 \pm .2$
Monitor	Screen Resolution (pixels)	1920 × 1080
	Aspect Ratio	16:9
	Pixel Size (mm)	0.276
Lens	Diameter (mm)	50 ± 0.025
	Focal Length (mm)	200 ± 4
	Aperture Ratio (F#)	4.0
	Lens Type	Achromatic
	Wavelength Range (nm)	750-1550

4.2.1 Distinctly Spaced Data

In this example, multiple short-exposure images were averaged to produce four long-exposure images with an exposure time of 250 ms each. An example of collected data is shown, is shown below in Fig. 55. Using center of mass calculation, we estimate the two-point sources to be approximately 10.3 pixels apart.

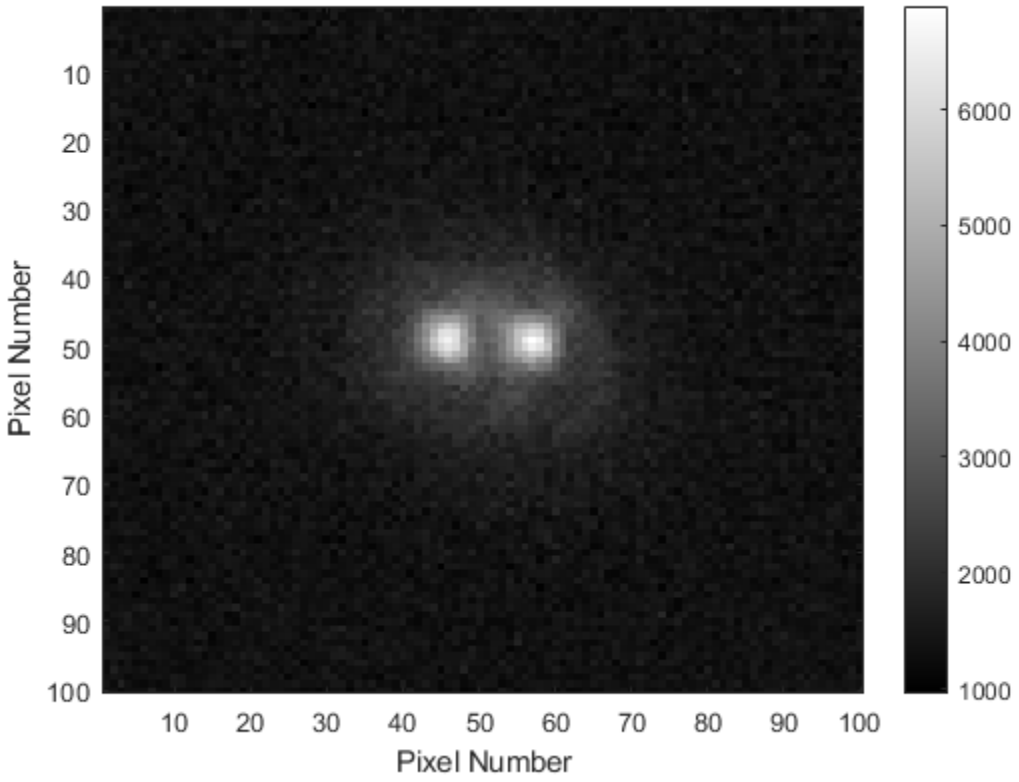

Fig. 55. Distinctly Spaced Collected Data

Just as we have done before, running the frame of simulated data through the two-dimensional blind deconvolution algorithm generates an estimate for the object. In this example, due to the wide spacing and low levels of atmospheric turbulence, we run only 20,000 iterations of each algorithm to produce an object estimate. To solve for the true value of r_0, we repeat the above process for various r_0 values, and calculating the log-likelihood using Eq. (50) as described earlier. Example plots of this calculated within the two-dimensional algorithm and one-dimensional algorithm from the above-mentioned collected data are shown in Fig. 56 and Fig. 57, respectively. We observe that the lowest plateau value is approximately 13cm on both plots and we use this as our estimated r_0.

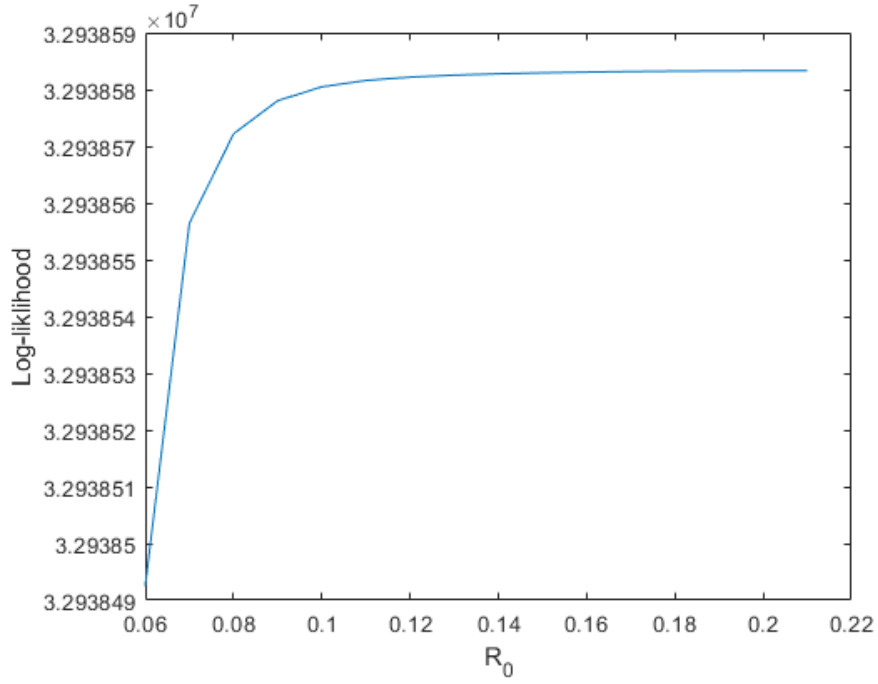

Fig. 56: Log-Likelihood Plot (Two-Dimensional Algorithm)

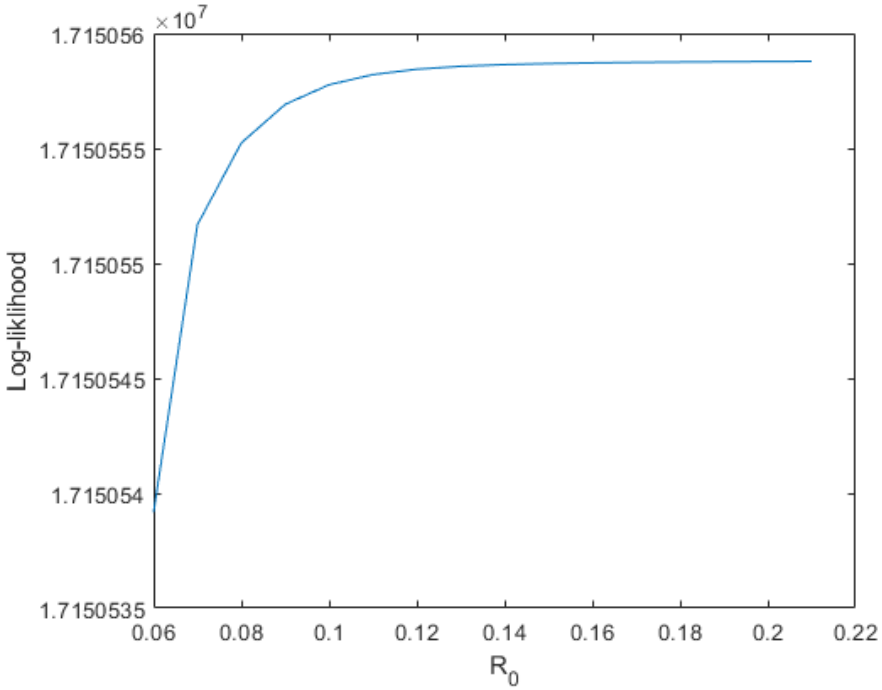

Fig. 57: Log-Likelihood Plot (One-Dimensional Algorithm)

The object estimate achieved using the estimated value of r_0 and the two-dimensional algorithm is magnified for detail and is shown in Fig. 58.

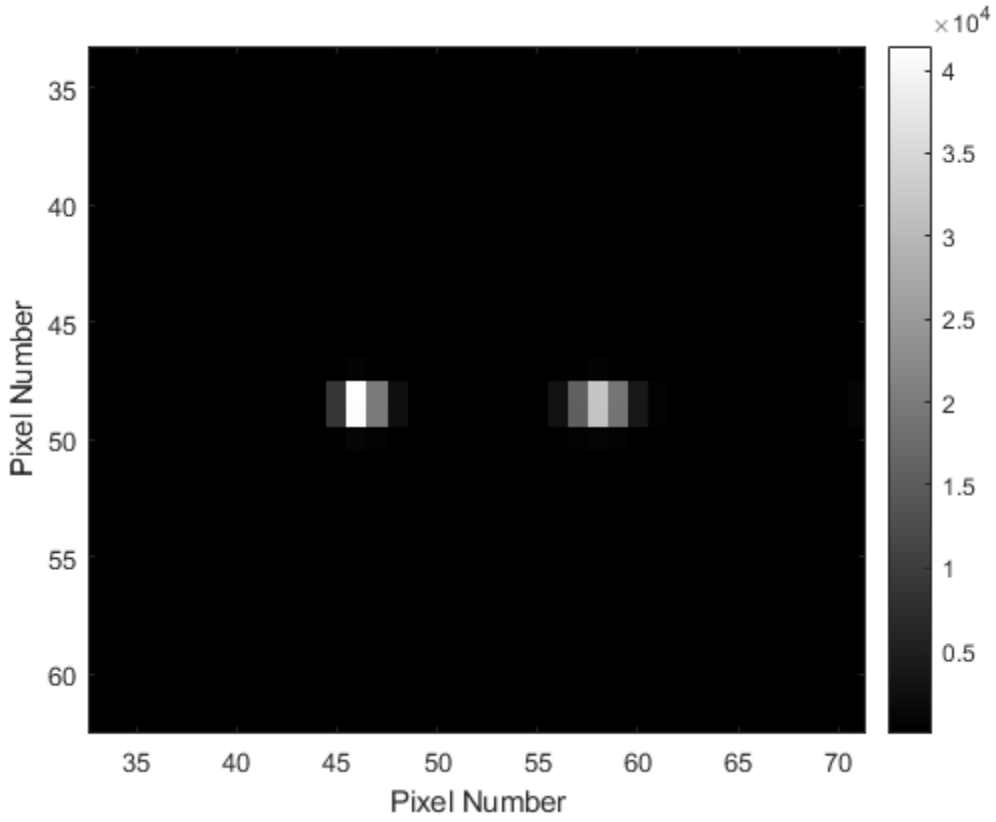

Fig. 58. Two-Dimensional Object Deconvolution Result (r_0=14cm)

Using the same estimated value of r_0 we run the data through the one-dimensional blind deconvolution algorithm to again generate estimates for the object and the PSF. Once again, the object estimate achieved using this one-dimensional algorithm is magnified for detail and shown in Fig. 59.

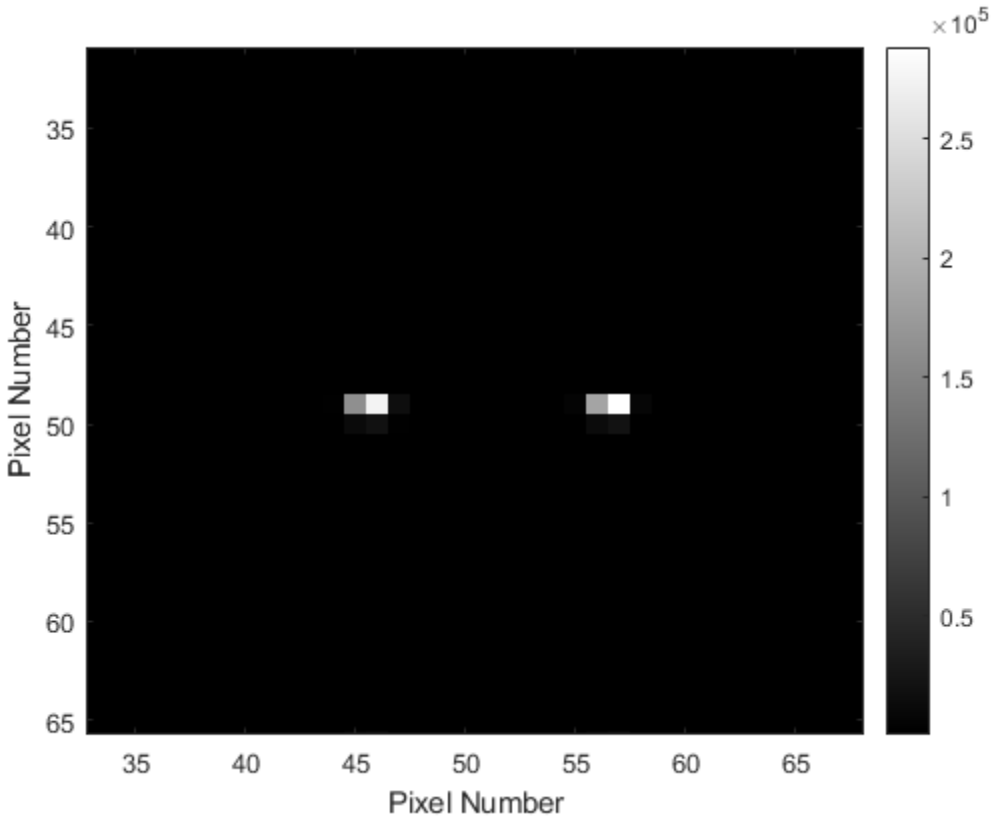

Fig. 59. One-Dimensional Object Deconvolution Result (Lab Data)

We observe that both algorithms converged at an object estimate of two sources spaced approximately the same distance apart. The one-dimensional algorithm appears to produce a sharper object estimate with more distinct point sources than that produced by the two-dimensional algorithm. In the following sub-sections, we will compare the speed and accuracy of the two algorithms.

4.2.1.2 Speed

The time required to perform deconvolution for each algorithm is shown below in Table 13. For this dataset we observe that the average that two-dimensional algorithm requires 11.11 seconds to complete 20,000 iterations. The one-dimensional algorithm takes only 0.4643 seconds on average, approximately 24 times faster.

Table 11: Speed Comparison

Trial		Time Required (seconds)	Speed Ratio (2D/1D)
1	**1D**	0.5060	22.71
	2D	11.49	
2	**1D**	0.4752	23.48
	2D	11.16	
3	**1D**	0.4405	24.27
	2D	10.69	
4	**1D**	0.4354	25.52
	2D	11.11	
Average	**1D**	**0.4643**	**23.93**
	2D	**11.11**	

4.2.1.1 Accuracy

With the exception of MSE error, we use the same performance metrics as used in the previous sections, including pixel spacing and brightness ratio. MSE error cannot be calculated in this case because we do not have access to an exact true image with which

we can compare the estimated object. Table 12 shows the estimated spacing, brightness ratio, and the associated errors.

Table 12: Accuracy Comparison (Distinctly Spaced Data)

Trial		Spacing (Pixels)	Spacing Error	Brightness Ratio	Brightness Error
1	**1D**	11	6.8%	1.17	17%
	2D	11	6.8%	1.06	6%
2	**1D**	10	3.9%	0.81	19%
	2D	10	3.9%	0.82	18%
3	**1D**	10	3.9%	1.09	9%
	2D	10	3.9%	0.84	16%
4	**1D**	9	12.6%	0.71	29%
	2D	9	12.6%	0.93	7%
Average	**1D**	**10**	**3.9%**	**0.94**	**6%**
	2D	**10**	**3.9%**	**0.91**	**9%**

We observe that for this dataset there is no significant difference in the accuracy between the two algorithms. This is likely due to the noticeably clear and distinct data set.

4.2.2 Closely Spaced Data

In this example, multiple short-exposure images were averaged to produce a single long-exposure image with an exposure time of 1 second. The collected data is shown below in Fig. 60. For this data collection the pixel spacing of the previous collection was reduced by half. We therefore estimate the spacing to be roughly 5.15 pixels between sources.

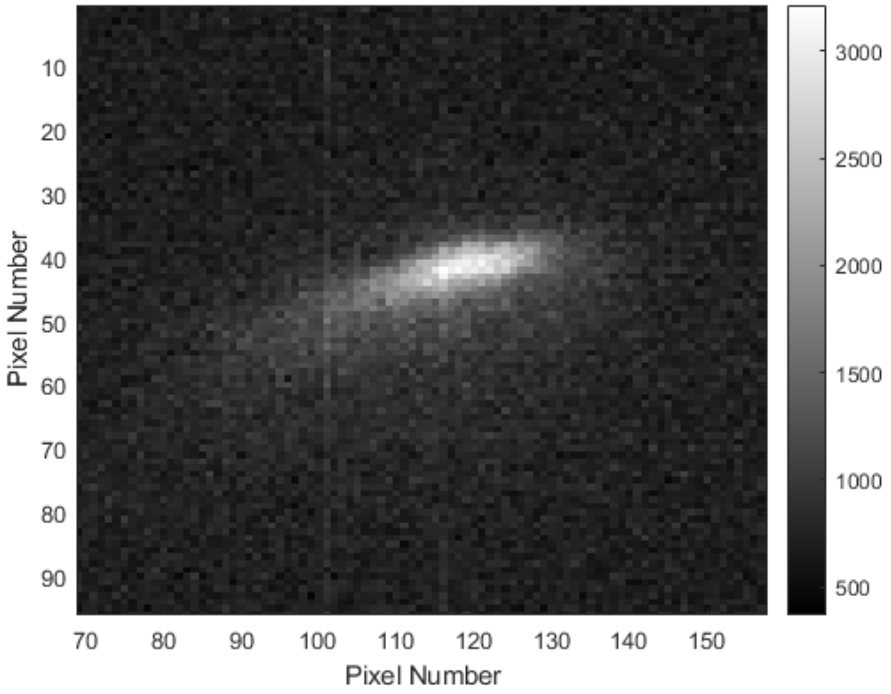

Fig. 60. Closely Spaced Collected Data

The simulated data was once again through the two-dimensional blind deconvolution algorithm generates an estimate for the object. To solve for the true value of r_0, using Eq. (50), again calculate the log-likelihood for various r_0 values, and compare them. In Fig. 61 and Fig. 62 we again show plots of this calculated within the two-dimensional algorithm and one-dimensional algorithm from the collected data. This data was processed using true laboratory optical values and therefore the r_0 values are at a smaller scale. However, the same principle applies and the seeing parameter can be solved for exactly as before. We observe that the lowest plateau value is approximately 0.35cm on both plots and we use this as our estimated r_0.

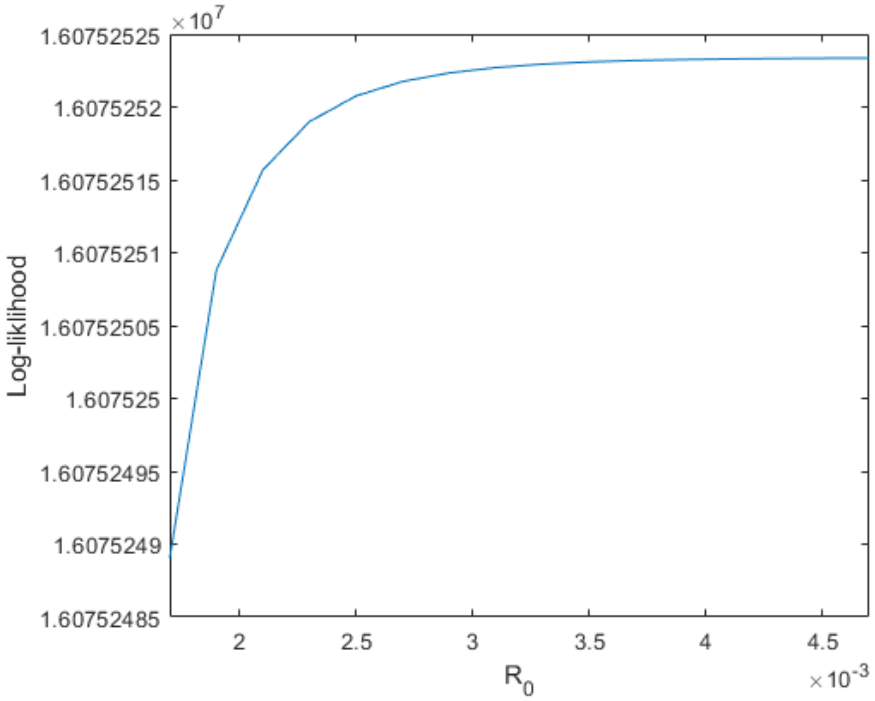

Fig. 61: Log-Likelihood Plot (Two-Dimensional Algorithm)

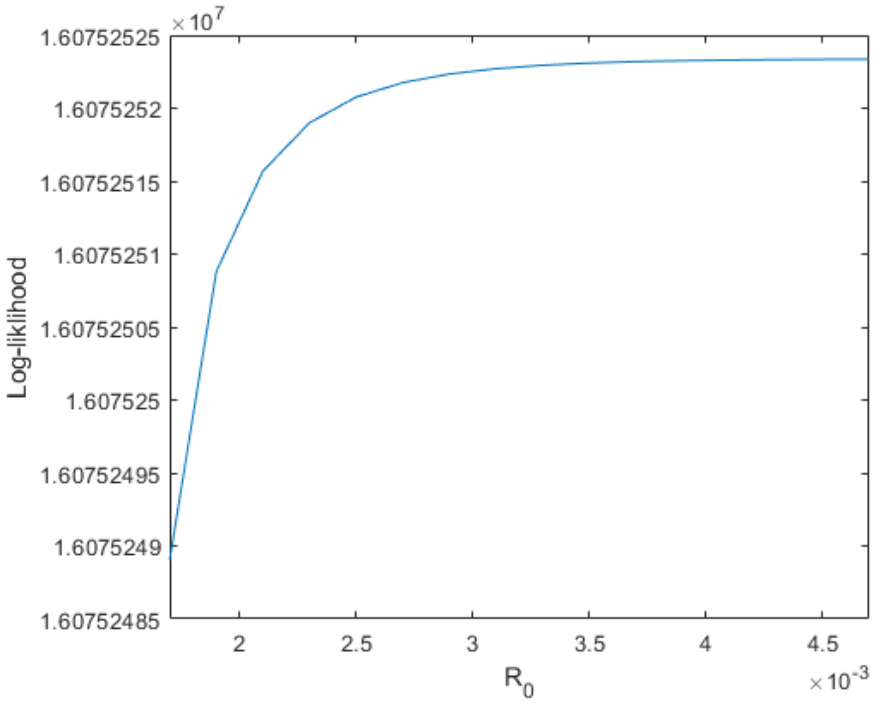

Fig. 62: Log-Likelihood Plot (One-Dimensional Algorithm)

The object estimate achieved using the estimated value of r_0 and the two-dimensional algorithm is magnified for detail and is shown in Fig. 63.

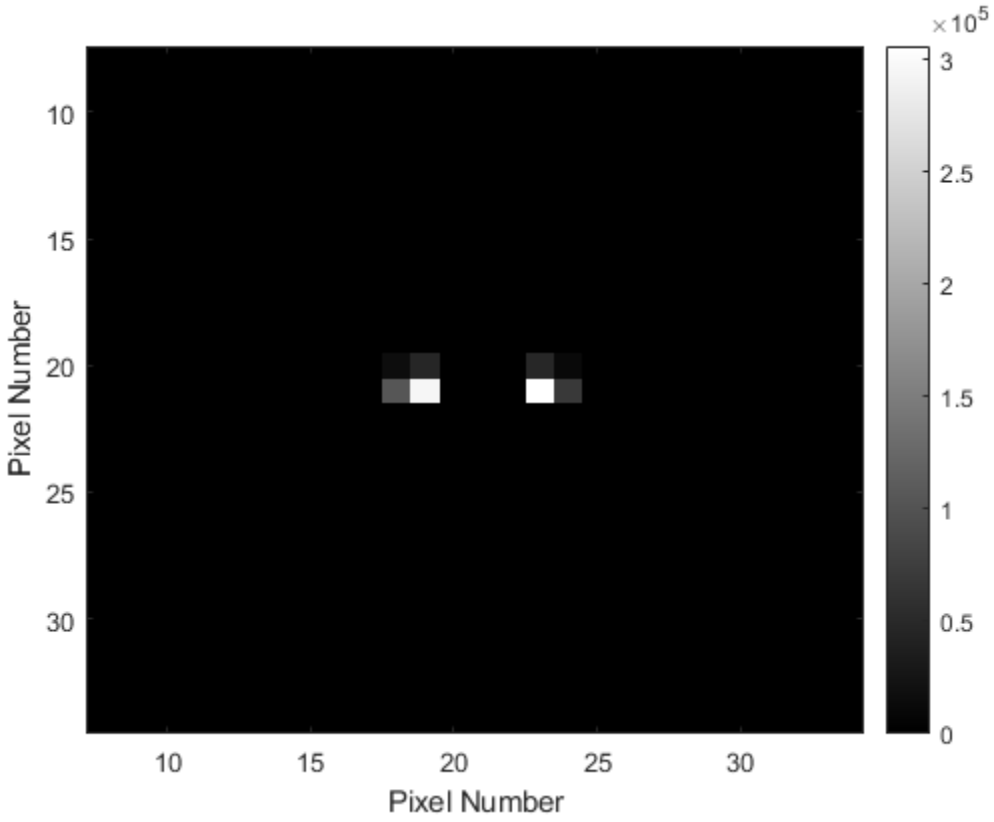

Fig. 63. Two-Dimensional Object Deconvolution Result (Lab Data)

Using the same estimated value of r_0 we run the data through the one-dimensional blind deconvolution algorithm to again generate estimates for the object and the PSF. Once again, the object estimate achieved using this one-dimensional algorithm is magnified for detail and shown in Fig. 64.

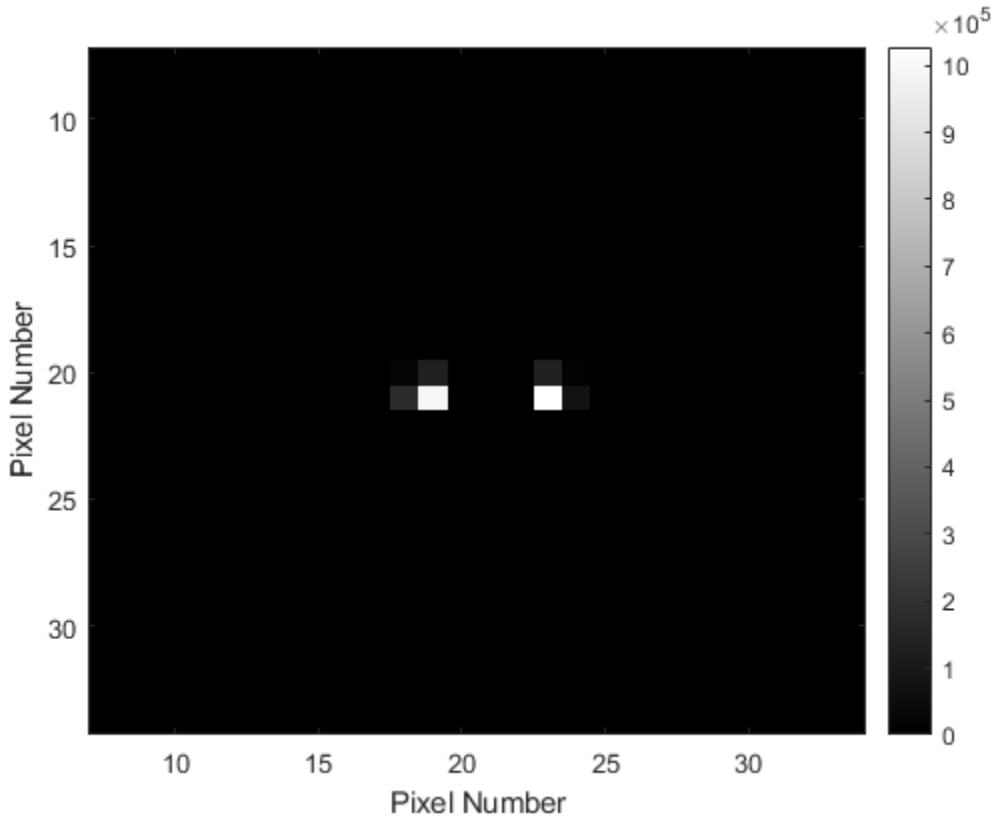

Fig. 64. One-Dimensional Object Deconvolution Result (Lab Data)

As was observed in the previous section, we see that both algorithms converged at an object estimate of two sources spaced approximately the same distance apart. The one-dimensional algorithm produces a sharper object estimate with more distinct point sources than that produced by the two-dimensional algorithm. The one-dimensional algorithm point source intensity values are approximately three times greater. In the following sub-sections, we will compare the accuracy and speed of the two algorithms.

4.2.2.2 Speed

The time required to perform deconvolution for each algorithm is shown below in Table 13. We observe that for this dataset the on average that two-dimensional algorithm

requires 47.66 seconds to complete 100,000 iterations. The one-dimensional algorithm takes only 1.963 seconds on average, approximately 25 times faster.

Table 13: Speed Comparison

Trial		Time Required (seconds)	Speed Ratio (2D/1D)
1	**1D**	1.923	25.18
	2D	48.50	
2	**1D**	1.920	26.41
	2D	50.78	
3	**1D**	1.947	25.97
	2D	50.62	
4	**1D**	2.062	23.66
	2D	48.72	
Average	**1D**	**1.963**	**24.96**
	2D	**49.66**	

4.2.2.1 Accuracy

We use the same performance metrics as used in the previous sections, including pixel spacing and brightness ratio. Though, the deconvolution was run multiple times as described above, it does not result in different results with the same data. Table 12 shows the estimated spacing, brightness ratio, and the associated errors.

Table 14: Accuracy Comparison (Distinctly Spaced Data)

	Spacing (Pixels)	Spacing Error	Brightness Ratio	Brightness Error
1D	**4.75**	**7.8%**	**0.95**	**5%**
2D	**4.75**	**7.8%**	**1.06**	**6%**

Once again, we observe that for this dataset there is no significant difference in the accuracy between the two algorithms.

V. Conclusion

This chapter explains the conclusions drawn from the results of testing using laboratory data and simulated data. It also expounds upon possible future research and testing that could be done to improve the speed, flexibility, and useability of the proposed one-dimensional algorithm.

5.1 Conclusions

Through simulation, it has been shown that the one-dimensional blind deconvolution algorithm is approximately 30 times faster than the two-dimensional blind deconvolution algorithm when both perform the same number of iterations. It has also been shown that the one-dimensional blind deconvolution algorithm is able to estimate more correctly the spacing between two sources regardless of the brightness ratio. It has also been shown that, on average, the algorithm was able to determine the brightness ratio slightly more accurately in most cases.

Using laboratory data, it was again confirmed that the one-dimensional algorithm can perform deconvolution faster than the two-dimensional algorithm. The lab-collected data was able to be processed approximately 25 times faster using the one-dimensional blind deconvolution compared with the two-dimensional algorithm. With the limited data available we observed no significant difference in the accuracy metrics between the two

algorithms. Further data collection and testing may yield more conclusive results in this area.

5.2 Future Work

To further improve the speed of the proposed algorithm, each object component vector could be solved for in parallel, possibly decreasing the time required by as much as 50%. Shifting the computational load to the GPU would likely increase the processing speed significantly as well.

Deriving an update equation for the PSF would simplify the application of the algorithm by eliminating the two-step process that involves first estimating the PSF and then choosing the correct deconvolution result. Incorporating the PSF update equation directly could possibly yield more accurate results as well.

This algorithm could also be further tested using actual imagery collected with a ground-based telescope. However, this type of data should not cause any issues when processing if long-exposure imagery is used and the object is spatially separable. Initial processing may be required depending on the level of noise in the imagery.

While the scope of this research extended only to objects that could be accurately separated into two components, this algorithm could be expanded to work with any object. By separating the two-dimensional image into multiple components, including diagonal sums, any object could be accurately estimated. Each object component could be solved in parallel and could still be extremely fast.

Bibliography

[1] M. R. Brown, "One-Dimensional Multi-Frame Blind Deconvolution Using Astronomical Data for Spatially Separable Objects," Air Force Institute of Technology, 2020.

[2] H. G. Lewis, "Sensitivity of the Space Debris Environment to Large Constellations and Small Satellites," *J. Br. Interplanet. Soc.*, no. 70, pp. 105–117, 2017.

[3] S. Kitajima, S. Abe, T. Hanada, and Kawamoto, "Influences of MEGA constellations on the orbital environment," in *Proceedings of the International Astronautical Congress, IAC*, 2016.

[4] T. Wekerle, J. B. Pessoa Filho, L. E. V. L. da Costa, and L. G. Trabasso, "Status and Trends of Smallsats and their Launch Vehicles — An Up-to-date Review," *J. Aerosp. Technol. Manag.*, vol. 9, no. 3, pp. 269–286, Aug. 2017, doi: 10.5028/jatm.v9i3.853.

[5] C. L. Matson and K. Borelli, "Parallelization and Automation of a Blind Deconvolution Algorithm," in *2006 HPCMP Users Group Conference (HPCMP-UGC'06)*, 2006, pp. 327–332, doi: 10.1109/HPCMP-UGC.2006.57.

[6] R. J. Noll, "Zernike polynomials and atmospheric turbulence*," 1976.

[7] H. A. Ferwerda, "Frits Zernike: His Life and Achievements," 1994, pp. 2–9, doi: 10.1117/12.171862.

[8] R. Garner, "Hubble's Mirror Flaw," *NASA*, 2015. [Online]. Available: https://www.nasa.gov/content/hubbles-mirror-flaw. [Accessed: 19-Oct-2019].

[9] E. P. Goodwin and J. C. Wyant, *Field Guide to Interferometric Optical Testing*.

Bellingham, WA: SPIE Press, 2006.

[10] "COSTAR (Corrective Optics Space Telescope Axial Replacement), Hubble, Flown," *National Air and Space Museum*, 2019. [Online]. Available: airandspace.si.edu/collection-objects/costar-corrective-optics-space-telescope-axial-replacement-hubble-flown. [Accessed: 19-Oct-2020].

[11] Z. Yang, "Incoherent Imaging in the Presence of Atmospheric Turbulence and Refractivity," University of Dayton, 2017.

[12] A. N. Kolmogrov, "The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers," *Proc. R. Soc. London. Ser. A Math. Phys. Sci.*, vol. 434, no. 1890, pp. 9–13, Jul. 1991, doi: 10.1098/rspa.1991.0075.

[13] R. G. Paxman, T. J. Rogne, B. A. Sickmiller, D. A. LeMaster, J. J. Miller, and C. G. Vollweiler, "Spatial stabilization of deep-turbulence-induced anisoplanatic blur," *Opt. Express*, vol. 24, no. 25, p. 29109, 2016, doi: 10.1364/oe.24.029109.

[14] P. Guo, Q. Yin, and J. Yu, *Advances in Neural Networks – ISNN 2016*, vol. 9719. Cham: Springer International Publishing, 2016.

[15] J. D. Drummond, "Adaptive optics Lorentzian point spread function," 1998, p. 1030, doi: 10.1117/12.321648.

[16] D. L. Fried, "Optical Resolution Through a Randomly Inhomogeneous Medium for Very Long and Very Short Exposures," *J. Opt. Soc. Am.*, vol. 56, no. 10, pp. 1372–1379, Oct. 1966, doi: 10.1364/JOSA.56.001372.

[17] D. L. Fried, "Limiting Resolution Looking Down Through the Atmosphere," *J. Opt. Soc. Am. A*, vol. 56, no. 10, pp. 1380–1384, 1966.

[18] T. R. Corle and G. S. Kino, "Introduction," in *Confocal Scanning Optical*

Microscopy and Related Imaging Systems, Elsevier, 1996, pp. 3–4.

[19] F. Sroubek and J. Flusser, "Multichannel blind deconvolution of spatially misaligned images," *IEEE Trans. Image Process.*, vol. 14, no. 7, pp. 874–883, Jul. 2005, doi: 10.1109/TIP.2005.849322.

[20] A. S. Carasso, "Direct Blind Deconvolution," *SIAM J. Appl. Math.*, vol. 61, no. 6, pp. 1980–2007, Jan. 2001, doi: 10.1137/S0036139999362592.

[21] T. F. Chan and C. K. Wong, "Total variation blind deconvolution," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 370–375, 1998, doi: 10.1109/83.661187.

[22] T. F. Chan and C. K. Wong, "Convergence of the alternating minimization algorithm for blind deconvolution," *Linear Algebra Appl.*, vol. 316, no. 1–3, pp. 259–285, 2000, doi: 10.1016/S0024-3795(00)00141-5.

[23] T. J. Schulz, "Multiframe blind deconvolution of astronomical images," 1993.

[24] E. M. Algorithm, A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the," 1977.

[25] F. Pernkopf and D. Bouchaffra, "Genetic-based EM algorithm for learning Gaussian mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1344–1348, Aug. 2005, doi: 10.1109/TPAMI.2005.162.

[26] L. Li, "GADEM: A Genetic Algorithm Guided Formation of Spaced Dyads Coupled with an EM Algorithm for Motif Discovery," *J. Comput. Biol.*, vol. 16, no. 2, pp. 317–329, Feb. 2009, doi: 10.1089/cmb.2008.16TT.

[27] A. M. Martínez and J. Vitrià, "Learning mixture models using a genetic version of the EM algorithm," *Pattern Recognit. Lett.*, vol. 21, no. 8, pp. 759–769, Jul. 2000, doi: 10.1016/S0167-8655(00)00031-3.

[28] X.-L. Meng, “The EM algorithm and medical studies: a historical linik,” *Stat. Methods Med. Res.*, vol. 6, no. 1, pp. 3–23, Feb. 1997, doi: 10.1177/096228029700600102.

[29] J. P. Klein, “Semiparametric Estimation of Random Effects Using the Cox Model Based on the EM Algorithm,” *Biometrics*, vol. 48, no. 3, p. 795, Sep. 1992, doi: 10.2307/2532345.

[30] T. Isobe, E. D. Feigelson, and P. I. Nelson, “Statistical methods for astronomical data with upper limits. II - Correlation and regression,” *Astrophys. J.*, vol. 306, p. 490, Jul. 1986, doi: 10.1086/164359.

[31] K. J. Lee, L. Guillemot, Y. L. Yue, M. Kramer, and D. J. Champion, “Application of the Gaussian mixture model in pulsar astronomy - pulsar classification and candidates ranking for the Fermi 2FGL catalogue,” *Mon. Not. R. Astron. Soc.*, vol. 424, no. 4, pp. 2832–2840, Aug. 2012, doi: 10.1111/j.1365-2966.2012.21413.x.

[32] Joseph W. Goodman, *Statistical Optics*. John Wiley & Sons, 2000.

[33] Z. Wang and A. C. Bovik, “Mean Squared Error : Love It or Leave It ?,” *IEEE Signal Process. Mag.*, vol. 26, no. January, pp. 98–117, 2009.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
26-03-2021	Master's Thesis	1 Oct 2019 - 25 Mar 2021

4. TITLE AND SUBTITLE

Spatially Separable Blind Deconvolution of Long Exposure Astronomical Imagery

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Lee, Justin, S, 1st Lt

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
Wright-Patterson AFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-21-M-057

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

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RDSM

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

12. DISTRIBUTION/AVAILABILITY STATEMENT

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

13. SUPPLEMENTARY NOTES

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

14. ABSTRACT

A spatially separable blind deconvolution algorithm is demonstrated that achieves a significantly faster processing time and superior sensitivity when processing long-exposure image data of unresolvable objects from a ground-based telescope. The proposed approach takes advantage of the structure of the long exposure point spread function's radial symmetric characteristics to approximate it as a product of one-dimensional horizontal and vertical intensity distributions. Objects at geosynchronous or geostationary orbit also can be well approximated as being spatially separable as they are, in general non-resolvable. The algorithm's performance is measured by computing the mean-squared error compared with the true object as well as the processing time required to perform the blind deconvolution. It will be shown that images processed by the proposed technique will possess, on average, a lower mean-squared error than images that are processed through the traditional two-dimensional blind deconvolution approach. In addition, the one-dimensional algorithm will be shown to perform the deconvolution significantly faster.

15. SUBJECT TERMS

^multi-frame blind deconvolution, spatially separable objects, spatially separable PSF, point-spread function, astronomical data deconvolution, one-dimensional blind deconvolution

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 C. Cain, AFIT/ENG
U	U	U	UU	108	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3636 x4716 stephen.cain@afit.edu