Spatially Separable Blind Deconvolution of Long Exposure Astronomical Imagery

Justin S. Lee, 1st Lt, USAF
Air Force Institute of Technology, Wright-Patterson AFB, OH 45324
Stephen C. Cain, Ph.D.
Air Force Institute of Technology, Wright-Patterson AFB, OH 45324

ABSTRACT

In this paper, 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 objects that are at geosynchronous orbit 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 is treated as an unknown variable in the image reconstruction problem.

Keywords: blind deconvolution, space object detection, imaging through turbulence

1. INTRODUCTION

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. This would significantly decrease processing time by decreasing the number of Fourier transforms and through the elimination of all two-dimensional Fourier transforms, drastically decreasing processing time required.

In short exposure imaging scenarios, the two-dimensional PSF is rarely spatially separable, but when an imaging system produces a long exposure image, the PSF tends to take on a shape similar to that of a two-dimensional Gaussian or a two-dimensional Lorentzian function depending on the amount of atmospheric compensation utilized during the exposure [1], [2]. Atmospherically compensated long exposure images make a superior starting point for obtaining high-resolution images of objects in space over images gathered with short exposures for many reasons. The primary causes of this are that compensated images achieve higher spatial bandwidth of the raw data and a lower amount of readout noise injected into the observation. Other algorithms have been proposed for achieving blind deconvolution of images in two-dimensions and will be used for comparison purposes in this research [3]–[5].

To achieve spatially separable blind deconvolution, two conditions must be met. First, the image of the object under observation predicted by geometric optics must be able to be described as the outer product of a horizontal vector and a vertical vector. This is generally true if this image as a matrix has a rank of one. Although most satellite images are not spatially separable, when they are imaged from distances greater than approximately 35,000 km, or the distance between geosynchronous earth orbit (GEO) and earth's surface, they appear as nearly point sources. Collections of these objects, therefore, would appear as collections of point sources, which will be shown in this paper to be spatially separable. The second condition is that the PSF as a matrix must also be able to be described as the outer product of a

horizontal vector and a vertical vector. This will be shown in this paper to be generally true for astronomical telescopes with little to no optical aberrations taking long-exposure images through the turbulent atmosphere.

To begin, we will discuss some of the models that are utilized in this simulation and comparison. These include models for satellite images, both optical and atmospheric transfer functions, and statistical models for the data used. Next, we will derive the one-dimensional blind deconvolution algorithm utilizing the statistical data models previously explored. Then, results of the performance comparison between a previously used two-dimensional algorithm and our newly derived one-dimensional algorithm will be discussed. Finally, we conclude that the proposed one-dimensional blind deconvolution algorithm proposed in this paper outclasses similar two-dimensional algorithms in both speed and accuracy.

2. MODELS UTILIZED

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.

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 (1) and (2). 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 and P_{tot} is the point spread function of the system.

$$H_{tot}(f_x, f_y) = H_{opt}(f_x, f_y) * H_s(f_x, f_y) \tag{1}$$

$$P_{tot}(f_x, f_y) = \mathcal{F}^{-1}\{H_{tot}(f_x, f_y)\} \tag{2}$$

2.2.1. Optical Transfer Function

The OTF due to the can be described as the autocorrelation of the pupil function. This is shown below in Eq. (3). This model is valid for systems with or without optical aberrations.

$$H_{opt}(f_x, f_y) = \frac{\iint_{-\infty}^{\infty} H\left(x + \frac{f_x}{2}, y + \frac{f_y}{2}\right) H^*\left(x - \frac{f_x}{2}, y - \frac{f_y}{2}\right) dxdy}{\iint_{-\infty}^{\infty} |H(x,y)|^2 \, dxdy} \tag{3}$$

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 optical transfer function (OTF) computed previously by Fried [6]. This description of the long exposure OTF is shown below in Eq. (4). 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 [7]. Also, in Eq. (4) is α, which 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. (5)

$$H_s(f_x, f_y) = e^{-3.44\left(\frac{\bar{\lambda}\sqrt{f_x^2+f_y^2}}{r_0}\right)^{5/3}\left[1-\alpha\left(\frac{\sqrt{f_x^2+f_y^2}}{\theta_0}\right)^{1/3}\right]} \tag{4}$$

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

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.

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. (6). The incomplete data is defined as d_k and the complete data is described as ζ_k., x and y describe pixels in the source plane, z and w describe pixels in the detector plane.

$$d_k(z,w) = \sum_x \sum_y \zeta_k(z,w,x,y) \tag{6}$$

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 (7) and (8).

$$d_{k1}(z) = \sum_w d_k(z,w) \tag{7}$$

$$d_{k2}(w) = \sum_z d_k(z,w) \tag{8}$$

Substituting our definition of the incomplete data from Eq. (6) into Equations (7) and (8) we obtain Equations (9) and (10), respectively.

$$d_{k1}(z) = \sum_w \sum_x \sum_y \zeta_k(z,w,x,y) \tag{9}$$

$$d_{k2}(w) = \sum_z \sum_x \sum_y \zeta_k(z,w,x,y) \tag{10}$$

We can also describe the statistical expected value of the incomplete data in Eq. (11) 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. B is defined as the background radiation level. The image intensity is the convolution of the spatially separated object and PSF as shown in Eq. (12).

$$E[d_k(z,w)] = i_k(z,w,o_1,o_2,h_1,h_2) \tag{11}$$

$$i_k(z,w,o_1,o_2,h_1,h_2) = \sum_x \sum_y o_1(x)o_2(y)h_1(z-x)h_2(w-y) \tag{12}$$

We now separate the expectation of d_k(z,w) shown in Equations (11) and (12) into one-dimensional vectors, starting with d_{k1}(z) as shown below in Equations (13)-(21).

$$E[d_{k1}(z)] = i_{k1}(z,o_1,h_1) \tag{13}$$

$$i_{k1}(z,o_1,h_1) = \sum_w \sum_x \sum_y o_1(x)o_2(y)h_1(z-x)h_2(w-y) \tag{14}$$

$$i_{k1}(z,o_1,h_1) = \sum_x \sum_y [\sum_w h_2(w-y)]o_1(x)o_2(y)h_1(z-x) \tag{15}$$

By definition, a PSF sums to one as shown in Eq. (16). Using this property we can simplify Eq. (15) yielding Eq. (17).

$$\sum_w h_2(w-y) = 1 \tag{16}$$

$$i_{k1}(z,o_1,h_1) = \sum_x \sum_y o_1(x)o_2(y)h_1(z-x) \tag{17}$$

For this algorithm to work correctly, o_2 must sum to one as shown in Eq. (18). 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 (53) - (55).

$$\sum_y o_2(y) = 1 \tag{18}$$

$$i_{k1}(z,o_1,h_1) = \sum_x [\sum_y o_2(y)]\, o_1(x)h_1(z-x) \tag{19}$$

$$i_{k1}(z,o_1,h_1) = \sum_x o_1(x)h_1(z-x) \tag{20}$$

Similarly for d_{k2}(w), we now separate the expectation of d_k(z,w) shown in Equations (11) and (12) into one-dimensional vectors,

$$E[d_{k2}(w)] = i_{k2}(w,o_2,h_2) \tag{21}$$

$$i_{k2}(w,o_2,h_2) = \sum_z \sum_x \sum_y o_1(x)o_2(y)h_1(z-x)h_2(w-y) \tag{22}$$

$$i_{k2}(w,o_2,h_2) = \sum_x \sum_y [\sum_z h_1(z-x)]o_1(x)o_2(y)h_2(w-y) \tag{23}$$

Again, a PSF sums to one as shown above in Eq. (16). Using this property we can simplify Eq. (23) yielding Eq. (24).

$$i_{k2}(w, o_2, h_2) = \sum_x \sum_y o_1(x) o_2(y) h_2(w - y) \tag{24}$$

In this case, we can treat the sum of o_1 as a known constant value because our algorithm will first calculate o_1 before estimating o_2 in which case we can simply multiply by this known value represented in Eq. (26) as the constant o_1^{sum}.

$$i_{k2}(w, o_2, h_2) = \sum_y [\sum_x o_1(x)] o_2(y) h_2(w - y) \tag{25}$$

$$i_{k2}(w, o_2, h_2) = \sum_y o_1^{sum} o_2(y) h_2(w - y) \tag{26}$$

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. (27).

$$E[\zeta_k(z, w, x, y)] = o_1(x) o_2(y) h_1(z - x) h_2(w - y) \tag{27}$$

We can also separate this expectation of the complete data as shown below in Equations (28) and (29).

$$E[\zeta_{k1}(z, x)] = o_1(x) h_1(z - x) \tag{28}$$

$$E[\zeta_{k2}(w, y)] = o_2(y) h_2(w - y) \tag{29}$$

2.4. Performance Metrics

As a measure of accuracy, the mean-squared error (MSE) will be calculated for the final estimated objects and PSFs resulting from the one-dimensional and two-dimensional blind deconvolution algorithms. This method has been thoroughly proven and used as a metric for accuracy over many years. [8] The MSE is calculated using Eq. (30), 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.

$$\boldsymbol{MSE = \frac{1}{m * n} \sum_{x=1}^{m} \sum_{y=1}^{n} [T(x, y) - o_1(y) o_2(x)]^2} \tag{30}$$

To measure the speed of each algorithm, we can simply determine which algorithm accomplishes the blind deconvolution, on average, in the least amount of time.

3. ALGORITHM DERIVATION

The EM algorithm is an effective method of solving for an of an unknown object and an unknown PSF introduced from the atmosphere and telescope aberration. 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 [9]. The steps and derivation found below show the underlying mathematical models and operations that make this blind deconvolution possible. In this paper we will briefly explore each step. 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

3.1. Generate Probability Function for Incomplete Data

The incomplete data is assumed to be Poisson-distributed at each point on the detector plane. Each pixel and frame are also assumed to be statistically independent. We can, therefore, define a probability function for one pixel of one frame below in Eq. (31).

$$P[d_k(z,w)] = \frac{i_k(z,w,o_1,o_2,h_1,h_2)^{d_k(z,w)} e^{-i_k(z,w,o_1,o_2,h_1,h_2)}}{d_k(z,w)!} \tag{31}$$

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 Eq.s (32) and (33). 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}.

$$P[d_{k1}(z)] = \frac{i_{k1}(z,o_1,h_1)^{d_{k1}(z)} e^{-i_{k1}(z,o_1,h_1)}}{d_{k1}(z)!} \tag{32}$$

$$P[d_{k2}(w)] = \frac{i_{k2}(w,o_2,h_2)^{d_{k2}(w)} e^{-i_{k2}(w,o_2,h_2)}}{d_{k2}(w)!} \tag{33}$$

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 Eq. (34) where k is defined the frame number.

$$P[d_{k1}(z)\forall(z) \in I_2] = \prod_k \prod_z \frac{i_{k1}(z,o_1,h_1)^{d_{k1}(z)} e^{-i_{k1}(z,o_1,h_1)}}{d_{k1}(z)!} \tag{34}$$

$$P[d_{k2}(w)\forall(w) \in I_2] = \prod_k \prod_w \frac{i_{k1}(w,o_2,h_2)^{d_{k2}(w)} e^{-i_{k2}(w,o_2,h_2)}}{d_{k2}(w)!} \tag{35}$$

3.2. 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 Eq. (36) where L is the log-likelihood function.

$$L(o_1,h_1) = \sum_k \sum_z d_{k1}(z) \ln[i_{k1}(z,o_1,h_1)] - i_{k1}(z,o_1,h_1) - \ln[d_{k1}(z)!] \tag{36}$$

$$L(o_2,h_2) = \sum_k \sum_w d_{k2}(z) \ln[i_{k2}(w,o_2,h_2)] - i_{k2}(w,o_2,h_2) - \ln[d_{k2}(w)!] \tag{37}$$

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 Eq. (38) the term is removed and will be disregarded in the remaining steps.

$$L(o_1, h_1) = \sum_k \sum_z d_{k1}(z) \ln[i_{k1}(z, o_1, h_1)] - i_{k1}(z, o_1, h_1) \tag{38}$$

$$L(o_2, h_2) = \sum_k \sum_w d_{k2}(w) \ln[i_{k2}(w, o_2, h_2)] - i_{k2}(w, o_2, h_2) \tag{39}$$

Finally, we substitute the definition of i_{k1} and the incomplete data into Eq. (38) to produce Eq. (40) in terms of the complete data, o_1 and h_1. We likewise substitute the definition of i_{k2} and the incomplete data into Eq. (39) to produce Eq. (41) in terms of the complete data, o_2 and h_2. L_{1CD} is defined as the log-likelihood function of the complete data.

$$L_{1CD}(o_1, h_1) = \sum_k \sum_z \sum_x \zeta_{k1}(z, x)\{\ln[o_1(x)] + \ln[h_1(z - x)]\} - o_1(x)h_1(z - x) \tag{40}$$

$$L_{2CD}(o_2, h_2) = \sum_k \sum_w \sum_y \zeta_{k2}(w, y)\{+\ln[o_2(x)] + \ln[h_2(w - y)]\} - o_1^{sum} o_2(y)h_1(z - x) \tag{41}$$

3.3. Derive Expectation

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 (42) and (43). We can also put these equations in terms of the complete data as shown in Equations (44) and (45).

$$Q_1(o_1, h_1) = E[L_{1CD}(o_1, h_1)|d_{k1}(z)] \tag{42}$$

$$Q_2(o_2, h_2) = E[L_{2CD}(o_2, h_2)|d_{k2}(w)] \tag{43}$$

$$Q_1(o_1, h_1) = \sum_k \sum_z \sum_x E[\zeta_{k1}(z, x)|d_{k1}(z)]\{\ln[o_1(x)] + \ln[h_1(z - x)]\} - o_1(x)h_1(z - x) \tag{44}$$

$$Q_2(o_2, h_2) = \sum_k \sum_w \sum_y E[\zeta_{k2}(w, y)|d_{k2}(w)]\{\ln[o_1^{sum}] + \ln[o_2(y)] + \ln[h_2(w - y)]\} - o_1^{sum} o_2(y)h_2(w - y) \tag{45}$$

3.4. Maximize Expectation

We now need to maximize the expectation shown in Eq. (44) 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. (46). We will then maximize o_2 by taking the derivative of Eq. (44) with respect to o_2 and setting it equal to zero and then solving for o_2 as well. This is shown in Eq. (49). 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.

$$\frac{dQ_1(o_1, h_1)}{do_1(x_0)} = \sum_k \sum_z \frac{E[\zeta_{k1}(z, x)|d_{k1}(z)]}{o_1(x_0)} - h_1(z - x) = 0 \tag{46}$$

$$\frac{dQ_2(o_2, h_2)}{do_2(y_0)} = \sum_k \sum_w \frac{E[\zeta_{k2}(w, y)|d_{k2}(w)]}{o_2(y_0)} - o_1^{sum} h_2(w - y) = 0 \tag{47}$$

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 (53) - (55). With these assumptions, we easily solve for o_1 from Eq. (46) as shown below in Eq. (48) 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.

$$o_1^{new}(x_0) = \frac{\sum_k \sum_z \sum_w \sum_y E[\zeta_k(z,w,x,y)|d_k(z,w)]}{K} \tag{48}$$

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 (49) and (50), yielding our second update equation.

$$\frac{dQ(o_1,o_2,h_1,h_2)}{do_2(y_0)} = \sum_k \sum_z \sum_w \sum_x \frac{E[\zeta_k(z,w,x,y)|d_k(z,w)]}{o_2(y_0)} - (l + o_1^{sum}) = 0 \tag{49}$$

$$o_2^{new}(y_0) = \frac{\sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)]}{l + o_1^{sum}} \tag{50}$$

To simplify, we can rename the sum of l and o_1^{sum} as L as shown in Eq. (51). Making this substitution gives the update equation for o_2 shown in Eq. (52).

$$L = l + o_1^{sum} \tag{51}$$

$$o_2^{new}(y_0) = \frac{\sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)]}{L} \tag{52}$$

Now, using Eq. (49), we can solve for L. This is outlined in Equations (53) - (55)

$$o_2(y)L = \sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)] \tag{53}$$

$$L\sum_y o_2(y) = \sum_y \sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)] \tag{54}$$

$$L = \sum_y \sum_k \sum_z \sum_w \sum_x E[\zeta_k(z,w,x,y)|d_k(z,w)] \tag{55}$$

These update equations for o_1 and o_2, given in Equations (48) and (52), will enable us to calculate improving estimates for the object in the blind deconvolution algorithm at each iteration.

3.5. 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. (56), using Bayes Theorem, the probability of the complete data given the incomplete data is given, where *P* is the probability notation.

$$P\big(\zeta_{k1}(z,x)\big|d_{k1}(z)\big) = \frac{P(\zeta_{k1}(z,x) \cap d_{k1}(z))}{P(d_{k1}(z))} \tag{56}$$

$$P\big(\zeta_{k2}(w,y)\big|d_{k2}(w)\big) = \frac{P(\zeta_{k2}(w,y) \cap d_{k2}(w))}{P(d_{k2}(w))} \tag{57}$$

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. (58).

$$P[d_{k1}(z)] = \frac{i_{k1}(z,o_1,h_1)^{d_{k1}(z)} e^{-i_{k1}(z,o_1,h_1)}}{d_{k1}(z)!} \tag{58}$$

$$P[d_{k2}(w)] = \frac{i_{k2}(w,o_2,h_2)^{d_{k2}(w)} e^{-i_{k2}(w,o_2,h_2)}}{d_{k2}(w)!} \tag{59}$$

Using the relationship between the complete and incomplete data from Eq. 1, 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 (60)-(62). We will use solve for the probability of vertical component, P[$d_{k1}(z)$] and apply the form of result to P[$d_{k2}(w)$].

$$d_1 = \zeta_{k1}(z,x_0) \tag{60}$$

$$d_2 = \sum_{x \neq x_0} \zeta_{k1}(z,x_0) \tag{61}$$

$$d_{k1}(z) = d_1 + d_2 \tag{62}$$

The expected value for d_1 and d_2 are shown below in Equations (63) and (64).

$$E[d_1] = m_1 = o_1(x_0)h_1(z-x_0) \tag{63}$$

$$E[d_2] = m_2 = \sum_{x \neq x_0} o_1(x_0)h_1(z-x_0) \tag{64}$$

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. (65). In Eq. (66), it can then be expressed in terms of d_{k1} and d_1 due to their relationship given in Eq. (61).

$$P(d_1 \cap d_2) = \frac{{m_1}^{d_1} e^{-m_1}}{d_1!} \frac{{m_2}^{d_2} e^{-m_2}}{d_2!} \tag{65}$$

$$P(d_1 \cap d - d_1) = P(d_1 \cap d_k) = \frac{{m_1}^{d_1} e^{-m_1}}{d_1!} \frac{{m_2}^{d_k - d_1} e^{-m_2}}{(d_k - d_1)!} \tag{66}$$

The probability of the incomplete data, from Eq. (58), can now be written in terms of m_1 and m_2 yielding Eq. (67).

$$P[d_k(z,w)] = \frac{(m_1 + m_2)^{d_k(z,w)} e^{-(m_1+m_2)}}{d_k(z,w)!} \tag{67}$$

Finally, we solve for the conditional probability, shown in Eq. (68), which is clearly a binomial distribution.

$$\frac{P(d_1 \cap d_{k1})}{P(d_{k1})} = \frac{d_{k1}!}{d_1(d_{k1} - d_1)} \left(\frac{m_1}{m_1 + m_2}\right)^{d_1} \left(\frac{m_2}{m_1 + m_2}\right)^{d_{k1} - d_1} \tag{68}$$

In order to solve for the expectation of the complete data given the incomplete data, N and p are defined in Equations (69) and (70), respectively.

$$N = d_{k1}(z) \tag{69}$$

$$p = \frac{m_1}{m_1 + m_2} \tag{70}$$

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. (71).

$$E[\zeta_{k1}(z,x)|d_{k1}(z)] = Np = d_{k1}(z) \frac{o_1(x_0)h_1(z - x_0)}{\sum_x o_1(x)h_1(z - x)} \tag{71}$$

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. (72).

$$E[\zeta_{k2}(w,y)|d_{k2}(w)] = Np = d_{k2}(w) \frac{o_2(y_0)h_2(w - y_0)}{\sum_y o_2(y)h_2(w - y)} \tag{72}$$

3.6. Solve for Update Equations

The final step to arrive at an update equation for o_1 is to take Eq. (48) and replace the expectation of the complete data given the incomplete data with the result from Eq. (71). This produces the update equation that will be used in the blind deconvolution algorithm and is shown in Eq. (73). Similarly, for o_2 we take Eq. (52) and again replace the expectation of the complete data given the incomplete data with the result from Eq. (71). This yields the update equation for o_2 and is shown in Eq. (74).

$$o_1^{new}(x_0) = \frac{\sum_k \sum_z \frac{d_{k1}(z) o_1^{old}(x) h_1^{old}(z - x_0)}{\sum_x [o_1(x)h_1(z - x)] + B}}{K} \tag{73}$$

$$o_2^{new}(y_0) = \frac{\sum_k \sum_w \frac{d_{k2}(w) o_2^{old}(y) h_2^{old}(w - y_0)}{\sum_y [o_2(y)h_2(w - y)] + B}}{L} \tag{74}$$

It is important to note that the update equation for o_1 is not dependent on the estimate of o_2 and vice-a-versa. This makes it possible to compute these estimates in parallel, which would enable further speed increases. This was not implemented in the trials shown in Section 4.

3.7. Solve for the Seeing Parameter

To solve for the correct PSF, the algorithm is run multiple times using various r_0 values. A log-likelihood is calculated for each solution using Eq. (38). 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.

4. RESULTS

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 (3) and (5), 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 parameter will be set to 14cm for these trials. 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 binary system is modeled using two pixels separated by ten pixels as shown in Fig. 1. The PSF is generated as described in the paragraph above and in section 2.2. An example is shown below in Fig. 2. Simulated PSF.

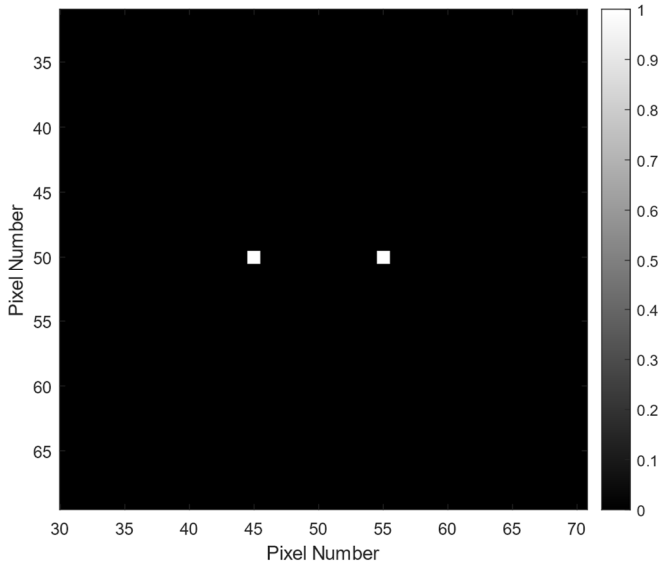

Fig. 1. Binary Star System Model

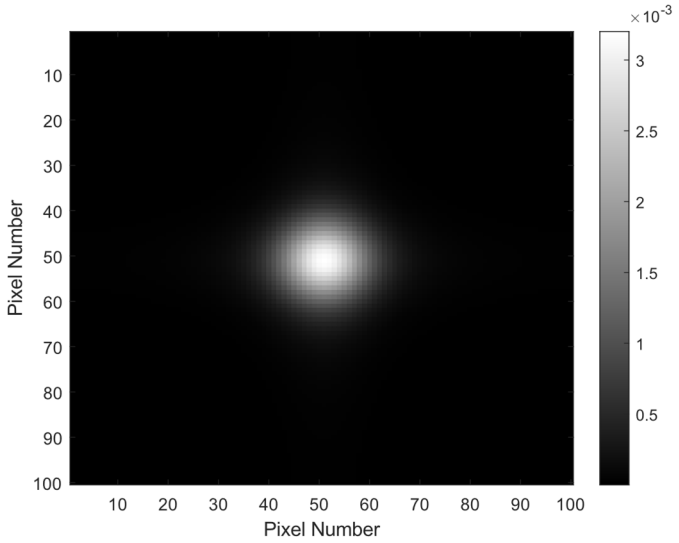

Fig. 2. Simulated PSF

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. 3.

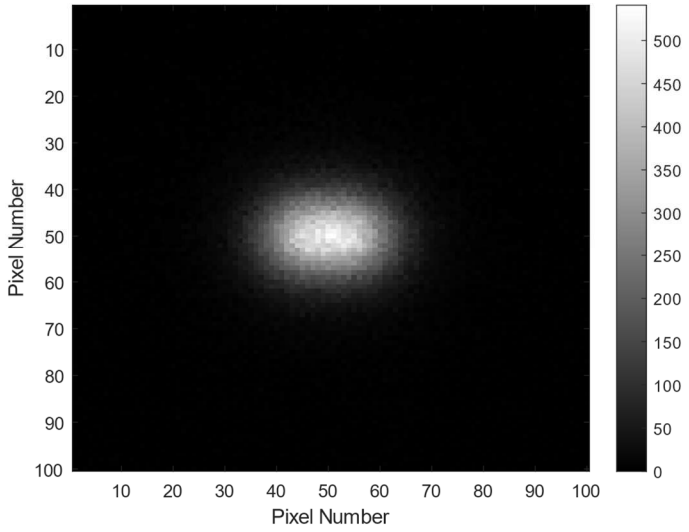

Fig. 3. Simulated data

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. 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. 4.

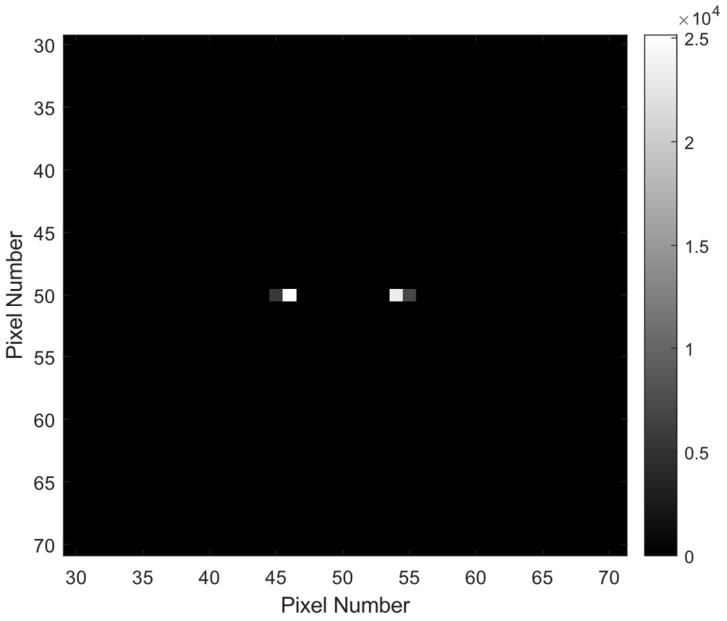

Fig. 4. Two-Dimensional Object Deconvolution Result

Running these two frames of data through the one-dimensional blind deconvolution algorithm generates estimates for the object and the PSF. The object estimate achieved using this one-dimensional algorithm is magnified for detail and is shown in Fig. 5.

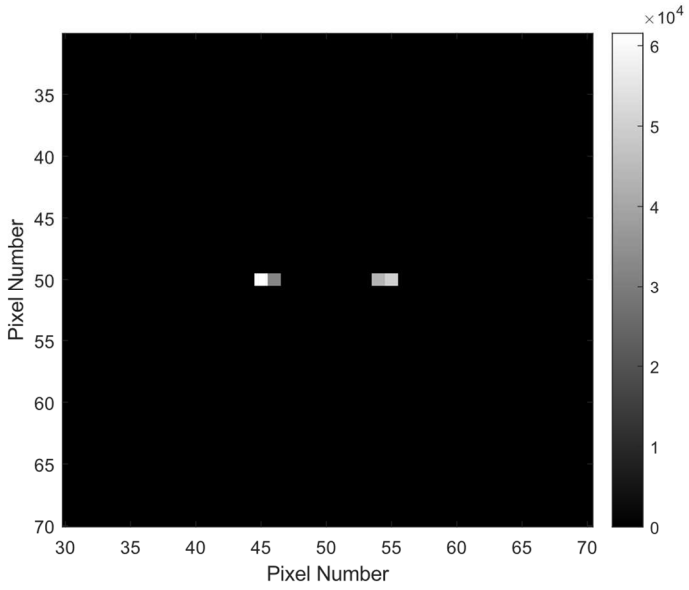

Fig. 5. One-Dimensional Object Deconvolution Result

Upon inspection, we see that both algorithms generated relatively accurate results in comparison to the true object. 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. Accuracy

Using the mean-squared error technique, calculated using Eq. (30), we can compare the accuracy of the two-dimensional algorithm with that of the one-dimensional algorithm. Each algorithm performed the deconvolution fifty times with Poisson noise randomly generated for each trial. 200,000 iterations of the algorithm were performed in each trial. Fig. 6. Mean-Squared Error Comparison shows the calculated MSE for each trial and Table 1 shows the average results of our mean-squared error calculation.

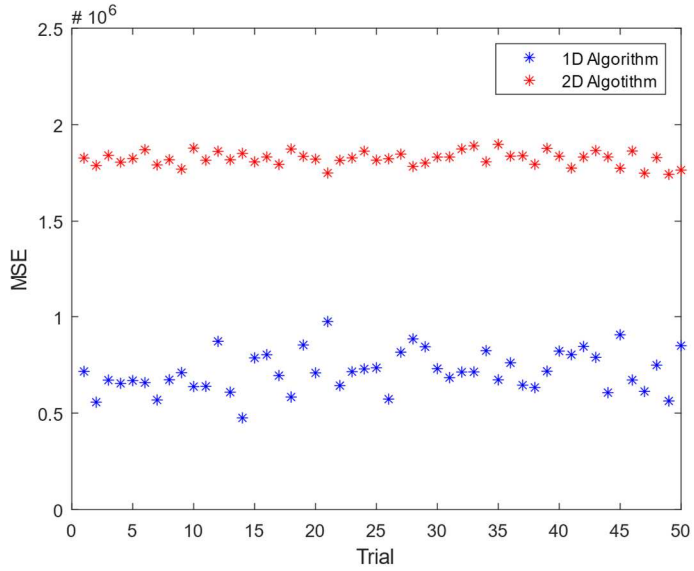

Fig. 6. Mean-Squared Error Comparison

Algorithm Used	Average Mean-Squared Error
One-Dimensional	7.15×10^{5}
Two-Dimensional	1.82×10^{6}

Table 1. Average Mean-Squared Error

We observe that the mean-squared error for our one-dimensional algorithm is lower than that of the two-dimensional algorithm in all cases and is 61% lower on average.

4.2. Speed

To measure how quickly each algorithm can perform the required deconvolution, we simply use MATLAB's built-in functions to record the time used. Fig. 7. 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.

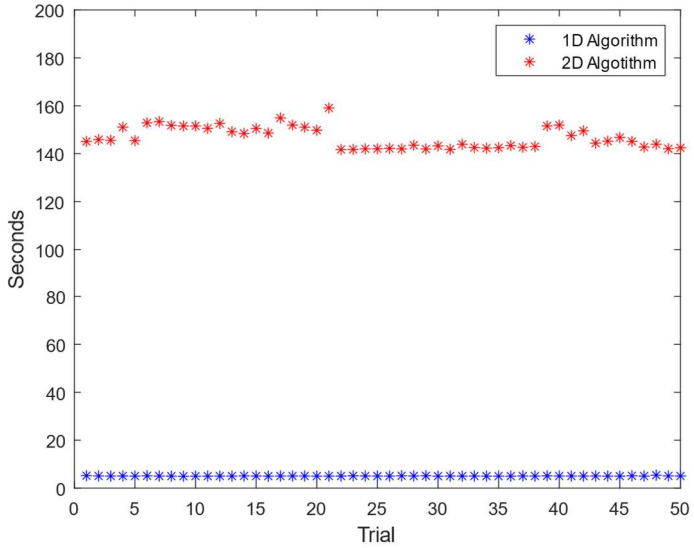

Fig. 7. Time Required Comparison

Algorithm Used	Average Time Required (Seconds)
One-Dimensional	5.059
Two-Dimensional	146.7

Table 2. Average Time Required

We observe that, on average, the one-dimensional algorithm performs the blind deconvolution 29 times faster than the two-dimensional algorithm.

5. CONCLUSIONS

It has been shown, through simulation, that the one-dimensional blind deconvolution algorithm was able to more accurately reconstruct the object based on a lower mean-squared error when compared to that of the two-dimensional algorithm. It has also been shown that the one-dimensional blind deconvolution algorithm was able to perform the deconvolution more quickly. This is significant because it could enable real-time processing in many situations where previously deconvolution had to be done in post-processing. Further speed increase could be attained using parallel computing.

6. REFERENCES

[1] K. Ahi, "Mathematical Modeling of THz Point Spread Function and Simulation of THz Imaging Systems," *IEEE Trans. Terahertz Sci. Technol.*, vol. 7, no. 6, pp. 747–754, Nov. 2017, doi: 10.1109/TTHZ.2017.2750690.

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

[3] J. Huang and M. Shen, "Multiframe blind deconvolution restoration of atmospheric turbulence-degraded images based on noise characteristic," *Guangxue Xuebao/Acta Opt. Sin.*, 2008, doi: 10.3788/AOS20082809.1686.

[4] A. S. Carasso, "APEX blind deconvolution of color Hubble space telescope imagery and other astronomical data," *Opt. Eng.*, vol. 45, no. 10, p. 107004, Oct. 2006, doi: 10.1117/1.2362579.

[5] R. M. Aung and S. C. Cain, "Multi-Frame Blind Deconvolution of Closely Spaced Dim Stellar Objects," 2019.

[6] D. L. Fried, “Limiting Resolution Looking Down Through the Atmosphere,” *J. Opt. Soc. Am. A*, vol. 56, no. 10, pp. 1380–1384, 1966.
[7] Joseph W. Goodman, *Statistical Optics*. John Wiley & Sons, 2000.
[8] J. R. Fienup, “Space Object Imaging Through The Turbulent Atmosphere,” *Opt. Eng.*, vol. 18, no. 5, 1979.
[9] E. M. Algorithm, A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data via the,” 1977.