Phase Retrieval by Flattening the Wavefront

Ante Qu
Advised by Professor Jason W. Fleischer

A senior thesis submitted in partial fulfillment
of the requirements for the degree of
Bachelor of Arts in Physics
at Princeton University

May 4, 2015

This paper represents my own work in accordance with University regulations.

Ante Qu

Abstract

Many objects of interest in imaging, such as biological cells or turbulent air, are phase-only objects that are transparent and thus produce little to no contrast in wide-field microscopes. The phase accumulated by this light carries important information about the refractive index and the thickness of the object. We propose a method for retrieving the phase by using a spatial light modulator (SLM) to conjugate the phase of the object, flattening the wavefront of light passing through the SLM and the object. After we flatten the wavefront, the resulting configuration on the SLM is the conjugate of the phase image, which we can easily invert to recover the original phase image. This method retrieves the phase without using any prior knowledge about the object.

Our algorithm performs a decomposition of the image into basis functions and searches for the coefficients that yield the flattest output intensity pattern. This algorithm takes advantage of the fact that a relatively small number of basis elements can store the majority of the information in the image. Popular phase retrieval methods such as the Gerchberg–Saxton algorithm can only converge to the phase image under light that is sufficiently coherent. From our simulations, we find that our method consistently produces correlations of over 99% with the original phase image, using either incoherent or coherent light and only 10% as many basis elements as the number of pixels in the image. We believe this result is a strong indication that this method will be able to reliably retrieve a direct phase image in the laboratory.

Phase Retrieval by Flattening the Wavefront
Ante Qu

Acknowledgements

I am indebted to my advisor, Prof. Jason Fleischer, who has met with me regularly and has continually offered me many insights that have helped me re-appreciate the beauty of physics. I would like to thank Alexandre Goy, who spent countless hours of his time explaining various concepts of optics to me and making sure that I understood the intuition. I also want to thank Jen-Tang Lu for showing me how to set up the optics equipment and run some of the simulation code and for giving me Image I (Smiley Face), which has kept me smiling the whole year whenever I run my simulations. A special mention goes to Victor Luu for creating the other sample images for my simulations. Alaka Halder, Kelly Kremer, Calvin Gross, Ming-Yee Tsang, and Vincent Po also deserve special mention for reading through my thesis and providing feedback. The Class of 1955 Fund funded this thesis.

Thank you to my parents, who have loved me, given me the opportunity to come to Princeton, and continually checked up on me throughout my time here. To my brother Alan and my cousin Grace, thank you for being great friends.

I have been blessed this year by the company of my three roommates, Joshua Bocarsly, Calvin Gross, and Daniel Hwang. Furthermore, I also thank Ming-Yee Tsang for being a good roommate last year and a great friend (in general).

Manna Christian Fellowship and Princeton Evangelical Fellowship have both helped me grow so much during my Princeton career. I am also thankful to Allen Fang, my disciplee, who has also helped me grow a ton this year, as well as Vincent Po, my thesis minion, for being so willing to do fun things for me like cleaning my desk and making me macarons. Many thanks also to the Manna Class of 2015, which provided me with a home away from home since I came to Princeton. This class suggested the thesis phrase, "It's not about you," which alludes to how life isn't just about ourselves; we are part of a larger story. To the Manna Class of 2017, thank you for just collectively being around everywhere and willing to chat with or feed me. You have made my day during the toughest days of the semester.

Thank you to the Brown Food Cooperative, for making this year fun and for keeping me fed. I have been very blessed by the community that formed this year. Lastly, I want to thank everyone who kept me company[1] while I thesised.

> *For I am sure that neither death nor life, nor angels nor rulers, nor things present nor things to come, nor powers, nor height nor depth, nor anything else in all creation, will be able to separate us from the love of God in Christ Jesus our Lord. (Romans 8:38–39,* ESV*)*

May 4, 2015
Ante Qu

[1] Diane Cho, Tiffany Huang, Florence Hsiao, my roommates, Ming-Yee Tsang, David Kong, Victoria Su, Samuel Kim, Rachel Reiss, Shimin Ooi, Victor Luu, Andrew Min, Steph Jeong, Evan Chow, Karis Yi, Paula Pacheco, Amanda Li, Michelle Kim, Lisa Lee, Alaka Halder, Michael Li, Kelly Kremer, Jeffmin Lin, Kate Shulgina, Sophie Wang, Jojo Cheng, Eunhae Park, and Ming-Ming Tran

Soli Deo gloria

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Light has many properties, two of which are intensity and phase. When treating light as a wave, its intensity is related to its amplitude, and its phase is the fraction of the wave cycle that the electric (E) and magnetic (B) fields have completed at a point in time and space. While we can observe most objects using just the intensity of the light coming from the object, many objects of interest in imaging, such as biological cells or density variations in transparent media, are phase-only objects that are transparent. and thus provide little or no contrast when viewed through wide-field imaging systems. Conventional cameras cannot detect the phase of the optical waves because their exposure times, which are the durations over which the energy from light is integrated, are much larger than the period of the electromagnetic oscillations of the light. The intensity of light that passes through such objects is not affected, whereas the phase of the light waves may change strongly. This change in phase is known as the wave retardation due to the object. The wave retardation accumulated by this light carries information about the effective optical path length, which is the product of the refractive index and the thickness of the object. This information is especially important in both atmospheric contexts, for understanding phenomena such as turbulence,[1] and biological contexts, for understanding the structure of microscopic living organisms.[2]

Phase retrieval[3][4] is the process of recovering the phase information lost when making physical measurements that are only sensitive to intensity. Various past approaches to solve this problem include the Gerchberg–Saxton algorithm,[5] phase contrast microscopy,[6] and Schlieren imaging.[7] We will soon explain the Gerchberg–Saxton algorithm and phase contrast microscopy to illustrate some of the drawbacks of these methods that our approach will be able to address.

Another property of light is its coherence. The coherence of light measures how

likely a phase measurement of a light wave at one point in time and space will be able to accurately predict the phase of the wave after it has travelled for some time. If the light is not monochromatic, then the different wavelengths would cause the light to lose coherence as it propagates, as different frequencies of light change their phase at different rates. Most lights from lasers are relatively coherent, while light from room lighting is often incoherent. Incoherent light behaves differently from coherent light as it propagates: incoherent light has a much shorter correlation time than coherent light, so information from interference patterns is lost due to averaging by detectors. As a result, interference patterns do not appear as easily under incoherent illumination.

One of the most commonly used phase-retrieval methods, which requires a coherent illumination of the object, is the Gerchberg–Saxton algorithm.[5][8] This algorithm is a numerical approach developed in the 1970s that takes two images of an object, one in the object plane and one after propagating into the far field, and iteratively calculates the phase by propagating it in simulation. While this approach is useful for recovering a direct phase image, it requires the illumination to be coherent: this is because the phase is not well-defined under incoherent lighting, making it difficult to be able to simulate both the forward and the inverse propagations.

A technique that enables phase retrieval under incoherent lighting is phase contrast microscopy.[6] It is a revolutionary technique developed in the 1940s that enables a user to see phase shifts in light by using a two-lens setup with a phase-shift mask at the focus between both lenses. If the phase-shift mask were narrow enough, the phase contrast microscope would be able to produce a direct phase image, but it is not feasible to produce such a narrow mask. In practice, the user can only see phase contrasts, which are the sharp boundaries of regions of smooth phase. While this technique enables an image of phase contrasts, it does not reconstruct the direct phase image.

To address these drawbacks, we propose an approach for phase retrieval that "inverts" the distortions due to the sample. This approach combines structured illumination[9][10] and adaptive optics[11] with a spatial light modulator (SLM) to "pre-compensate" the effect of the object, so that after passing a flat beam through both the SLM and the object, the light looks like a flat beam again. When the camera records a flattened wavefront, optimization is achieved and the SLM displays a phase-conjugated version of the object phase. This approach can produce a direct phase image, not only with coherent illumination, but also with incoherent

illumination. The quality of the reconstruction does not deteriorate as the illumination gets less and less coherent. As a result, our approach enables phase retrieval without the need for coherent light.

Before proceeding to the lab to test this approach with an actual setup, we simulated the process in order to test it without the noise and other difficulties associated with the experiment. In addition, the relative ease of simulations enabled us to easily fine-tune the process without expending large amounts of time. Since simulations have limitations, an actual experiment will ultimately be necessary to confirm the feasibility of our process. However, the actual experiment has not yet been performed, but we designed the simulations to match the experimental conditions as closely as possible.

In Chapter 2 we will introduce the fundamental principles of wave optics, describe the Gerchberg–Saxton in more detail, and introduce the concept of incoherent light. We will then discuss the details of our phase retrieval process in Chapter 3, starting with the optical setup, followed by the math that motivates the development of our specific process, finishing with the implementation details for the exact process. Afterwards, in Chapter 4, we will present the results of our simulations and show that the reconstruction qualities for both coherent light are acceptable. In Chapter 5 we will show the same results for incoherent light and show that under incoherent light, our process performs much better than the Gerchberg–Saxton algorithm. We expect that these results indicate promise for a successful experimental demonstration of our process.

Phase imaging is important for many applications where the user needs to observe properties of a transparent object. Current methods such as the Gerchberg–Saxton algorithm or phase contrast microscopy either require coherent light or do not produce a direct phase image. Our method opens up the possibility of reconstructing a direct phase image under conditions of incoherent light, which can be very helpful in everyday usage.

Chapter 2

Theory

In this chapter we will introduce the fundamental principles of wave optics. First we will derive the wave equation from Maxwell's equations in Section 2.1. We will then introduce the phase retrieval problem in Section 2.2 and introduce notation in the context of our wave equations. Using this notation, we will derive in Section 2.3 the Huygens–Fresnel equation, the Fresnel diffraction equation, and the quadratic phase factor of a lens due to the paraxial approximation, all of which are standard tools in wave optics. Using these tools, we will show that the right lens placed at the correct location performs a spatial Fourier transform from an input to an output plane. Afterwards, we will present in Section 2.4 the Gerchberg–Saxton algorithm for phase retrieval, which we will use as a standard against which we will compare our process. Lastly, we will discuss in Section 2.5 the concept of incoherent light and how we will test both the Gerchberg–Saxton and our process using partially coherent light.

2.1 Wave Equation

This is the derivation from Griffith's *Introduction to Electrodynamics*[12] of the wave equation of light. We start with Maxwell's equations in a vacuum with no charge or current:

$$\nabla \cdot \boldsymbol{E} = 0; \tag{2.1}$$

$$\nabla \cdot \boldsymbol{B} = 0; \tag{2.2}$$

$$\nabla \times \boldsymbol{E} = -\frac{\partial \boldsymbol{B}}{\partial t}; \tag{2.3}$$

$$\nabla \times \boldsymbol{B} = \mu_0 \epsilon_0 \frac{\partial \boldsymbol{E}}{\partial t}. \tag{2.4}$$

Applying a curl to equation (2.3), we have that

$$\nabla \times (\nabla \times \boldsymbol{E}) = -\frac{\partial}{\partial t}(\nabla \times \boldsymbol{B}); \tag{2.5}$$

$$\nabla(\nabla \cdot \boldsymbol{E}) - \nabla^2 \boldsymbol{E} = -\frac{\partial}{\partial t}(\nabla \times \boldsymbol{B}). \tag{2.6}$$

Substituting equations (2.4) and (2.1) into equation (2.7),

$$\nabla(0) - \nabla^2 \boldsymbol{E} = -\mu_0 \epsilon_0 \frac{\partial^2}{\partial t^2} \boldsymbol{E}; \tag{2.7}$$

$$\nabla^2 \boldsymbol{E} = \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \boldsymbol{E}. \tag{2.8}$$

where we let c be the speed of light in a vacuum, defined as the velocity of this wave. A similar analysis on the equivalent equations for $\boldsymbol{B}$ would produce the same wave equation for the B-field. Because these are coupled equations of divergence-free fields, $\boldsymbol{E}$ and $\boldsymbol{B}$ must be transverse waves.

In a vacuum, the x, y, and z components of $\boldsymbol{E}$ and $\boldsymbol{B}$ would each satisfy the scalar wave equation,[1]

$$\nabla^2 u(P,t) = \frac{1}{c^2} \frac{\partial^2}{\partial t^2} u(P,t). \tag{2.9}$$

[1] In reality, the wave is a transverse wave with no field component along the direction of propagation. This statement is still true, however, because 0 satisfies the equation.

2.2 Phase Retrieval

Let us consider a coherent monochromatic plane wave. Without loss of generality, let us assume that the direction of propagation of light is in the z direction. We can use the complex-valued function $u(P,t)$ to denote the E-field perturbation at position P at some time t and in some polarization (either x or y). To get back the actual physical field, we would just take the real part of our function u.[2] For a coherent monochromatic wave, we can use a complex phasor $U(P)$, which represents the amplitude and phase as a function of position, so that $u(P,t) = U(P)\exp(-j\omega t)$. This allows us to simplify the math further without having to ever deal with the constant-frequency time-varying sinusoid.

Consider a thin object A that lives in the x,y plane. When the thin object is placed in the path of light, the effect of the object is to apply a multiplication to the light field. This can be described by a transmittance function, $t_A(x,y)$, which describes the multiplicative change that the light undergoes as it passes through the object. When light with a complex field of $u_l(x,y)$ passes through the object, it exits with a complex field of $u_{l'}(x,y) = t_A(x,y)u_l(x,y)$.

A phase-only object is an object A that has no absorption, i.e. $|t_A| = 1$. We will let $\phi_A(x,y) = -\arg(t_A(x,y))$, the wave retardation or phase of the object. Since we can only measure[3] the intensity $I(x,y) = |u(x,y)|^2$ at each point, it can be difficult to reconstruct the function $\phi_A(x,y)$. For our purposes, phase retrieval is the process of recovering the phase image, $\phi_A(x,y)$, at each location (or pixel) of a phase-only object.[3]

2.2.1 Tools available

To solve this phase retrieval problem, we have lenses, phase spatial light modulators (SLMs), and cameras. A camera allows us to measure the intensity of light at each pixel on a plane. As we will soon show, a lens is a tool that essentially enables us to take the spatial Fourier transform of the field. We can do this by placing the lens a focal length after the input field, and measuring the output field at a focal length after the lens. If we place two lenses in our path such that the first lens is

[2] The complex value is just meant to simplify the math. In case there is confusion, note that it is not meant to represent two dimensions, so the imaginary axis of u for the x-direction of the E-field *does not refer to the E-field in the y direction.* Instead, we can just apply the same analysis for both polarizations of light and add them up.

[3] Again, this is based on the assumption in the footnote in the introduction, that our detector integrates the light over many periods.

a focal length after the previous object, the second lens is two focal lengths after the first lens, and the next object (or the detector) is a focal length after the second lens, then we effectively perform two Fourier transforms and get back the original image, up to a constant phase difference.

A phase SLM is a pixelated device that can be inserted in the optical path and can generate an arbitrary transmittance function t_A on a plane. This transmittance is phase-only, and the phase values of our SLM span a range that is greater than 2π. The transmittance can be individually configured at each pixel by connecting the SLM to a computer and using it as a monitor.

2.3 Wave-Optics Analysis

In this section we will show how to compute the propagation of light using Fourier optics. We will start with the wave equations derived from Maxwell's equations earlier, and we will proceed to derive the Huygens–Fresnel principle,

2.3.1 Huygens–Fresnel Principle

This is the derivation of the Huygens–Fresnel principle from Goodman's *Introduction to Fourier Optics*[13] of the Rayleigh–Sommerfeld formulation of diffraction, which predicts the light field at one point in space by summing the aggregate contributions from different points in an aperture. This equation will be used in the next section (Section 2.3.2) to derive Fresnel diffraction.

We start with our definition of the complex phasor U, and substitute it into the wave equation:

$$u(P,t) = U(P)\exp(-j\omega t); \tag{2.10}$$

$$\nabla^2 u(P,t) = \frac{1}{c^2}\frac{\partial^2}{\partial t^2}u(P,t), \tag{2.11}$$

$$\nabla^2 U(P)\exp(-j\omega t) = -\frac{\omega^2}{c^2}U(P)\exp(-j\omega t), \tag{2.12}$$

$$0 = (\nabla^2 + k^2)U(P), \tag{2.13}$$

where k is the wave number ω/c. This is the Helmholtz equation.

In addition, we will use Green's theorem, stated in the following way: let $U(P)$ and $G(P)$ be two complex-valued functions of position and V be a closed volume

in space. If U and G are continuously second-differentiable, then we have

$$\int_V (U\nabla^2 G - G\nabla^2 U)\,dv = \int_{\partial V} \left(U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n} \right) ds, \tag{2.14}$$

where dv is a differential volume unit, ds is a differential surface unit, ∂V is the enclosing surface of V, and $\partial/\partial n$ is a partial derivative in the outward normal direction on the surface, away from V.

Suppose we want to find the complex phasor $U(P_0)$ at point P_0. We can let $U(P)$ from the theorem be our complex phasor $U(P)$, and we can choose any function G that satisfies the requirements of the theorem. We will call this the Green's function of the problem. Kirkhoff chooses $G(P_1)$ to be a spherical wave expanding from P_0. This would become

$$G(P_1) = \frac{\exp(jkr_{01})}{r_{01}}. \tag{2.15}$$

This has a singularity at $P_1 = P_0$; however, we can pick our volume V to be a large region of space surrounding the point P_0, but excluding a small region of diameter ϵ that contains P_0. We note that in this region, G also satisfies the Helmholtz equation,

$$(\nabla^2 + k^2)G = 0. \tag{2.16}$$

This means that we can simplify the left-hand side (LHS) of Green's theorem to zero:

$$\int_V (U\nabla^2 G - G\nabla^2 U)\,dv = -\int_V (UGk^2 - GUk^2)\,dv = 0. \tag{2.17}$$

Let S be the outer boundary of the region and S_ϵ be the inner boundary of the region V. Since the LHS of Green's theorem is zero, we can set the right-hand side (RHS) equal to zero:

$$0 = \int_{\partial V} \left(U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n} \right) ds; \tag{2.18}$$

$$0 = \int_{S_\epsilon} \left(U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n} \right) ds + \int_S \left(U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n} \right) ds. \tag{2.19}$$

The partial derivative is

$$\frac{\partial G(P_1)}{\partial n} = \cos(\boldsymbol{n}, \boldsymbol{r}_{01})\left(jk - \frac{1}{r_{01}}\right)\frac{\exp(jkr_{01})}{r_{01}}, \tag{2.20}$$

where $\cos(\boldsymbol{n}, \boldsymbol{r}_{01})$ is the cosine of the angle between $\boldsymbol{n}$ and $\boldsymbol{r}_{01}$.[13]

If we choose S_ϵ to be a sphere with radius $\epsilon/2$, the cosine is -1. In addition, as $\epsilon \to 0$ and using the fact that U is continuous at P_0, the integral over S_ϵ converges to $4\pi U(P_0)$.[4] Substituting into equation (2.17), we have that

$$U(P_0) = \frac{1}{4\pi}\int_S \left(\frac{\partial U}{\partial n}G - U\frac{\partial G}{\partial n}\right)ds. \tag{2.21}$$

Consider a function f that both satisfies the Helmholtz equation over V and is continuously differentiable at P_0. We can choose $G' = G - f$ as another Green's function, and this Green's function would also give us the same result for $U(P_0)$ in equation (2.21). This is because the LHS of Green's theorem would remain zero since f satisfies the Helmholtz equation over V, and the integral over S_ϵ of the f component of G' would converge to zero because f is continuously differentiable and the integration domain approaches zero as $\epsilon \to 0$. As a result, equation (2.21) remains identical, except with G' instead of G.

This can help us solve another problem: suppose we have light coming through an aperture E in a plane. We want to measure the phasor $U(P_0)$ at P_0. We can now choose $f(P_1)$ as another spherical wave, with the center P_2 at the reflection of P_0 across the plane. Then we can choose a region enclosed by a boundary that has a planar portion (S_1) on the plane that contains E and spreads out as a sphere (S_2) of radius R away from P_0. Since the region is on P_0's side of the plane that contains E, P_2 is not in this region, so f satisfies the conditions we just gave. Therefore we can let $G_- = G - f$ be our Green's function, which is

$$G_-(P_1) = \frac{\exp(jkr_{01})}{r_{01}} - \frac{\exp(jkr_{21})}{r_{21}}. \tag{2.22}$$

This function vanishes on the plane that E resides on. To match with our physical problem U must also vanish in the region of S_1 that is outside aperture E. This means that the integral vanishes for the region of S_1 outside of E, since U and G_- are both zero in this region. Secondly, if the field U satisfies the Sommerfeld radiation condition, which states that U vanishes at least as fast as a spherical wave,[13]

[4] See Goodman page 42[13] for a line by line evaluation of this limit.

the integral in the region S_2 will also vanish as $R \to \infty$. This leaves us with only an integral over the aperture E remaining:

$$U(P_0) = \frac{1}{4\pi} \int_E \left(\frac{\partial U}{\partial n} G_- - U \frac{\partial G_-}{\partial n} \right) ds \tag{2.23}$$

$$= -\frac{1}{4\pi} \int_E U \frac{\partial G_-}{\partial n} \, ds. \tag{2.24}$$

This is the *first Rayleigh–Sommerfeld solution*.[13]

The corresponding partial derivative of G_-, for $P_1 \in E$, is

$$\frac{\partial G_-(P_1)}{\partial n} = 2\cos(\boldsymbol{n}, \boldsymbol{r}_{01}) \left(jk - \frac{1}{r_{01}} \right) \frac{\exp(jkr_{01})}{r_{01}}. \tag{2.25}$$

When $r_{01} \gg \lambda$, the jk term would be much greater than $\frac{1}{r_{01}}$. Hence we can drop the $\frac{1}{r_{01}}$ term, leaving

$$\frac{\partial G_-(P_1)}{\partial n} = 2jk\cos(\boldsymbol{n}, \boldsymbol{r}_{01}) \frac{\exp(jkr_{01})}{r_{01}}. \tag{2.26}$$

Substituting into the first Rayleigh–Sommerfeld solution (Equation (2.24)), we have the Huygens–Fresnel principle:

$$U(P_0) = \frac{k}{2\pi j} \int_E U(P_1) \cos\theta \frac{\exp(jkr_{01})}{r_{01}} \, ds, \tag{2.27}$$

where θ is the angle between the normal to surface E facing away from P_0 and the vector from P_0 to P_1, the point on E.

This equation says that the points on a wavefront can be treated as new point sources, with a directivity pattern $\cos\theta$ and complex amplitudes proportional to the phasor value $U(P_1)$ and the wave number k.

2.3.2 Fresnel Diffraction Equation

The Fresnel Diffraction Equation is an equation that approximates the propagation of a light wave from one (x, y) plane to another in the z direction. To derive this, we will need to use the Fresnel approximation, which is accurate for any propagation distance that is much longer than the wavelength of light.[5]

[5]Although the Fresnel approximation at first glance seems to be only accurate within a region where the displacement in (x, y) is much smaller than the displacement in z, the integral over these

First we will present the explicit form of this equation: consider a source plane and a destination plane, both parallel to each other with a separation distance of z. Let (ξ, η) be coordinates within the source plane. The diffracting aperture or any interesting objects lie in the (ξ, η) plane. Let (x, y) be coordinates within the destination plane, where we make our measurement. We will let $U(x, y)$ refer to the complex phasor at the point (x, y). In terms of an actual $\boldsymbol{E}$ field, the real part of U would represent the phasor in one direction of polarization (e.g. E_x), and the same exact equation can be separately applied to all directions of polarization. The equation is then: ([13])

$$U(x,y) = \frac{e^{jkz}}{j\lambda z} e^{j\frac{k}{2z}(x^2+y^2)} \int\left\{U(\xi,\eta)e^{j\frac{k}{2z}(\xi^2+\eta^2)}\right\} e^{-j\frac{2\pi}{\lambda z}(x\xi+y\eta)}\, d\xi\, d\eta. \tag{2.28}$$

To derive this equation, we can start with the Huygens–Fresnel equation, and convert it to rectangular coordinates:

$$U(P_0) = \frac{1}{j\lambda}\int U(P_1)\frac{\exp(jkr_{01})}{r_{01}}\cos\theta\, ds. \tag{2.29}$$

Substituting $P_0 = (x, y)$, $P_1 = (\xi, \eta)$, and $\cos\theta = z/r_{01}$,

$$U(x,y) = \frac{z}{j\lambda}\int U(\xi,\eta)\frac{\exp(jkr_{01})}{r_{01}^2}\, d\xi\, d\eta, \tag{2.30}$$

where

$$r_{01} = \sqrt{z^2 + (x-\xi)^2 + (y-\eta)^2}. \tag{2.31}$$

To perform the Fresnel approximation, we do a binomial expansion of r_{01}/z, such that we keep as many terms as needed so that there is one surviving term that varies in (x, y). In other words, we let

$$b = \left(\frac{x-\xi}{z}\right)^2 + \left(\frac{y-\eta}{z}\right)^2, \tag{2.32}$$

approximations will allow the error to cancel out for large (x, y) so that the approximation will be accurate even for (x, y) comparable or larger than z. See Goodman pages 68-72[13] for a more detailed discussion.

and assume that $b \ll 1$ to approximate

$$r_{01} = z\sqrt{1+b} \tag{2.33}$$

$$\approx z\left(1+\frac{1}{2}b\right), \text{ or} \tag{2.34}$$

$$\approx z, \tag{2.35}$$

where the first approximation is used for the r_{01} within the exponential and the second is used for the r_{01} in the denominator.

Substituting our approximations into equation (2.30), we have that ([13])

$$U(x,y) = \frac{1}{j\lambda z}\int U(\xi,\eta)\exp\left\{j\frac{k}{2z}[2z^2+(x-\xi)^2+(y-\eta)^2]\right\}d\xi\,d\eta \tag{2.36}$$

$$= \frac{\exp(jkz)}{j\lambda z}\int U(\xi,\eta)\exp\left\{j\frac{k}{2z}[(x-\xi)^2+(y-\eta)^2]\right\}d\xi\,d\eta \tag{2.37}$$

$$= \frac{e^{jkz}}{j\lambda z}e^{j\frac{k}{2z}(x^2+y^2)}\int U(\xi,\eta)\exp\left[j\frac{k}{2z}(\xi^2+\eta^2)\right]e^{-j\frac{k}{z}(x\xi+y\eta)}\,d\xi\,d\eta. \tag{2.38}$$

This equation is now identical to equation (2.28). We can also see that the integral is a Fourier transform of the product of the field in the original plane and a quadratic phase factor. So essentially, Fresnel diffraction is a multiplication by a quadratic phase factor, followed by a Fourier transform, followed by another multiplication by a quadratic phase factor and some constants.

Although we assumed that $b \ll 1$, it turns out the integral integrates out the error well enough that this leading-term approximation is accurate almost everywhere, as long as the planes are separated by many wavelengths apart ($z \gg r_01$). See Goodman pages 68 to 72 for a more detailed explanation of the behavior of this approximation.[13]

2.3.3 Quadratic Phase Factor of a Lens

A lens is defined as a *thin lens* if rays leaving the lens experience negligible translation from their positions of entrance. For a thin lens, the total phase delay experienced by a ray entering at coordinates (x,y) is proportional to the thickness of the lens, assuming it is made of a uniform material. We can let the thickness be $\Delta(x,y)$. We will now derive a quadratic approximation for this phase delay.[13]

We can approximate a lens as being composed of three portions, which are a left-facing slice of a sphere ($\Delta_1(x,y)$), a central rectangular slice ($\Delta_2(x,y)$), and a

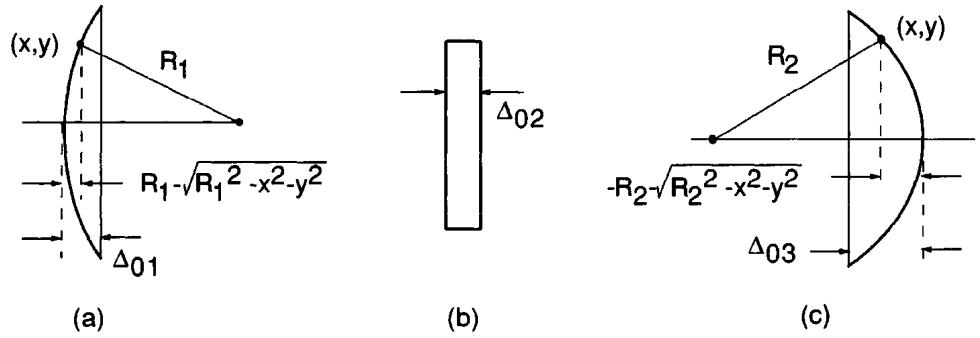


Figure 2.1: Adapted from the illustration on page 99 of Goodman[13] for the calculation of the thickness function. (a) illustrates how to calculate Δ_1, (b) illustrates how to calculate Δ_2, and (c) illustrates how to calculate Δ_3.

right-facing slice of a sphere ($\Delta_3(x,y)$). Using the right triangles from the diagram in Figure 2.1,[13]

$$\Delta_1(x,y) = \Delta_{01} - \left(R_1 - \sqrt{R_1^2 - x^2 - y^2}\right) \tag{2.39}$$

$$= \Delta_{01} - R_1\left(1 - \sqrt{1 - \frac{x^2+y^2}{R_1^2}}\right); \tag{2.40}$$

$$\Delta_2(x,y) = \Delta_{02}; \tag{2.41}$$

$$\Delta_3(x,y) = \Delta_{03} + R_2\left(1 - \sqrt{1 - \frac{x^2+y^2}{R_2^2}}\right). \tag{2.42}$$

Summing the three thicknesses,

$$\Delta(x,y) = \Delta_0 - R_1\left(1 - \sqrt{1 - \frac{x^2+y^2}{R_1^2}}\right) + R_2\left(1 - \sqrt{1 - \frac{x^2+y^2}{R_2^2}}\right), \tag{2.43}$$

where Δ_0 is a constant, equal to the sum of the constants above.

To derive the quadratic phase factor, we make the paraxial approximation, which is a binomial expansion of the value under the square root, $\sqrt{1-x} \approx 1 + \frac{1}{2}x$ for small x. For this approximation to be accurate, the x, y values of interest must be much smaller than R, which is on the order of the focal length. Applying the

approximation,

$$\Delta(x,y) = \Delta_0 - \frac{x^2+y^2}{2}\left(\frac{1}{R_1} - \frac{1}{R_2}\right). \tag{2.44}$$

The total phase delay, as a function of position, for a lens with a material that has an index of refraction n, is

$$\phi(x,y) = kn\Delta(x,y) + k[\Delta_0 - \Delta(x,y)], \tag{2.45}$$

therefore, the lens transmittance is

$$t_L(x,y) = \exp[jk\Delta_0]\exp[jk(n-1)\Delta(x,y)]. \tag{2.46}$$

Furthermore, the definition of focal length is

$$\frac{1}{f} = (n-1)\left(\frac{1}{R_1} - \frac{1}{R_2}\right). \tag{2.47}$$

Substituting in $\Delta(x,y)$ into equation (2.46),

$$t_L(x,y) = \exp[jkn\Delta_0]\exp\left[-j\frac{k}{2f}(x^2+y^2)\right]. \tag{2.48}$$

This equation is a quadratic phase factor that the lens applies, in the paraxial equation, to the field. In the future, when we use equation (2.48), we will leave out the constant phase factor, since it is applied equally to all (x,y) in this region.

2.3.4 Fourier Transforms from Lenses

The quadratic phase factor enables lenses to essentially perform 2D Fourier transforms. Intuitively, this is because the quadratic phase factor from the lens can cancel the quadratic phase factor in the Fresnel equation (2.28), allowing for an unmodified Fourier transform. We will consider the example, as illustrated in Figure 2.2, of a collimated plane wave incident on an object (with transmittance t_O) placed a distance f in front of a lens (on the *focal plane*). In this example, the camera is placed a distance f behind the lens (on the *back focal plane*). In fact, this is the only optical setup that we will use in our process.

Let A be the amplitude of the plane wave. Let $U_O(x,y), U_L(x,y)$ represent the light transmitted by the input object and the light incident on the lens, respec-

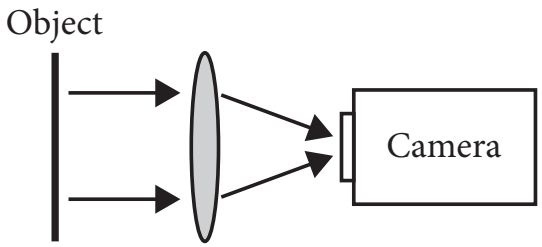


Figure 2.2: This is an example setup with an object a distance f in front of a lens with focal length f and a camera a distance f behind the lens.

tively, and let $F_O(f_X, f_Y), F_L(f_X, f_Y)$ represent the Fourier spectrums of U_O and U_L, respectively.[13] With this representation, $U_O = At_O$,

$$F_O(f_X, f_Y) = \mathcal{F}\{At_O\}, \text{ and} \tag{2.49}$$

$$F_L(f_X, f_Y) = \mathcal{F}\{U_L\}. \tag{2.50}$$

Using the form of the Fresnel equation listed in equation (2.37),

$$U_L(x,y) = \frac{\exp(jkz)}{j\lambda z} \int U_O(\xi,\eta) \exp\left\{ j\frac{k}{2z}[(x-\xi)^2 + (y-\eta)^2] \right\} d\xi\, d\eta, \tag{2.51}$$

$$U_L(x,y) = \int U_O(\xi,\eta) h(x-\xi, y-\eta)\, d\xi\, d\eta, \tag{2.52}$$

where

$$h(x,y) = \frac{\exp(jkz)}{j\lambda z} \exp\left[j\frac{k}{2z}(x^2+y^2) \right]. \tag{2.53}$$

The convolution theorem states that the Fourier transform of the convolution of two functions f and g is the product of the Fourier transforms of the functions, or in other words, ([13])

$$\mathcal{F}(f * g) = \mathcal{F}(f)\mathcal{F}(g). \tag{2.54}$$

Applying this theorem to equation (2.52),

$$F_L(f_X, f_Y) = F_O(f_X, f_Y) \exp[-j\pi\lambda f(f_X^2 + f_Y^2)], \tag{2.55}$$

where we dropped the constant phase factor in front.[13]

When the (x, y) values we are concerned with are smaller than the lens, the light

distribution after the lens, $U_{L'}$, is

$$U_{L'} = U_L \exp\left[-j\frac{k}{2f}(x^2+y^2)\right]. \tag{2.56}$$

Using the Fresnel equation from equation (2.28) to propagate this light by a focal length, the final light distribution, U_f, is

$$U_f(x,y) = \frac{e^{jkf}}{j\lambda z} e^{j\frac{k}{2f}(x^2+y^2)} \int U_L(\xi,\eta) e^{-j\frac{2\pi}{\lambda f}(x\xi+y\eta)}\, d\xi\, d\eta \tag{2.57}$$

$$= e^{j\frac{k}{2f}(x^2+y^2)} F_L(\frac{x}{\lambda f}, \frac{y}{\lambda f}), \tag{2.58}$$

where we dropped the constant phase factor in front, and the quadratic phase factors conveniently cancel. Substituting in equation (2.55), we have that

$$U_f(x,y) = F_O(\frac{x}{\lambda f}, \frac{y}{\lambda f}) \tag{2.59}$$

$$= \int t_A(\xi,\eta) \exp\left[-j\frac{2\pi}{\lambda f}(x\xi+y\eta)\right] d\xi\, d\eta. \tag{2.60}$$

This means that the light field at the back focal plane is a Fourier transform of the transmittance of the object, with distances scaled by a factor of k/f.

2.4 Linear Gerchberg–Saxton—A Phase Retrieval Algorithm

One of the existing algorithms for phase retrieval is the Gerchberg–Saxton (GS) algorithm. In this section, we introduce the GS algorithm because we will be comparing the results of our process with the results of the GS algorithm. This is a purely numerical algorithm that is designed to reconstruct the phase from two intensity images, provided that the optical transformation in between the systems is linear and known. The transformation is usually a Fourier transform, which represents propagation through a lens in the setup described above. We would measure the intensity image before the transformation, $|U_O|^2$, and the intensity image after the transformation, $|U_f|^2$. (See Figure 2.3 for an illustration of this iterative algorithm.) The algorithm consists of the following four simple steps:[5]

1. Fourier transform an estimate, $\hat{U}_O$ of the initial light to get an estimate, $\hat{U}_f$,

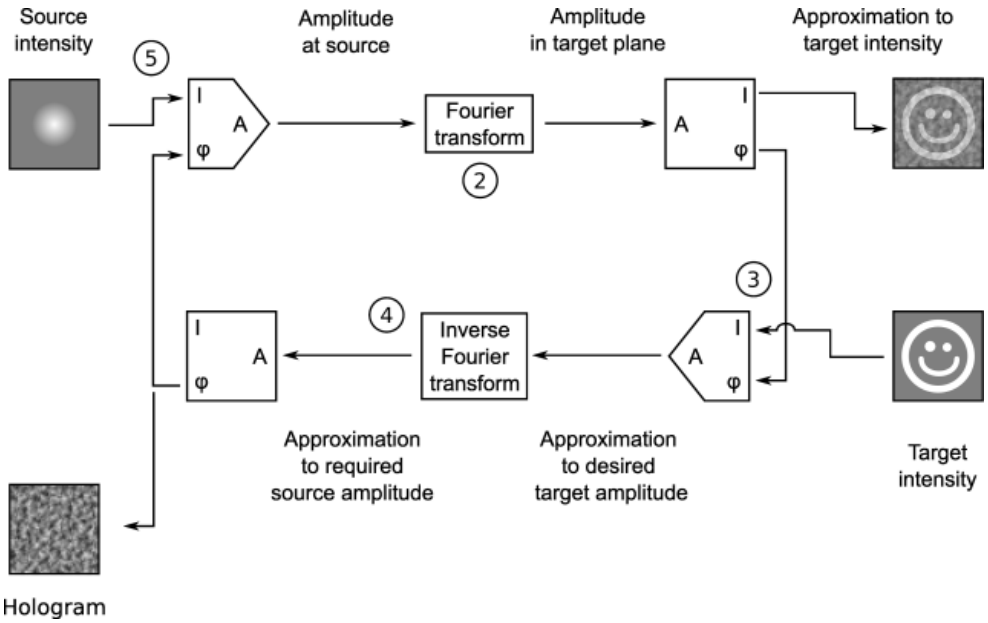


Figure 2.3: This illustration of the Gerchberg–Saxton algorithm, is adapted from the Wikipedia page for the Gerchberg–Saxton.[14] The four-step process approximates the phase φ of the object using the source and the target intensity images.

of the final light.

2. Replace the modulus of the resulting estimate of the final light, $\hat{U}_f$ with the measured modulus, $|U_f|$.

3. Inverse Fourier transform the modified estimate of the final light, $\hat{U}_f$, to get an estimate, $\hat{U}_O$, of the initial light.

4. Replace the modulus of the resulting estimate of the initial light, $\hat{U}_O$ with the measured modulus, $|U_O|$.

These steps would repeat until the difference in $\hat{U}_O$ between two consecutive iterations is small enough. The modulus of $\hat{U}_O$ is the measured $|U_O|$, and the argument of the resulting $\hat{U}_O$ is the final phase estimate.

This algorithm can easily be generalized to a larger class of algorithms, where we replace the Fourier transform ($\mathcal{F}$) with any other linear transformation on the wave. For example, we could put in Fresnel propagation as the transformation instead of the Fourier transform, and obtain our two intensity images simply by moving the camera along the path rather than inserting a lens. As long as the transformation is linear, the algorithm is considered a linear GS algorithm.

Although there are some known convergence issues in certain cases, these issues have been studied extensively and mostly solved by many researchers including Fienup and Wackerman 1986.[15] Our implementation of the GS algorithm takes advantage of the methods described in [15] to overcome some of these modes of stagnation. This ensures a fair comparison against the best GS algorithm for this purpose.

However, even with these improvements, our simulations show that the GS algorithm only converges for propagation under coherent and almost-coherent light. Because the phase is not well-defined over a displacement in time or space, no pair of functions exist that can both describe the propagation forward and the propagation backwards of incoherent light. Therefore when incoherent light is used, the Gerchberg–Saxton fails to converge.

2.5 Incoherent Light

The theory discussed so far assumes that the light we have is coherent. When the phase relationships between different parts of a light field become statistically unrelated, the light is said to be incoherent. To be precise, light is coherent if its behavior over time can be represented in the form

$$u(P,t) = U_t(P)\exp(-j\omega t) \tag{2.61}$$

for some real-valued frequency ω and phasor $U_t(P)$ that varies slowly and predictably over time. If this is true, then the phase relationship between two different positions P_1 and P_2 in space can be described by the relationship between the complex phases of $U_t(P_1)$ and $U_t(P_2)$.

To describe incoherent light, we can first rewrite the polychromatic light $u(P,t)$ as the following product:

$$u(P,t) = U(P,t)\exp(-j\overline{\omega}t), \tag{2.62}$$

where $\overline{\omega}$ is the mean frequency of the optical wave.[13] Then $U(P,t)$ is similar to a phasor, except it is time-varying. We can now introduce the concept of a mutual intensity, which is the time-averaged covariance between U at two locations:

$$J(P_1,P_2) = \langle U(P_1,t)U^*(P_2,t)\rangle_t, \tag{2.63}$$

where $\langle\cdot\rangle_t$ denotes a time average. This mutual intensity is a measure of the "spatial coherence"[6] of the light between two points. It is useful because most detectors are much slower than the frequency of light and therefore integrate the effect of light over time.

When light is completely incoherent, the covariance is zero whenever $P_1 \neq P_2$, and nonzero when $P_1 = P_2$. When light is completely coherent, the phasors do not vary over time, and so J is simply the product of the two phasors, $U(P_1)U^*(P_2)$.

We will now show, based on Goodman's derivations on pages 131 to 135,[13] that when light is perfectly coherent, we have an imaging system that is linear in *complex amplitude*, and when light is perfectly incoherent, we have an imaging system that is linear in *intensity*.

Suppose the propagation of light can be described by a transfer function $h(u, v)$ such that the propagated light U_f is the convolution of h and the initial light U_i:

$$U_f(x, y, t) = \int h(x - \xi, y - xi) U_i(\xi, \eta, t)\, d\xi\, d\eta. \tag{2.64}$$

We note that the intensity, I_f, is often the squared time-average of the magnitude of the phasor, since detectors are a lot slower than frequencies. In other words,

$$\begin{aligned} I_f(P) &= \langle |U_f(P,t)|^2 \rangle_t && (2.65)\\ &= \int dP_1\, dP_2 h(P - P_1) h^*(P - P_1) \langle U_i(P_1, t) U_i^*(P_2, t) \rangle_t && (2.66)\\ &= \int dP_1\, dP_2 h(P - P_1) h^*(P - P_1) J_i(P_1, P_2). && (2.67) \end{aligned}$$

When U_i is completely coherent, as we noted, $J = U_i(P_1)U_i^*(P_2)$. This means that

$$\begin{aligned} I_f(P) &= \int dP_1\, dP_2 h(P - P_1) h^*(P - P_1) U_i(P_1) U_i^*(P_2) && (2.68)\\ &= \left| \int dP_1 h(P - P_1) U_i(P_1) \right|^2 = |U_f(P)|^2. && (2.69) \end{aligned}$$

The last integral is a summation on the complex amplitude U_i, which means that this system is linear in the complex amplitude U_i. On the other hand, when U_i is completely incoherent, we can write $J = CI_i(P_1)\delta(P_1 - P_2)$[7], where C is some constant

[6] See Goodman page 131-135[13] for a good definition of this term.

[7] The δ is not exact, as the correlation length of J cannot get smaller than a wavelength. See [13] for further explanation.

and I_i is the squared magnitude of U_i. Substituting into equation (2.67),

$$I_f(P) = C \int dP_1\, dP_2 h(P - P_1) h^*(P - P_1) I_i(P_1) \delta(P_1 - P_2) \tag{2.70}$$

$$= C \int dP_1 \left| h(P - P_1) \right|^2 I_i(P_1). \tag{2.71}$$

This last integral is a summation on the intensity I_i, meaning that this system is linear in the intensity I_i. Intuitively, the cross terms of the intensity calculation averaged out to zero, which resulted in only the individual intensities mattering in this sum.

Besides completely coherent and completely incoherent light, there is partially coherent light, which is light that is coherent at a close distance but incoherent at a large distance. The spatial correlation is defined as

$$G(P_1, P_2) = \frac{|J(P_1, P_2)|}{|\langle U(P_1) \rangle \langle U(P_2) \rangle|}.$$

Based on our definition of J earlier, it is equal to 1 on the diagonal entries. On the off-diagonal entries, the correlation is equal to 1 if the light is coherent or 0 if the light is incoherent. However, if the light is partially coherent, the correlation can be neither equal to 1 nor 0 on the off-diagonal entries. In fact, the correlation $G(P_1, P_2)$ falls off as P_1 gets farther away from P_2, and oftentimes it is somewhat exponential, so that we can define a correlation length as the characteristic length of the decay. This correlation length describes as the speckle radius r_{speckle}. Speckles are regions where the phase is very correlated, and this radius describes the size of speckles that can form in the illumination. A large speckle size indicates very coherent light, while a small speckle size indicates almost-incoherent light.

Lastly, for incoherent light, it is no longer appropriate to talk about the phase of the light after passing through an object. Instead, we use the term "wave retardation," which measures the path length (multiplied by the wave number) that light experiences as it passes through the object at different locations. This term has the same meaning as the object phase for coherent light, but it is more appropriate to use wave retardation when we discuss incoherent light.

We have briefly introduced incoherent light in terms of mutual intensity and spatial correlation. More detailed discussions of the optics of incoherent light and imaging can be found in references such as [16], [17], and [18].

Chapter 3

Methodology: Our Phase-Retrieval Process

In this chapter we will describe our process, which retrieves the phase of a pure-phase object by estimating an illumination that conjugates the phase of the object. This problem requires us to work within a high dimensional space (for example, guessing 10 values each for 60,000 pixels would require on the order of $10^{60,000}$ guesses). To reduce the number of total guesses required, we can break this process up into separately guessing values within each dimension, one dimension at a time. The basic idea is to guess linearly-independent patterns (which we will refer to as "basis elements") whose combination, in the correct mixture (which we will refer to as the "coefficients"), would accurately describe the shape of the object. With each pattern, we would use the output of an optical process to determine how good the guess is.

Our description of our process assumes a real laboratory environment, up until Section 3.5, at which point we will provide details for our simulation. We will first start by describing in Section 3.1 the optical setup that will enable our process. We will then discuss in Section 3.2 the mathematics that describe this setup and use it in Section 3.3 to motivate the decisions we made in creating our process. Afterwards, we will explicitly describe the procedure for our process in Section 3.4 and lastly detail the simulation process in Section 3.5.

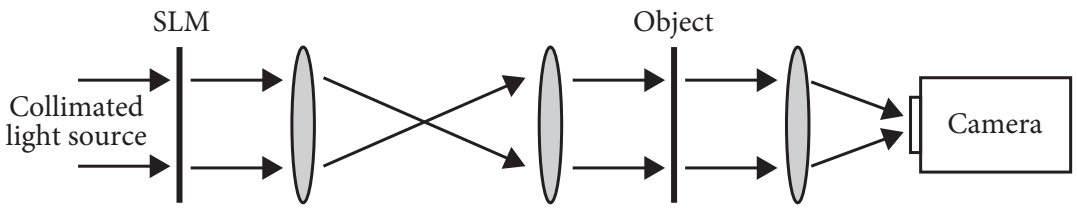


Figure 3.1: This is an illustration of our optical setup, as adapted from our submission to the COSI conference 2015.[19] A collimated quasi-monochromatic light source is sent onto a spatial light modulator (SLM). The surface of the SLM is imaged on the object using a $4f$ relay and the light is focused on the camera with a third lens. The phase pattern on the SLM is optimized in order to compensate (conjugate) the object phase.

3.1 Optical Setup

The optical principle of this method is depicted in Figure 3.1. We start by sending a collimated beam of monochromatic light onto a phase-only spatial light modulator (SLM). The SLM will allow us to pattern our incoming illumination. After the SLM, we send the patterned illumination into a $4f$ lens relay, which reproduces the same illumination onto a plane displaced by four focal lengths. We then place a slide with the object on the other end of the $4f$ lens relay. After the light passes through the object, we send it into another lens placed a focal distance away, with our camera at the opposite focus of this lens. In this setup, the light essentially gets modified by the SLM followed by the object, and then gets propagated through a lens. If the SLM pattern perfectly conjugates the object, then the light after transmission through the SLM and the object would have a flat wavefront. This light would then focus through the lens into a single spot on the camera.

Note that the setup from the object plane to the camera is the same setup as the setup for the Gerchberg–Saxton method. Since the camera can only measure the intensity, it cannot directly give us information about the phase of the object. Phase retrieval processes, a category to which our process belongs, are processes designed to use this intensity image to help recover the phase of the object.

3.2 Wave-Optical Formulation of the Setup

First we note that the $4f$ system only contributes a constant phase difference, assuming that the lenses are identical. This is because, notwithstanding constant phase factors, each system of a propagation by f, a lens, and another propagation

by f, combine into a Fourier transform. The composition of two Fourier transforms simply negates the original field, which is essentially a constant phase factor of π. Disregarding this constant phase difference, the $4f$ system leaves the field unchanged, and we can treat the object and the SLM as being on the same plane.

Let ϕ_A be the wave retardation, or phase, of the object, ϕ_{SLM} be the phase of the SLM, and f be the focal length of the last lens. Then

$$t_A = \exp(-j\phi_A), \tag{3.1}$$

$$t_{\text{SLM}} = \exp(-j\phi_{\text{SLM}}), \tag{3.2}$$

and let

$$t(\xi, \eta) = t_A(\xi, \eta) t_{\text{SLM}}(\xi, \eta), \tag{3.3}$$

the combined transmittance of the SLM and the object.

Then the complex light field at the camera, $u(x, y)$, is described by

$$u(x, y) = \frac{A}{j\lambda f} \int t(\xi, \eta) \exp\left[-j\frac{2\pi}{\lambda f}(\xi x + \eta y)\right] d\xi \, d\eta. \tag{3.4}$$

Our setup allows us to adjust ϕ_{SLM} and measure the intensity, $I(x, y) = |u(x, y)|^2$. In addition, we are able to remove the object and put in an empty slide, which we will represent by $t_0(\xi, \eta) = \exp(0) = 1$. This is the diffraction from an object with a constant phase of zero.

3.2.1 Locations of different Fourier modes in the output image

For practical measurements, we need to determine the physical location that corresponds to each Fourier mode $\exp(j(k_x x + k_y y))$. Consider the diagram in Figure 3.2. Let $\boldsymbol{r}$ refer to the 2D coordinate (x, y) in the camera plane. Light can be rewritten as a superposition of plane waves that point at different directions. For the light departing from the center of the lens arriving at location $\boldsymbol{r}$, the ray has an angle of θ with respect to the normal. This means that

$$k_r = k \sin\theta, \tag{3.5}$$

$$k = \frac{2\pi}{\lambda} \tag{3.6}$$

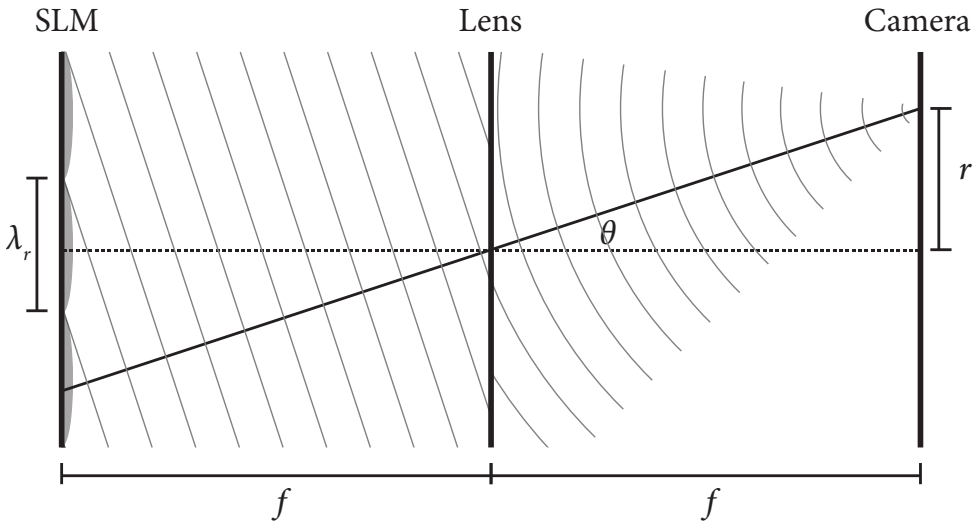


Figure 3.2: This diagram illustrates the scaling between the Fourier mode and the location of the focused spot on the back focal plane of the lens. It shows both how $k_r = k \sin\theta$ and $k_r = 2\pi/\lambda_r$. The lines indicate wavefronts of constant phase and are separated by a wavelength. λ_r is the distance between adjacent minimums of the SLM modulation, and it is also equal to the distance on the SLM between adjacent wavefronts of the light ray.

where k_r is the component in the XY plane of the wave vector $\boldsymbol{k}$. According to the diagram, $\sin\theta = r/f$, where $r = \sqrt{x^2 + y^2}$ and f is the focal length of the lens. In addition, this light ray is the direction of the plane wave before the lens, so k_r also represents the wave number for the spatial oscillations in the SLM plane. As a result, $k_r = 2\pi/\lambda_r$, where λ_r is the spatial wavelength in direction $\boldsymbol{r}$ in the SLM plane. This means that

$$k_r = \frac{2\pi r}{\lambda f} = \frac{2\pi}{\lambda_r}. \tag{3.7}$$

The lowest-order integer Fourier mode on the SLM plane[1] would occur at a λ_r equal to the size of the SLM, and the finest Fourier mode on the SLM would occur when λ_r becomes the distance between SLM pixels. To find the mapping between a pixel in the camera image and the amplitude of a coefficient of Fourier mode, we would simply evaluate

$$r = \frac{\lambda f}{\lambda_r}, \tag{3.8}$$

[1] An integer Fourier mode on the SLM refers to a mode that oscillates along the length of the SLM an integer number of times, or where $k_r = \frac{2\pi n}{L}$ where L is the width of the SLM and n is an integer.

plugging in the values for the lowest mode: the focal length for f, the SLM width L for λ_r, and the wavelength for λ. This r would correspond to the distance between two consecutive integer Fourier modes on the SLM. This would be useful later on, when we choose a basis that closely corresponds to the integer Fourier modes on the SLM.

3.3 Mathematical Overview of Wavefront Flattening

In this section we will provide a mathematical description of our problem and the motivation behind our process. The explicit instructions, which will be standalone, will be described in Section 3.4.

We want to find ϕ_{obj}, the phase of the object. To do this, we can measure a reference of a known phase and use it to measure the accuracy of our estimates. Since we know the phase of the empty slide, we can use the empty slide as our reference. We measure the reference by putting in the empty slide t_0 and using a constant default phase on the SLM. Let L be the width of the SLM, and assume the SLM is square. Our illumination after the SLM would therefore be a square function with width L and value 1 inside the square, which can also be written as

$$u_{\text{SLM}}(\xi,\eta) = \text{rect}\left(\frac{\xi}{L},\frac{\eta}{L}\right). \tag{3.9}$$

The complex light field at the camera is then described by

$$u_{\text{ref}}(x,y) = \frac{A}{j\lambda f}\int_{|\xi|,|\eta|\le L/2} \exp\left[-j\frac{2\pi}{\lambda f}(\xi x+\eta y)\right]d\xi\, d\eta. \tag{3.10}$$

This Fourier transform is simply

$$u_{\text{ref}}(x,y) = \text{sinc}\left(\frac{\pi L}{\lambda f}x,\frac{\pi L}{\lambda f}y\right). \tag{3.11}$$

Let $I_{\text{ref}}(x,y) = |u_{\text{ref}}(x,y)|^2$. This is the intensity output that we observe at the camera for this reference setup.

Now that we know what a reference setup looks like, we can reinsert the object so that the phase of the object becomes ϕ_{obj}. If we can then find a SLM phase, ϕ_{SLM}, so that the intensity output (I) exactly resembles I_{ref}, then we know that

$$t(\xi,\eta) = 1, \tag{3.12}$$

where t is the combined transmittance of the SLM and the object, and we leave out any constant phase factors. We can say $t(\xi,\eta) = 1$ because in our optical setup, the flat wave with no phase variation is the unique field that can produce the intensity output I_{ref}. The reasoning is that a flat wave with no phase variation coming from the SLM will be Fourier transformed by the lens into a focus of a small fixed size inversely proportional to the size of the SLM aperture. Even though only the intensity can be measured, the phase within the focus is known to be constant. A perturbation in the input field with a size smaller than the SLM will produce a perturbation in the Fourier transform that is larger than the focus. On the other hand, no perturbation to the input field can be larger than the SLM, so therefore, no modulations to the output field can be smaller than the focus. Hence, a plane wave is the only field that can possibly produce a spot in the focal plane of the lens as small as this focus. I_{ref} is thus uniquely produced by a plane wave. If the field is disturbed by the object, the only way to restore the plane wave is to compensate the object by its conjugated phase function.

If we achieve $t(\xi,\eta) = 1$, then

$$1 = t(\xi,\eta) = t_{\text{SLM}}(\xi,\eta)t_{\text{obj}}(\xi,\eta) \tag{3.13}$$

$$= \exp[-j(\phi_{\text{SLM}}(\xi,\eta) + \phi_{\text{obj}})]. \tag{3.14}$$

This can be satisfied if

$$\phi_{\text{SLM}}(\xi,\eta) = -\phi_{\text{obj}}(\xi,\eta), \tag{3.15}$$

Therefore the correct ϕ_{SLM} value is $-\phi_{\text{obj}}$ for flattening the wavefront, and once we find it, we can simply negate the phase to image the object. For notation purposes, we will use ϕ_{SLM} to refer to the correct SLM phase that flattens the wavefront and $\hat{\phi}_{\text{SLM}}$ to refer to the final estimate of ϕ_{SLM} that our algorithm can converge to.

To estimate the correct ϕ_{SLM} value that makes the intensity output I resemble I_{ref}, we let $\hat{\phi}_k(\xi,\eta)$ be the k-th estimate that we place on the SLM, and $I_k(x,y)$ be the intensity output from setting the SLM modulation to $\hat{\phi}_k(\xi,\eta)$. As k gets large, $\hat{\phi}_k(\xi,\eta)$ converges to $\hat{\phi}_{\text{SLM}}(\xi,\eta)$, our final estimate. We then attempt to minimize the error E_k, which we will define as how distant I_k is from I_{ref}. Our error metric

is the square of the Euclidean norm of the difference,

$$E_k = \|I_k(x,y) - I_{\text{ref}}(x,y)\|_2^2 \tag{3.16}$$

$$= \int |I_k(x,y) - I_{\text{ref}}|^2 \, dx \, dy \tag{3.17}$$

$$\approx \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} |I_k(x_i,y_j) - I_{\text{ref}}(x_i,y_j)|^2, \tag{3.18}$$

where the x_i, y_j values are pixels on a rectangular grid, and N_x and N_y are the dimensions of the grid. Furthermore, E_k is an error value that we can compute from real physical measurements.[2] If $E_k = 0$, $I_k = I_{\text{ref}}$, and as we just stated above, this implies that $\hat{\phi}_k = \phi_{\text{SLM}} = -\phi_{\text{obj}}$.

The error E_k gives us an approximation of the distance between u_k and u_{ref}, and we attempt to iteratively guess $\hat{\phi}_k$ such that E_k decreases as k increases. The goal is for $\hat{\phi}_k$ to converge to ϕ_{SLM}, up to a constant phase difference.

We will now treat $\hat{\phi}_k$ as a discrete function that takes on values for input from $\{1,...,N_x\}\times\{1,...,N_y\}$, giving us a search space of $\mathbb{R}^{N_x N_y}$. Since the search space has a high number of dimensions, we can attempt to simplify this problem by searching one dimension at a time. We define a set of images, $\boldsymbol{M} = \{V_i\}, V_i \in \mathbb{R}^{N_x,N_y}$, to be a complete basis if the images in $\{V_i\}$ are linearly independent and $\text{span}(\boldsymbol{M}) = \mathbb{R}^{N_x N_y}$.

Based on this notation, given a complete orthogonal basis $\boldsymbol{M}$, we can represent any image $\hat{\phi}_k$ as a linear combination of basis elements. In other words, we can create a coordinate system $(a_1, a_2, ..., a_{N_x N_y})$ made of the scalar coefficients of each basis element so that

$$\hat{\phi}_k = \sum_{V_i \in \boldsymbol{M}} a_i V_i. \tag{3.19}$$

One difficulty is that the transformation from the input (object and SLM) phase ϕ to the intensity output I is nonlinear. As a result, we cannot simply solve this problem by separately minimizing E_k for every coefficient a_i and combining the results. Instead, we need to incrementally combine the estimates of previous coefficients when we optimize a new coefficient, and we also need to carefully choose the order in which we optimize the coefficients. In addition, we must also readjust

[2] While the error metric we would actually want can be better represented by $E_{\text{ideal}} = \|\phi_{\text{SLM}} - \hat{\phi}_{\text{SLM}}\|$, we use the difference in intensity in our optimization process because intensity can be directly observed in real physical measurements, while ϕ cannot.

coefficients that we have already optimized after optimizing other coefficients. Intuitively, the coefficients that yield the largest improvement in E_k should be the ones optimized first, since these would bring us closest to our desired result. In addition, not all basis elements should be necessary to express ϕ_{SLM} to a reasonable accuracy, so if we reduce the number of basis elements kept at the beginning, we would greatly improve our performance.

To be able to select the right basis elements, we can choose a basis whose coefficients can be approximated by the output image I_0, which is the intensity output of the object alone with no modulation from the SLM. This intensity output is the squared modulus of the Fourier transform of the object field $u_{\text{obj}}(\xi, \eta)$. Our basis, however, needs to be a decomposition of the possible values of the phase, $\phi(\xi, \eta)$, rather than the field, $u(\xi, \eta)$. We will show that these two are closely related: consider a phase function ϕ with relatively small variation,

$$\begin{aligned} \phi(\xi, \eta) &= \phi_0 + \delta(\xi, \eta); \\ u(\xi, \eta) &= C \exp(j\delta(\xi, \eta)) \\ &\approx C + jC\delta(\xi, \eta), \end{aligned}$$

where C is a complex number with modulus 1. So when the variation in phase is small, the variation in the field is a constant phase shift times the variation in phase. Therefore, for all Fourier modes other than the constant ($\boldsymbol{k} = 0$) mode, the magnitude of the Fourier transform of the field (u) is a good approximation of the magnitude of the Fourier transform of the phase (ϕ). While our phase is not actually small, this argument gives us an approximation for the right basis elements to use; later on in Section 4.3 we will mention that our process is actually extremely robust with regards to which exact basis elements are chosen.

This result means that we can use the information from the output image to estimate the magnitude of the values of the discrete Fourier transform of ϕ_{SLM}. With the correct scaling, each pixel in the output $I(x, y)$ can be mapped to the magnitude of the coefficient of a basis element of the discrete Fourier transform, $\exp(jk_\xi \xi)$ for some k_ξ. The scaling factor is defined by the relationship $r = \frac{k_r \lambda f}{2\pi}$ (where k_r refers to the k_ξ we need and r refers to the location) derived in Section 3.2.1. To take advantage of the fact that our output image encodes the amplitudes of the discrete Fourier transform of ϕ_{SLM}, we can choose a basis that closely resembles the discrete Fourier transform, which is the discrete cosine transform (DCT). We choose the discrete cosine transform because the phase, $\phi(\xi, \eta)$, is a real-valued function,

and using this basis naturally causes the coefficients of the cosine transform to also be real. We would then choose our subset of basis elements by looking at which pixels of the $I(x, y)$ output have the largest values. For our procedure, we can limit the optimization space to the set of N basis elements that correspond to the brightest N pixels of the output.[3] We can also arrange our process so that we estimate the brightest of this set first and revisit them the greatest number of times, because the largest coefficients make the biggest difference in reconstructing our image.

3.4 Instructions for Our Phase-Retrieval Process

In the previous section we just provided a roadmap for how we want to design our process. In this section, we will provide the specific details for implementing the process. We will estimate the object phase, ϕ_{obj}, by using the intensity image, $I_k(x, y)$, to help choose the SLM modulation to be as close to $-\phi_{\text{obj}}$ as possible. After the process converges, we would give the negated image of our SLM guess, $-\hat{\phi}_{\text{SLM}}(\xi, \eta)$, to the end user as our estimate for the object phase. While a few details from the last section are repeated here, they are included for completeness so that this procedure is standalone.

The first step is to replace the object with a transparent slide, place no modulation on the SLM, and record the intensity pattern on the camera. Since we are focusing flat light with a lens, the pattern should be simply a single bright spot on the camera. This pattern serves as the reference pattern and is the I_{ref} we described above. After recording this reference pattern, we then insert the object onto the slide and attempt to pattern the phase of the SLM in such a way that it compensates for the phase of the object. If the phase pattern on the SLM is the perfectly conjugated version of the sample phase, then the resulting light beam after transmission through the SLM and the object would be flat, and the reference pattern would be recovered on the detector. Therefore our process now, with the object in the slide, is to try to optimize the SLM pattern so that the intensity output reproduces the reference pattern. See Figure 3.3 for an illustration of our process.

The optimization of the SLM phase is performed by decomposing the phase pattern into basis functions and searching for the coefficients that minimize the difference between the detected intensity and the reference pattern. The complete basis decomposition that we use to optimize the phase image is the discrete cosine

[3] This is assuming each pixel is separated by the distance r as calculated from section 3.2.1. If they are not separated by this exact amount, we would rescale the output image first.

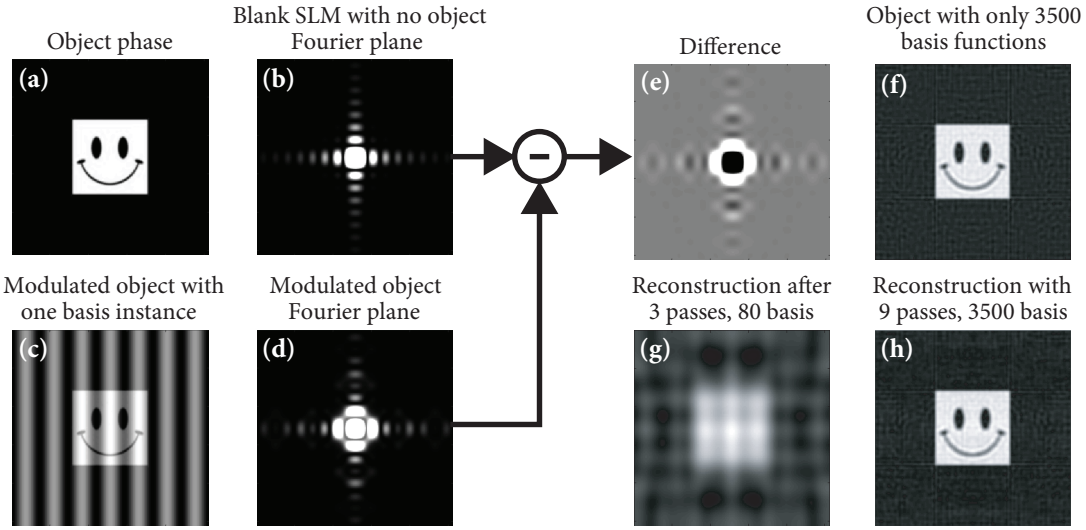


Figure 3.3: An illustration of the steps of our process, adapted from our COSI submission.[19] (a) Phase of the object. The gray scale represents the phase retardation, black being 0 rad and white being 1.7 rad. The image corresponds to the SLM area (square), which is illuminated with a uniform intensity. (b) Intensity detected on the camera from a blank SLM with no object. The scale has been adjusted to make the side lobes appear clearly. This pattern is the reference intensity I_{ref}. (c) Object phase modulated by one instance of a cosine transform basis element. (d) Intensity detected on the camera with the object in place and the modulation on the SLM. (e) Intensity patterns from (b) and (d) are subtracted. The optimization is performed on frame (e) and minimizes the norm. (f) Object projected onto the corresponding 3,500 basis elements with the knowledge of the true coefficients. This represents the best possible reconstruction with 3,500 basis elements and serves as a comparison standard. (g) Reconstructed object after performing 3 passes, optimizing the coefficients of 20, 40, and then 80 basis elements. The correlation of this reconstruction with the object is 0.879. (h) Reconstructed object after performing 9 passes, with each pass incrementally doubling the number of coefficients optimized until it iterates all 3,500 basis functions. The correlation with the object is 0.994.

transform (DCT). The optimization procedure is represented in the pseudocode in Algorithm 1.

1. We first place a constant modulation of zero phase on the SLM, and record the intensity output with only the object. We will refer to this intensity output as I_0. Since the field at the detector is a Fourier transform of the light field at the object plane, we can use the intensity output to approximate the discrete cosine spectrum of the phase image. This would work well for small-phased objects, since differences in field would approximate differences in phase.

Algorithm 1 Our Phase Retrieval Process

procedure OURPHASERETRIEVAL
    N $\leftarrow 6,500$ basis elements
    NPerCoeff $\leftarrow 33$
    NItersPerPass $\leftarrow 20$
    NPass $\leftarrow 9$

    Place the object in the object plane.
    SLM $\leftarrow$ The zero vector
    $I_0 \leftarrow$ Camera reading
    $I_{\text{basis estimate}} \leftarrow$ Rescale I_0 to correspond to basis elements of ϕ_A.
    BasisOrder $\leftarrow$ Sort pixels in $I_{\text{basis estimate}}$ by brightness
    CurrentGuessValues $\leftarrow$ The zero vector

    for *pass* from 1 to *NPass* **do**
        for i from 1 to *NItersPerPass* **do**
            WhichCoeff $\leftarrow$ *BasisOrder*$[i]$
            CurrentCoeff $\leftarrow$ *CurrentGuessValues*[*WhichCoeff*]
            GuessValues $\leftarrow \{\textit{CurrentCoeff}\} \cup \text{PickRandomValues}(\textit{NPerCoeff} - 1)$
            BestValue $\leftarrow \min_{v \in \textit{GuessValues}} E_k(v, \textit{CurrentGuessValues}, \textit{WhichCoeff})$
            CurrentGuessValues[*WhichCoeff*] $\leftarrow$ *BestValue*
        end for
        NItersPerPass $\leftarrow \min(N, 2\textit{NItersPerPass})$
    end for

    return ReconstructImage(*CurrentGuessValues*)
end procedure

2. We then pick N (which should be a number roughly between 1,000 to 6,500[4], depending on the desired final reconstruction quality) basis elements of the discrete cosine transform by mapping pixels from the initial output image I_0 using the $k_x = 2\pi x/\lambda f$ relationship derived in Section 3.2.1. We then take the N basis elements corresponding to the brightest pixels in I_0. The total number of basis elements required to render the object phase perfectly is equal to the number of pixel in the image, which in this case is 65,536 (256×256), but a smaller subset of these basis elements can provide a satisfactory result. The plan now is to progressively perform many iterations through all the basis elements to estimate their coefficients.

3. Sort the basis elements from the previous subset by order of brightness of the corresponding region in I_0. After determining the order, we perform multiple "passes" of optimizing the coefficients, with each pass iterating through all the previously-optimized coefficients and then adding more unoptimized coefficients. Let N_i be the number of elements that pass i will iterate through. We will start with $N_1 = 20$, and after a pass we will double N_i, and then double again every pass afterward, until N_i reaches N. Here is what each pass consists of:

   (a) For each basis element, we optimize the coefficient by first choosing 33 values within a range dictated by the image resolution, i.e. from $-\pi N_x$ to πN_x, N_x being the width of the image in pixels. 32 of these 33 values are chosen randomly, and the 33rd value is either 0 if it is the first pass that optimizes this coefficient, or the existing coefficient produced from the previous pass. For each value, c, we add the basis element multiplied by this value onto the SLM phase and measure the resulting intensity pattern I_k. (k is just an integer iterator that for the sake of notation distinguishes the current intensity from all other intensities recorded.) We then calculate the E_k-error score of this basis value, as described in the previous section (Section 3.3), by taking the Euclidean norm of the difference between I_k and I_{ref}.

   (b) Finally, we keep the c value that yields the lowest E_k as our coefficient for this basis element. For each subsequent coefficient, the previous coefficients are kept constant and remain on the SLM, so that the basis elements build incrementally on each other.

[4]These numbers were chosen as a balance between image quality and runtime. See Section 4.1.

(c) After optimizing the first N_i coefficients, we obtain a reconstruction. We call this operation a "pass". A more accurate reconstruction can be obtained by performing new passes using the same basis elements, each time keeping the previously estimated coefficients until they get updated in the new pass. After a fixed number of passes have occurred, set $N_{i+1} = 2N_i$. If N_{i+1} is larger than N, then we just set $N_{i+1} = N$.

4. We repeat the above steps until $N_i = N$, at which point we perform a fixed number of passes until the final reconstruction image, which is the negation of the phase modulation values on the SLM, is satisfactory. For our process, we found that 9 passes were satisfactory.

After our process is completed, we report $-\hat{\phi}_{\text{SLM}}$, or the negation of the last SLM modulation, as our resulting estimate of the phase image.

We will now argue that our process converges: after every time we perform step 3b, E_k can never increase, because we are taking the minimum of a set of values that includes the value that achieved the previous E_k. Since E_k is bounded below by 0, the sequence $\{E_k\}$ is a monotonically decreasing sequence with a lower bound, implying that this sequence converges (although not necessarily to 0). Our algorithm therefore converges, and we can stop at a point when E_k can no longer changes significantly. If this problem is well-enough behaved and we use enough basis elements, E_k hopefully converges to a value that is close to zero, so that our reconstruction accurately represents our object phase.

3.5 Simulation and Verification of Our Process

Before trying this process in the laboratory, we performed simulations in MATLAB to test and verify its effectiveness. In this section we will precisely describe how we simulated our process as described in Section 3.4 to test and verify its behavior. The results of these simulations are described in Section 4.1. The actual laboratory experiment is outside the scope of this thesis.

An example code for simulating the propagation of light through our system is in Appendix A. It first simulates the propagation in the z direction by a distance equal to the focal length. This propagation is performed using the beam propagation method (BPM)[20], which is a method that uses the slowly varying envelope approximation to propagate the light by taking a Fourier transform, multiplying by a propagation factor, and then taking an inverse Fourier transform. After the

propagation, the code multiplies the field by a quadratic phase factor calculated from the dimensions of the lens. After this phase factor, the code uses the BPM again to propagate the light by a further focal length. While this code can be generalized to setups with arbitrary propagation distances and lens sizes, our setup performs an exact Fourier transform, so we do not need the four Fourier transforms performed in this code. So to reduce the computation time, we simply used a single call to the MATLAB-provided fast Fourier transform (FFT) in most of our simulations. We found, by testing a few runs with both the simple FFT and the BPM, that the choice between the BPM code and the simple FFT makes a negligible difference in the output image.

We will first compute in Section 3.5.1 the locations in the camera output image that correspond to each basis element in our optimization method, and then we will in Section 3.5.2 compute the specific physical parameters our MATLAB FFT simulation would correspond to, so that we can accurately rescale the simulation to match the parameters of light in our lab.

3.5.1 Mapping the intensity output to basis elements

We will use the mapping derived in Section 3.2.1 to derive the location in the camera plane that describes the amplitude of each basis element. However, as a word of caution, the mapping is an approximation of where everything ought to be, as we are using a cosine basis rather than a Fourier basis, and our basis decomposes the phase, or the argument of the complex number, instead of the actual complex field. However, this approximation is good enough for deciding an ordering of which basis elements to optimize first.

If the image output of the camera has a pixel separation of δ, then we simply need to scale the image down by a factor of r/δ so that each pixel corresponds to a discrete Fourier mode. Plugging in the values for r, we have that

$$\text{Image scaling factor} = \frac{r}{\delta} = \frac{\lambda f}{N_x \delta_{\text{SLM}} \delta}, \tag{3.20}$$

where δ_{SLM} is the pixel separation of the SLM. This scaling factor would map pixels of the camera output to the exact cosine transform elements returned by MATLAB's

discrete cosine transform (DCT).[5] In our simulations, we chose

$$\begin{aligned} N_x &= 256, \\ \delta &= 8\ \mu\text{m}, \\ \delta_{\text{SLM}} &= 8\ \mu\text{m}, \\ \lambda &= 532\ \text{nm, and} \\ f &= 3\ \text{cm}, \end{aligned}$$

based on the physical dimensions of our lenses, SLMs, cameras, and lasers. This choice also allows the scaling factor between the separation between the output image pixels and the separation between discrete Fourier modes to be approximately one, which makes it easy for simulating the propagation. The actual values in the lab do not need to strictly be these values, since the only step at which we care about the scaling factor is the first time we choose our N basis elements based on the output image, and we can simply rescale this output image once to estimate the magnitudes of our discrete cosine modes.

Lastly, because the discrete cosine transform only has positive frequencies, we only needed the pixels in the lower-right quadrant of the propagated output image. Therefore for this mapping step, we pre-processed the image by setting the pixels in the upper-left, upper-right, and lower-left quadrants to zero, so that we only pick basis elements from the lower-right quadrant.

3.5.2 Propagating the Light through the System

We need a subroutine to simulate the propagation of light through the system so that we can simulate the intensity recorded by the camera. This simulation would be used whenever we need to record a camera image, such as when we record I_{ref}, or when we record I_k in order to compute E_k.

For coherent light, we simply compute the propagation by directly Fourier transforming the input field.

To decrease the boundary effects, we can increase the computational window size to beyond that of the SLM size, but our simulations show that this does not affect the convergence of the algorithm or the accuracy of the reconstructions. We have omitted the results for the larger computational size due to the larger runtimes needed to compute every reconstruction.

[5] Each reconstruction uses the exact command, `Image = idct2(ifftshift(Coefficients))`.

The fast Fourier transform (FFT) in MATLAB returns results such that $\Delta k = \frac{2\pi}{N\Delta x}$ in the resulting matrix, where Δk is the separation between adjacent values in the discrete Fourier matrix, and Δx is the spatial separation between adjacent values in the discrete input matrix. For an initial SLM with a distance of 8 microns between pixels, a camera with the same pixel size, a focal length of 3 cm, and light at 532 nm, the Fourier transform in MATLAB of a 256-pixel-wide image would correspond one-to-one between the pixels of the SLM and the pixels of the camera, as we listed at the end of the previous section (Section 3.5.1). The MATLAB Fourier transform on a 1024-pixel-wide SLM, which we actually have in the lab but will not use in our simulations, would correspond to a focal length of 12 cm. So once we move to the lab, since our SLM is bigger, we will be able to use lenses with longer focal lengths to confirm the results of the our simulations.

For incoherent or partially coherent light, we would generate the beam by first simulating light coming from point sources randomly chosen inside an aperture. We then Fourier transform this light, as if it were emitted onto a lens, and then use this Fourier transform as the illumination for our system. After propagating through the system, we would store the final intensity. Finally, we would sum up the intensities from each point source. Since each source emits coherent light, this is equivalent to summing up intensity outputs of coherent illuminations that are uncorrelated from each other. In our simulation, a tiny aperture corresponds to relatively coherent light, and a large aperture corresponds to a relatively incoherent light. Similar to the analysis in Section 3.2.1, if the aperture radius is r_{source}, then

$$\lambda_{\text{speckle}} = \frac{\lambda f}{r_{\text{source}}}. \tag{3.21}$$

This λ_{speckle} is the average separation between neighboring speckle peaks of the illumination right before the image plane. The correlation length, r_{speckle}, would correspond to the separation between the peak and the zero of the autocorrelation of the speckle distribution, which is approximately half the average separation between neighboring speckle peaks. We will use this approximation as the definition of r_{speckle}:

$$r_{\text{speckle}} = \frac{1}{2}\lambda_{\text{speckle}} \tag{3.22}$$

$$= \frac{\lambda f}{2r_{\text{source}}}. \tag{3.23}$$

We treat this value, r_{speckle}, as the speckle radius, which is the important distance in making our measurements. We define the speckle wave number, k_{speckle}, as $2\pi/r_{\text{speckle}} = 4\pi/\lambda_{\text{speckle}}$. These incoherent simulations are computed not only for a proof of concept, but also for a comparison with the Gerchberg–Saxton algorithm.

The point sources were chosen randomly. However, we noticed that for most simulations of 1,000 point sources or under, the fluctuations in the resulting illumination far exceed the fluctuations within the possible coefficients of one basis element, making it extremely difficult to compute useful E_k values to minimize. In the real experiment there will be a stable source of light, but in simulations we have to somehow maintain consistency for the propagations within the optimization. To do this, we randomize the locations of the point sources once, and then we reuse the same selection every time we need to propagate the light. As long as the point sources are dense enough, this simulation still accurately simulates incoherent propagation because we are still only adding up intensities.

Chapter 4

Simulation Results with Coherent Illumination

In this chapter we will discuss the results from simulating our process under coherent light. We will first present in Section 4.1 the reconstructions of various sample images and a discussion of the types of images that converge rapidly to a good quality. The reconstructions are in Table 4.1. We will also include a comparison against a modified version of our process with only one pass through all the basis elements in order to illustrate that it was necessary to perform multiple passes. We found that the one-pass runs peaked at 98% correlation, while the normal runs using our process could easily exceed 99% correlation as we increase the number of basis elements allotted.

After this section, we will analyze the runtime and convergence of the process in Section 4.2. We found that during each pass, the first half of the basis elements are necessary in order to improve upon the previous pass, but as we would expect, most of the improvement comes from the new basis elements added to the pass. Finally, we will discuss the errors and robustness of our algorithm in Section 4.3. We found that our algorithm very extremely robust to the exact basis elements chosen, the ordering of the basis elements, as well as the number of values chosen per coefficient.

4.1 Verification Metric and Results for Our Process

To verify the performance of our process, we used the correlation between the final guess and the original object. In addition to the correlation, we will also present the final 256×256 reconstructed images so that we can judge them by eye, as it is possible for images to have high correlations but miss obvious features such as the eyes and the mouth of a smiley face. We will present discussion on the effectiveness of our process with respect to the maximum number of basis elements allotted for a run. Afterwards, we will discuss and show that it was actually necessary for us to perform 9 passes by showing the results if we had only done one pass for each reconstruction. In addition, we will analyze the runtime data, and analyze the tradeoff between runtime and correlation.

In Table 4.1 we have a comparison of a few images and the best reconstructions we have achieved on these images, with each run limiting itself to iterate over 20, 75, 300, 1,000, 3,500, or 6,500 basis elements. Realistically, the number of basis elements needed for a reasonably clear image is somewhere between 1,000 and 3,500, as we can see from Table 4.1.

4.1.1 Reconstructed Images

Table 4.1 contains the final reconstructions using our process for nine different images, using six different numbers (N) of basis elements. In each original image, the color white indicates a phase of 1.82 (or 1.7 for Image I) and the color black indicates a phase of 0. The colors in our reconstruction images are scaled automatically so that we can visualize the range of phase values. Even though the scales are different, they are still very close to the scale of the original.

Each run uses 9 passes. The first pass goes through the coefficients of the first 20 basis elements, the next goes through the coefficients of the first 40, doubling with each pass until it reaches the total number of basis elements we allot. The ninth pass always does all of the allotted coefficients. So the $N = 20$ runs do a total of 180 iterations, where each "iteration" refers to guessing 33 possible values (chosen using stratified random sampling) for a basis element and choosing the one that gives the lowest E_k. The $N = 75$, $N = 300$, $N = 1{,}000$, $N = 3{,}500$, and $N = 6{,}500$ runs do a total of 585, 1,800, 4,260, 8,600, and 11,600 iterations, respectively.

Images A, C, F, H, and I explore the use of relatively narrow and long features, such as with the letter 'A', a very thin curve, the letter 'F', the phrase "It's not about

you." printed in a smaller font, or the mouth of a smiley face. Images B, D, and G explore the use of various rounded features, and Images E and I explore the use of square features. Images B, H, and I explore the use of complex multi-layered features. In images with rounded features (B, D, and G), 1,000 basis elements seem sufficient to produce a good, clear image, while in images with more square elements, the edges do not become clear until 3,500 basis elements. Lastly, most of these images need about 6,500 basis elements for the background to be almost completely smooth as well. So if only the content is important, it would suffice to use only 1,000 to 3,500 basis elements, which is 1.5% to 5.3% of the number of independent dimensions,[1] while if the background needs to be clear too, we would recommend using 6,500 basis elements, or 9.9% of the number of independent dimensions. These percentages are all under 10%, which means that our process is able to compress most of the information about the image into under 10% of the basis elements.

To check that it was actually necessary to perform multiple passes, we reran the algorithm doing only one pass in each run. The number of iterations in each run is now exactly equal to the number of basis elements allowed. Table 4.2 shows the results of these runs. While the general shapes are relatively good, both the foreground objects and the background are not as smooth as those produced using multiple passes. This difference is most pronounced in the runs with large N values, such as $N = 1{,}000$, $N = 3{,}500$, and $N = 6{,}500$. This is because there is only one chance to guess the values of the most important basis elements, and after later values are guessed, which would likely move the minimum for the first few basis elements, the algorithm never goes back to correct the earlier values, resulting in small artificial patterns appearing on the image. We will explain this effect further in the next section (Section 4.1.2), where we present correlation values.

[1] The image has 65,536 pixels total, corresponding to 65,536 independent dimensions.

Reconstructions using Various Numbers of Basis Elements (N),
Normal Procedure using 9 Passes

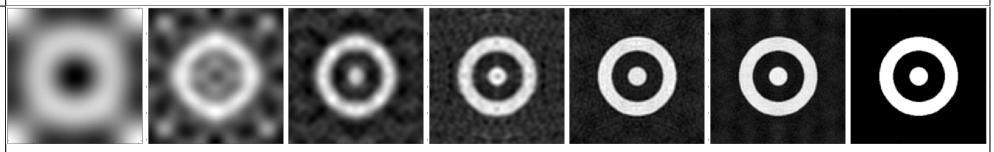
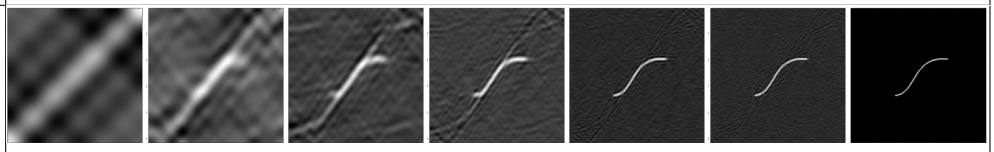
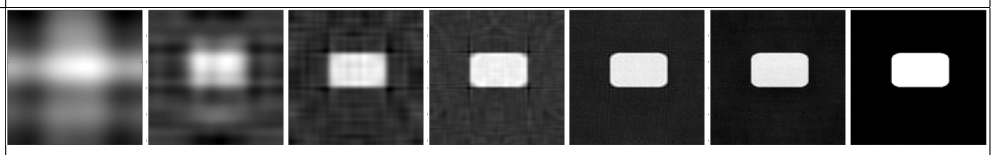
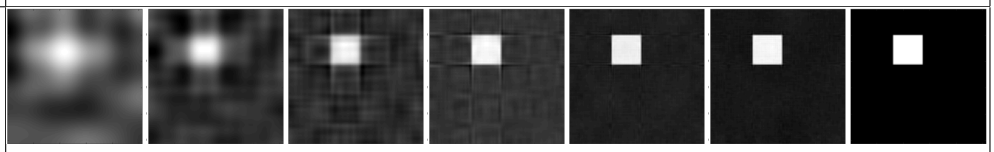
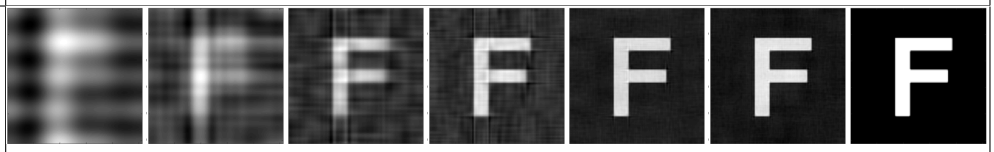
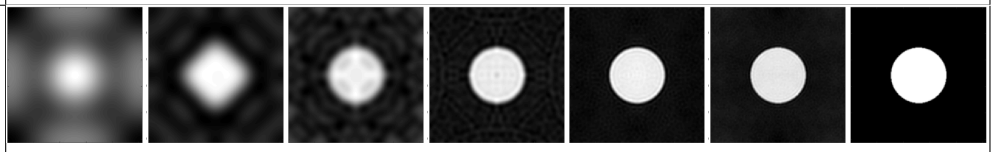
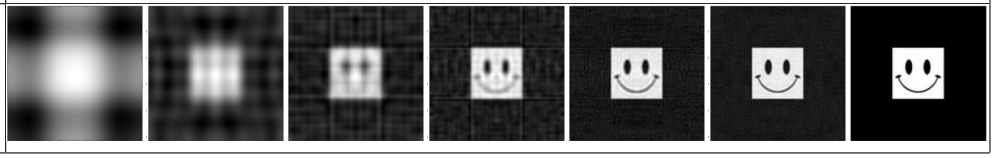
N	20	75	300	1,000	3,500	6,500	Original
Image A: Letter A							
Image B: Bullseye							
Image C: Curve							
Image D: Rounded Rectangle							
Image E: Square							
Image F: Letter F							
Image G: Circle							
Image H: Text							
Image I: Smiley Face							

Table 4.1: Each image is 256 pixels wide and 256 pixels tall. Each run uses 9 passes, and each pass checks 33 values per coefficient.

Reconstructions using Only One Pass

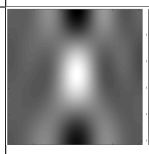
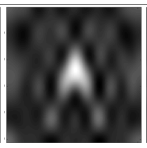
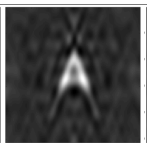
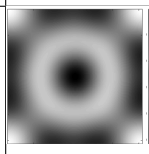
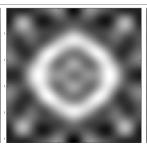
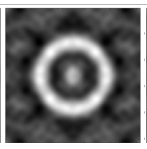
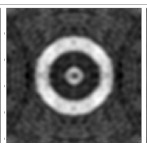
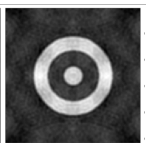
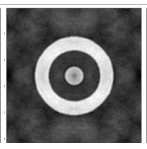
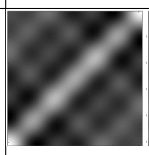
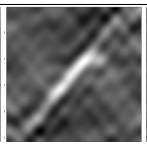
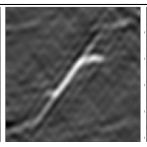
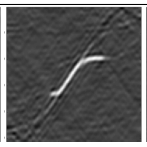
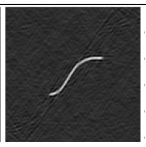
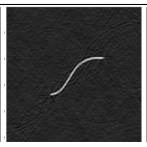
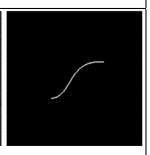
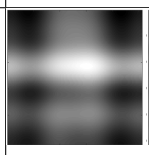
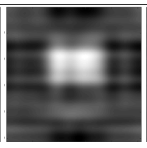
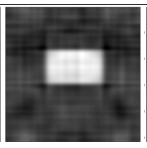
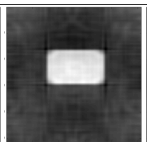
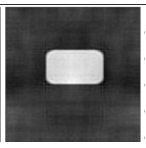
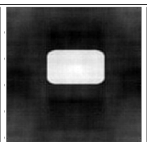
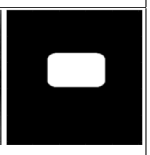
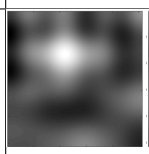
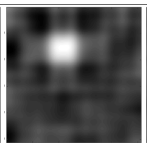
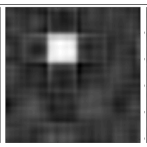
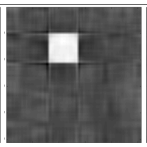
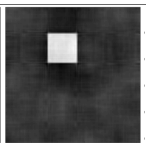
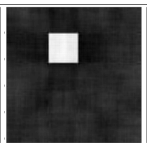
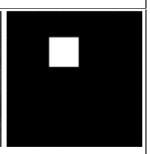
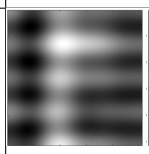
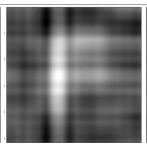
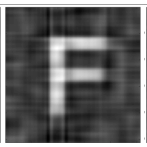
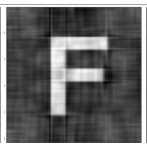
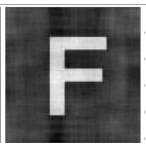
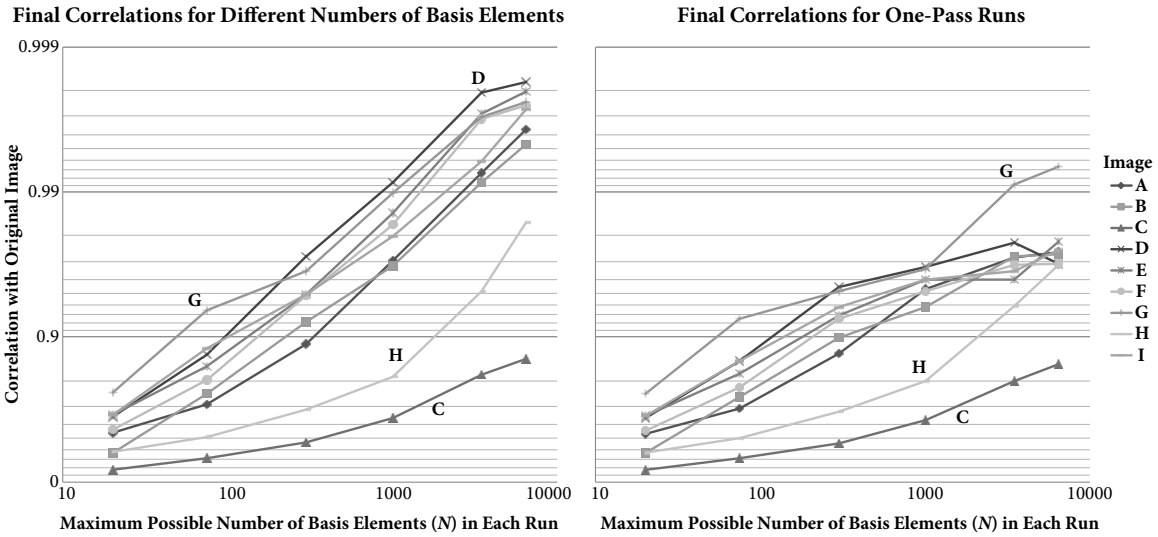
N	20	75	300	1,000	3,500	6,500	Original
Image A: Letter A							
Image B: Bullseye							
Image C: Curve							
Image D: Rounded Rectangle							
Image E: Square							
Image F: Letter F							
Image G: Circle							
Image H: Text							
Image I: Smiley Face							

Table 4.2: Each image is 256 pixels wide and 256 pixels tall. Each run uses 1 pass and checks 33 values per coefficient. This table shows that one pass is not sufficient to produce the quality of images in Table 4.1.

4.1.2 Correlation Values

To quantitatively assess our reconstructions, we calculated a correlation coefficient between the original image and each reconstruction. Figure 4.1a shows the correlation values for each run, using 9 passes. Images with thick features tend to have higher correlations than images with thin features. Reconstructions of Images C and H have lower correlations because they were made of only thin features, and our process appears to converge slower for thin features. In addition, the correlation function is more strict for thin features, since these features have to be in the exact right locations to improve the correlation. In all images, increasing the number of basis elements increases the final correlation, showing that our reconstruction does not deteriorate as we add more basis elements. The peak values are around 99.7% for most images (at 6,500 basis elements), although Images C and H have peak values of 85.8% and 98.4%, respectively.

To confirm that we actually needed to perform 9 passes instead of just performing a single pass through all the coefficients, we calculated the correlations from runs that only permitted one pass. Figure 4.1b shows these values, and they are roughly 2% lower than those of the normal runs, located in Figure 4.1a. These values show that by restricting to only one pass there is an artificial peak of about 98% correlation. This peak is likely here because the error in the first few basis elements can never get corrected. In other words, after guesses for later elements move the minimum, we never return to the first few basis elements to readjust our estimate. In addition, by only ever guessing 33 values for each of the first few basis elements, the likelihood of getting a very accurate result is low. In fact, the correlation drops by almost a constant 2% for all runs, regardless of the number of basis elements used. The correlation for Image G does not drop by a full 2% because it is a single round feature, which means that the first basis elements may not need as much re-optimization after the first pass. This peak at about 98% reduces the effectiveness of the procedure, because the peak implies that increasing the number of basis elements allotted can no longer do much to improve the reconstructions. This is the reason our standard procedure uses multiple (9) passes.



(a) (b)

Figure 4.1: These plots show the correlations between each reconstruction and the original image, with 20 to 6,500 basis elements allotted per run. This side-by-side comparison between normal (9 pass) and one-pass runs shows that our normal procedure is better than a procedure with only one pass. (a) illustrates the results from normal runs using 9 passes. The peaks are all at 6,500 basis elements, with correlations of 99.5% to 99.8%, with the exceptions of Image C and Image H, which were especially difficult to reconstruct. (b) illustrates the results from using only one run through all 6,500 basis elements. All the correlations except for Image G are about 2% lower than those of the normal run. As a result, there is an artificial peak at 98% correlation, which therefore reduces the effectiveness of using large numbers of basis elements.

4.2 Runtime Analysis and Correlation as a function of Iterations Performed

The runtime for the final process in the laboratory, which will take advantage of the SLM and the real-life optical system, will be proportional to the number of modulations we send to the SLM: we would perform a discrete cosine transform (takes $O(N_x N_y \log(N_x N_y))$ time) for each guess, and then send it to the SLM (whose time depends on the SLM response rate), capture an image with the camera, and then calculate the error E_k (takes $O(N_x N_y)$ time). After we perform the 17 to 33 guesses, we make a constant number of steps to store the answer, and then we continue to the next iteration. We do not count the time from simulating the propagation of light through the optical system because the real-life optical system is essentially instantaneous for lenses with focal lengths that fit in a lab.

The total number of iterations is 11,600 for 6,500 basis elements. With 33 guesses for each iteration, this adds up to 383,000 SLM modulations. If we simply save the E_k from the last iteration as the E_k associated with the value that was already there, then we would only need to perform 32 guesses for each iteration, or 371,000 SLM modulations.

Since we designed our process to double the number of iterations with each consecutive pass, an advantage is that it forms a geometric series, which when summed up is linear in terms of the number of basis elements allotted. To be precise, if the last c passes at the end proceed with the full number of coefficients, then our process would perform less than $(c+1)N$ iterations,[2] where N is the number of basis elements allotted. If we allow the process to run for p passes total, then

$$c = (p - \lfloor \log_2(N/20) \rfloor). \tag{4.1}$$

This allows for a very nice rule of thumb that results in a very good reconstruction (as shown above) as well as a linear runtime: simply set the number of passes to $\lfloor \log_2(N/10) \rfloor$, which will guarantee our algorithm to perform at most $2N$ iterations.

Figure 4.2 illustrates how the correlation relates to the number of SLM modulations as the run progresses, for a few representative runs. The curves take on a similar shape for all 54 runs. With each pass after the first pass, there is a shal-

[2] The +1 is from the sum of a geometric series with a ratio of $r = 2$ and a largest term under $N/2$. This sum is less than N.

low but rising slope for the first half while it is iterating the basis elements that have already been iterated. Then there is a sharp rise for the second half when it iterates through the new basis elements. It is also important that the curve in the second half of each pass is concave down, indicating that the basis that are iterated earlier are more important under our verification metric than those iterated later, and thus make a bigger difference in the correlation. This confirms that it was a good idea to choose the order of the basis elements based on the first output image with just the object.

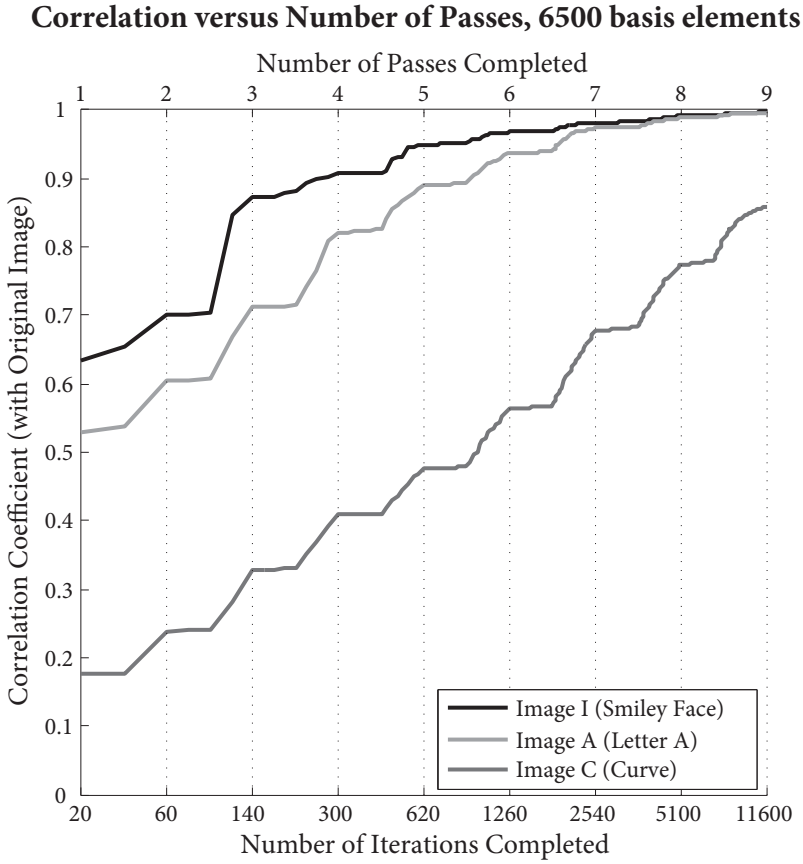


Figure 4.2: This plot shows the relationship between the correlations of each reconstruction with the original image versus the number of passes completed so far. The boundaries between consecutive passes are marked with dotted vertical lines. The black, dark silver, and gray lines show the correlation values for the run using 6,500 basis elements to reconstruct Image I, Image A, and Image C, respectively. The number of SLM modulations is 33 times the number of iterations.

4.3 Error and Robustness Analysis

Based on a few tests with our simulations, we found that our process is very robust to any errors or aberrations in the optical system, as long as that output is consistent. In other words, it only needs the lens setup to be *consistent*, not *accurate*. We found that if we perturb the simulation for the propagation a little bit, either by adding in randomness in the source illumination or by rescaling the camera output, we still converge to the best reconstruction. In fact, neither the choice of basis nor the ordering of the basis elements need to be perfect, as long as it roughly guesses the lower-frequency basis elements first. For example, we performed a few runs where we just selected the basis elements from I_{ref}, the output image of no object and no SLM (basically just the sinc() function), and we found that the final reconstructions were of similar quality to the same runs that used I_0, the output image of the object with no SLM modulation, to pick the basis elements.

In addition, there is also robustness in the values chosen for each coefficient: the values are randomized for each iteration and chosen over a large range, and as long as there are enough values guessed, the algorithm eventually finds an image that closely resembles the original. We found that decreasing the number of values per iteration from 33 to 17 or 9 also made very little impact on the final reconstruction quality.

Overall, we found that our process works quite well for coherent light, for a wide range of images. Only about 5 to 10 percent of the basis elements are needed to reconstruct a good-quality image, and this reconstruction takes about 5,100 to 11,600 iterations, or 163,000 to 371,000 SLM modulations. Each of these reconstructions reliably converges to over 99% correlation.

Chapter 5

Simulation Results with Incoherent Illumination

In this chapter we will discuss the results from simulating our process under incoherent light. We will first explain in Section 5.1 how we simulated the incoherent light, followed by a table of the results in Figure 5.1 showing that even as the light gets less coherent, the quality of the reconstructions remain the same.

We will then in Section 5.2 compare our process against the Gerchberg–Saxton algorithm. Figure 5.2 on page 53 compares the behavior of our process against the Gerchberg–Saxton algorithm, and it shows that our process significantly outperforms the Gerchberg–Saxton under partially coherent light. We found that the reconstructions from the Gerchberg–Saxton decrease in quality for incoherent source apertures with diameters greater than 30 μm, while our process continues to perform well for source apertures with diameters up to 150 μm.

Lastly, we will present in Section 5.3 some error analysis, mentioning that it was difficult to simulate incoherent propagation and that very large source apertures with diameters above 400 μm are untrustworthy.

5.1 Incoherent Light

To test our method on incoherent light, we alter our optical system in the following way:

1. Instead of a flat illumination, generate the illumination by randomly choosing a pixel (p) in a source aperture (E) with diameter (d) in the Fourier plane. Then take the Fourier transform of this pixel. Call this illumination u_{ill}.

2. Continue the simulation as usual, multiplying u_{ill} by the object image and the SLM image, and then taking the Fourier transform again to get the wavefield at the output plane.

3. Calculate the intensity (square of the modulus) of the image at the output plane. Call this intensity I_x.

4. Repeat the above steps, setting x to a different pixel each time, until illumination from all the pixels in the aperture E have been simulated. Keep a running sum of output intensities I_p, and return the output intensity, $I = \sum_{p \in E} I_p$.

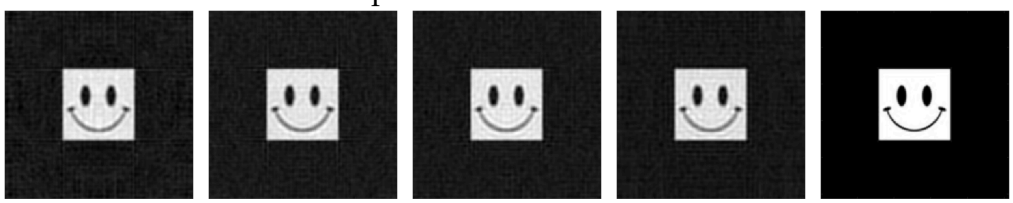
This is equivalent to simulating bright incoherent illumination from an aperture of diameter d, propagating the light from each illumination point separately, and then summing up the resulting fields from each point. Since there is no correlation between different illumination points, the final intensity is just the sum of the intensities from each point. Notice that in the case of $d = 1$ pixel, the illumination is flat, and we reproduce the same results as coherent light. However, as d increases, this is equivalent to simulating incoherent illumination from a larger aperture, and it simulates higher and higher incoherence across the illumination. The light would produce speckles of a smaller size as the source aperture gets larger.

Figure 5.1 illustrates the results of various incoherent simulations on Image I. Because the simulations are lengthy, we reduced the number of guesses per iteration from 33 to 17, and we only performed this simulation on Image I, which encompasses all the features in the other images (square, round, thin, and fat). As we can see, the difference in appearance between the coherent reconstruction and the incoherent reconstructions is far less noticeable than the difference between any of these reconstructions and the original. In other words, incoherence does not noticeably decrease the accuracy of the reconstructions.

To explore this further, we list the correlation values between each reconstruction and the original in Table 5.1. As we can see, most of the incoherent reconstructions actually exceed the coherent reconstruction in correlation. Therefore our process can be expected to work equally well under the conditions of incoherence in these simulations.

To better understand the size of these aperture sizes, we have listed the spatial wave numbers of the speckles k_{sp} in the last column of Table 5.1. We can see that the correlation does not decrease even when k_{sp} gets above 50,000 m^{-1} (As in, the speckles get smaller than a width of about 16 pixels wide, since higher k implies smaller speckle size), which is about the spatial frequency of many of the features in the smiley face image. We have so far only presented reconstructions for light that comes from incoherent source diameters of up to 150 μm. Additional simulations actually show that our method continues to produce reconstructions with over 99% correlation for source diameters up to 20,000 μm. However, we have omitted these additional simulations because the results become untrustworthy for incoherent source diameters at 400 μm or above, and we will discuss this further in Section 5.3.

Reconstructions of Image I using Incoherent Light
from Source Apertures with Different Diameters



(a) Coherent (b) 16 μm (c) 48 μm (d) 152 μm (e) Original

Figure 5.1: Each run uses 3,500 basis elements and 9 passes, and each pass checks 17 values per coefficient. Each image was optimized using a different source aperture size, and the values below the images are the diameters of the source aperture. A larger aperture indicates light that is more incoherent and speckles of a smaller size. The purpose of this figure is to show that our reconstructions do not deteriorate as light gets more incoherent, and as we can see, the reconstructions look almost identical to each other.

Correlation Values for Image I Reconstructions using Incoherent Light

Source Diameter (μm)	Speckle Wave Number (m^{-1})	Correlation
0	0	0.993
8	3,100	0.994
16	6,300	0.994
24	9,400	0.994
32	12,600	0.994
40	15,700	0.994
48	18,800	0.994
72	28,300	0.994
152	59,700	0.994

Table 5.1: This table lists the correlations of different image reconstructions of the smiley face using incoherent light. The width of the SLM would correspond to a speckle spatial frequency of about 3,000 m^{-1}, and a spatial frequency of 60,000 m^{-1} corresponds to a speckle radius of about 13 pixels in the smiley-face image.

5.2 Comparison of Our Process against the Gerchberg–Saxton

We simulated the Gerchberg–Saxton algorithm using the instructions and improvements specified in [15]. We used a spatially-filtered intensity image of the SLM as the input and an intensity image of the light after propagation through the optical system as the output. For this output image, we propagated it under both coherent light and partially-coherent light. The partially-coherent light was simulated using a source aperture of point sources feeding into a lens with a focal length of 10 cm placed before the SLM. The phase sample was Image F, which is the letter F, as it is the image that is easiest for the Gerchberg–Saxton to converge while still allowing us to recognize the distinctive features of this constant-thickness image. In this setup, the Gerchberg–Saxton converges for source apertures with diameters below 30 μm and begins to fail for those with diameters above. At around 45 μm, the correlation of the reconstruction is close to zero. This diameter corresponds to a speckle radius around that of the SLM width, which is 1,024 μm in these simulations. This makes sense because the input intensity image is just a flat image of the SLM, and the SLM is the most significant object in the image. As the speckles get smaller than this SLM, the phases within the SLM become uncorrelated and difficult to estimate.

However, our process does not depend on the ability to know the exact phase

of the light; it simply measures the flatness of the combined wave retardation between the SLM and the object. This means that it is expected to converge under incoherent light, too. We re-simulated our process with the exact same configurations and images as the Gerchberg–Saxton configurations. Note that these configurations are different from those in Section 5.1, hence our values are not the results from Table 5.1. We have placed recorded the numerical values of our results from these runs in Table B.1 in Appendix B. Our process consistently converged to correlations above 99%, even for source apertures of diameters as high as 20,000 μm. The results of the comparison between our process and the Gerchberg–Saxton algorithm for source aperture diameters of up to 50 μm are shown in Figure 5.2, which combines the raw data from Table B.1 with the results of the Gerchberg–Saxton algorithm.

As we can see, our process performs better for light of lower coherence. It is therefore a potential method for phase retrieval under incoherent light. In fact, the high correlation remains even for light with a speckle size smaller than most features in the object, meaning that it does not need coherence in any part of the object.

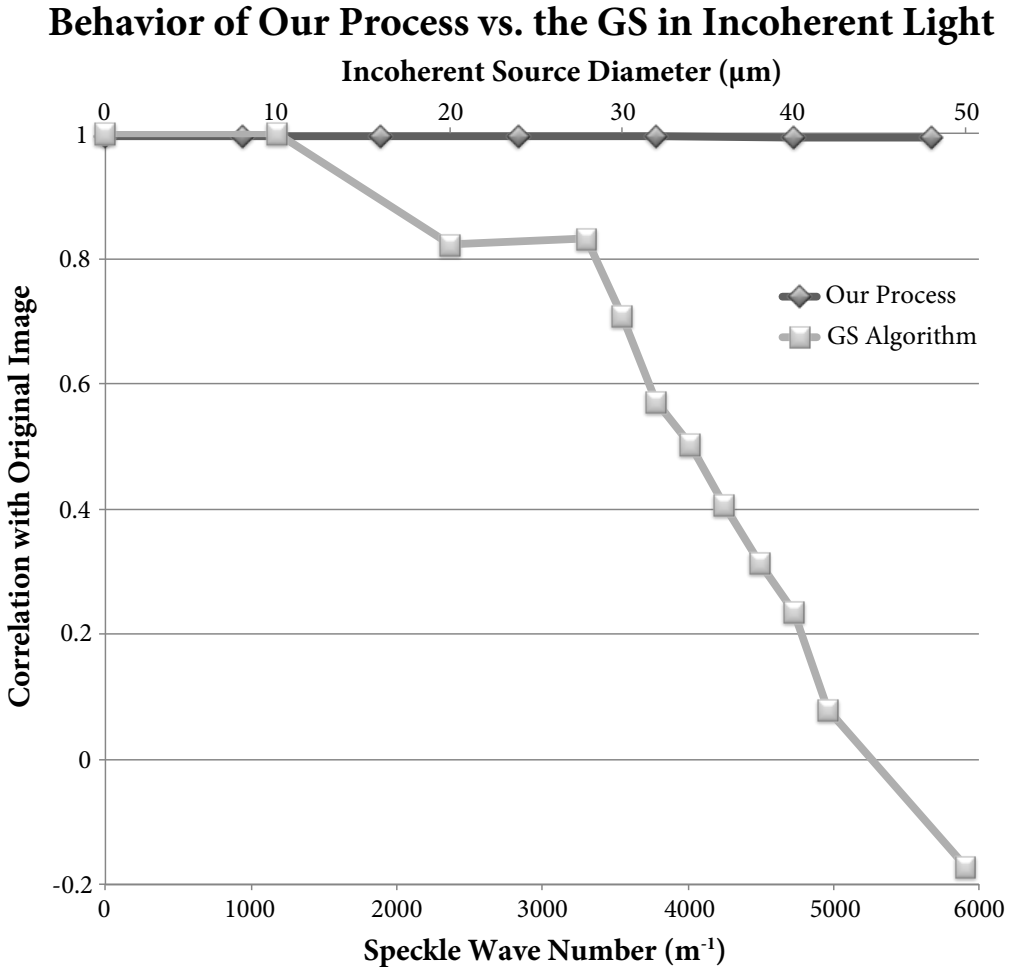


Figure 5.2: This plot compares the behavior between our process and the Gerchberg–Saxton algorithm. The Speckle Wave Number is 2π divided by the distance between speckles. The propagation used for the input to output image in the GS algorithm uses a lens with a focal length of 10 cm, and the original image is a letter F with height 600 μm. The SLM is 1,024 μm wide, which corresponds to a wave number of 6,000 m^{-1}. For this comparison, the same exact image, SLM size, and lens focal length are used for the propagation in our process. In addition, we limit each run to 800 basis elements and 7 passes, and each pass guesses 17 values per coefficient. These settings were chosen to reduce the runtime, since simulations for incoherent light are lengthy.

5.3 Error and Robustness Analysis

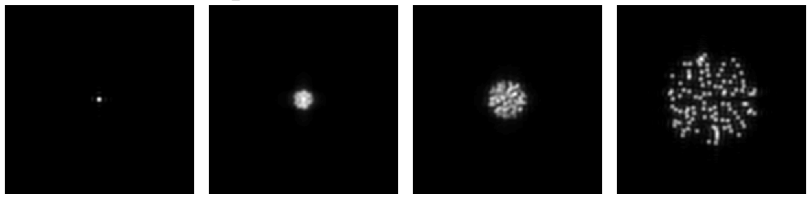
While our algorithm can tolerate inaccuracies in the optical system and the values guessed, we found that our algorithm is extremely sensitive to any inconsistency in the simulation of the optical propagation. This is because on each pass, the robustness depends on the ability for the old best value to be better than all the worse values again. In addition, the E_k's are quite close to each other in value for most of the coefficients guessed, implying that small inconsistencies in the optical system dominate the difference between E_k values. If the optical system is slightly inconsistent between two identical simulations, then some other value could by chance produce a lower E_k than the old best value, and so the true E_k (of a correct propagation) could actually increase after this iteration. As a result, the true E_k's no longer form a monotonically-decreasing sequence, and our algorithm would fail to converge. We expect this drawback to not be a problem in the laboratory because the light would have enough photons so that these fluctuations average out and become insignificant.

To overcome this drawback in our simulations, we deterministically chose the random point sources within the source aperture before each run and used these point sources for all propagations during the run. This allows each propagation to still be incoherent, and it enables the propagations to be consistent in their errors.

In addition, as we discussed in Section 5.1, we omitted the results for simulations of incoherent light coming from source apertures that have a diameter of 400 μm or above. In order for our simulations to finish in a reasonable amount of time, we used only 25 to 100 point sources from the source aperture for each call to our propagation algorithm. Consider the camera output images from Figure 5.3 for different incoherent source aperture diameters. The resulting intensity from each point source correspond to a bright spot on the camera where the point source was located, since both the source aperture and the camera are at the Fourier plane of the image. When the source aperture diameter is small, the camera outputs show the combined intensities from these point sources overlapping with each other. This means that we see the effects of incoherent light, which is that the intensities add without any of the interference cross-terms. However, as the source aperture diameter get larger, these point sources get spread out, and there is less overlapping intensity between them. At a diameter of 400 μm, the regions of illumination from the point sources almost do not overlap, and at 1000 μm, the regions are almost completely separated. When the illumination regions from

the point sources are all separated, there is very little summation of intensities at most pixels; instead, each point source has its own region of illumination. So even though our process converges well for illumination from these large source apertures, the results of these simulations would be untrustworthy because our algorithm would be able optimize on the undisturbed regions of illumination from each point source, without any of the effects of incoherence. However, this is not a very bad limitation of our simulations, because 400 μm is well beyond the amount of incoherence we are interested in. This source aperture diameter corresponds to a speckle size of under 5 pixels, or 40 μm, which is already smaller than most features in our object. In addition, as we will saw in Section 5.2, the Gerchberg–Saxton fails at source aperture diameters that are less than 40 μm, which is far smaller than 400 μm.

Camera-Plane Images using Incoherent Light from Source Apertures with Different Diameters



(a) Coherent (b) 200 μm (c) 400 μm (d) 1000 μm

Figure 5.3: This is a side by side comparison of the camera outputs from our propagation simulation for an image with a SLM that is 1,024 μm wide with a letter F phase image that is 600 μm tall. These illustrations show that we may not be able to trust the results of our simulations at 400 μm and beyond.

Overall, even though there are many drawbacks with our simulation method for incoherent light, we expect that these drawbacks exist only in simulation and will not be a problem when we test this process in the laboratory. Our simulations were reliable for the region over which we compared against the Gerchberg–Saxton algorithm, and we conclude based on our results from Figure 5.2 on page 53 that our algorithm performs better than the Gerchberg–Saxton algorithm for light with incoherence.

Chapter 6

Conclusion

We have introduced the basics of Fourier optics and phase retrieval, detailed the setup and implementation of our phase-retrieval process, simulated our process with coherent light, and simulated our process with incoherent light.

We found that our method of phase retrieval by using the SLM to conjugate the wave retardation of the object can reconstruct images reasonably well using both coherent and incoherent light without any prior knowledge regarding the phase sample. Most reconstructions produced over 99% correlation using less than 10% of the degrees of freedom (6,500 basis elements for 65,536 pixels). The performance under incoherent light surpasses that of the Gerchberg–Saxton algorithm, which does not converge well for incoherent light. Our method could therefore be useful as a method of direct phase imaging under incoherent light.

However, the success of this method has yet to be demonstrated in the laboratory, and the large runtime of this relatively brute-force method can potentially be a concern, depending on the speed at which SLMs can change their modulation. In the current configurations, we would need a SLM that can change ten thousand times per second coupled with a camera that can take ten thousand pictures per second to produce a 256×256 reconstruction within 30 seconds. Our method is a very brute-force method that, as of now, only uses aggregate information about the whole output image and does not actively take advantage of any spatial information related to specific regions or pixels in the output image. We suspect that smarter methods that actively take advantage of spatial information can converge much faster. Possible methods to try include using a wavelet basis instead of the cosine transform basis or simply choosing the SLM modulation based on the actual pixel-by-pixel information (as opposed to Euclidean norm) from the output. In addition, there now exist large computer-vision databases consisting of hundreds of

millions of existing images, and algorithms for processing big data are improving. As a result, there can potentially be algorithms yet to be discovered that could in real-time take advantage of statistics from existing similar images in the database as the reconstruction is converging.

We chose to use this brute-force method because it has a large convergence basin and is likely to converge for any initial image. It does not depend on the reliability of the first few iterations or the initial conditions, because it builds the image from scratch. As a result, this method was a suitable proof of concept for the idea of phase retrieval by using the SLM to flatten the wavefront, and we have successfully shown via simulations that it converges for most phase objects under both coherent and incoherent light.

The most immediate next step for this phase retrieval algorithm would be to test it in the laboratory under both incoherent and coherent light and produce results for images of different types. Once this brute-force algorithm has been shown to work, we can optimize wavefront flattening to become faster and more efficient at direct phase imaging under incoherent light.

Appendix A

Code

Here is an example MATLAB code, mostly written by Alexandre Goy, containing the beam propagation method,[20] the quadratic phase factor from the lens, and incoherent illumination. This code was used to produce the target image for both the Gerchberg–Saxton algorithm and for our process (when reconstructing the letter F) under incoherent light.

```
clear all;
close all;

Nx = 256;
Ny = 256;

slm_pixel_size = 8e-6;
dx = slm_pixel_size;
dy = dx;

Lx = Nx*dx;
Ly = Ny*dy;
dkx =2*pi/Lx;
dky =2*pi/Ly;

x = (-Nx/2+1:Nx/2)*dx;
y = (-Ny/2+1:Ny/2)*dy;
kx = (-Nx/2:Nx/2-1)*dkx;
ky = (-Ny/2:Ny/2-1)*dky;
```

```
[X, Y] = meshgrid(x, y);
R2 = X.^2 + Y.^2;
R = sqrt(R2);
[Kx, Ky] = meshgrid(kx, ky);
K2 = Kx.^2 + Ky.^2;
K = sqrt(K2);

n0 = 1.0;
lambda0 = 532e-9;
k0 = 2*pi/lambda0;
k = n0*k0;

lens_focal_length = 100e-3;
lens_phase = exp(-1j * k * (lens_focal_length - sqrt( ...
lens_focal_length^2 + R2)));


% Generate object

object_pattern = double(imread('letter256.png'));
object_pattern = object_pattern-min(object_pattern(:));
object_pattern = object_pattern/max(object_pattern(:));

% Spatial filter the object so that it does not hit the edge when
%   propagating
filter_bandwidth = 0.5*k*sin(atan(Lx/(2*lens_focal_length)));
spatial_filter = exp(-(K2/filter_bandwidth^2));
object_pattern_filtered = ifft2(fft2(object_pattern).*fftshift( ...
spatial_filter));

object_phase_mod = pi/2;
object = exp(1j*object_phase_mod*object_pattern_filtered);

% Use the focal length as the propagation distance
propagation_operator = fftshift(exp(1j*(k - sqrt(k^2 - Kx.^2 ...
```

```
- Ky.^2))*lens_focal_length));

source_radius = 0.08e-3;
% Radius of the incoherent source in therm of the spatial
%   frequency of the speckle
source_frequency_radius = k*sin(atan(source_radius ...
/lens_focal_length));
% Aperture within which the random point sources will be
%   placed
source = double(K <= source_frequency_radius);
% We initialize the detected intensity to 0 and we will add
%   the incoherent contribution
I_detection = zeros(Ny, Nx);
% Number of random sources
Nsource = 500;

I_illum = zeros(Ny, Nx);
I_illum(Ny/4+1:3*Ny/4, Nx/4+1:3*Nx/4) = 1;
I_illum = gaussian_filter(I_illum, 3);
A_illum= sqrt(I_illum);

for ind_source = 1:Nsource
% Randomly place a source at some distance from the axis
source_frequency_modulus = sqrt(rand(1))*source_frequency_radius;
azimuthal_angle_source = rand(1)*2*pi;
% Randomly pick an azimuthal angle for the source location
kxs = source_frequency_modulus*cos(azimuthal_angle_source);
kys = source_frequency_modulus*sin(azimuthal_angle_source);

% Uncomment these two lines to simulate coherent light.
% kxs = 0;
% kys = 0;

% Define the illumination as a function of the source location
%   in the frequency domain
u_illum = A_illum.*exp(1j*(kxs*X + kys*Y));
```

```
% u1 is the field right after the object
u1 = u_illum.*object;
% Propagate from the object to the lens
u_before_lens = ifft2(fft2(u1).*propagation_operator);
% Pass through the lens
u_after_lens = u_before_lens .* lens_phase;
% Propagate from the lens to the detector
u_detection = ifft2(fft2(u_after_lens).*propagation_operator);
% Add the intensity onto the detector
I_detection = I_detection + abs(u_detection).^2;

clf, subplot(1, 2, 1), imagesc(abs(u_before_lens)), axis equal, ...
colormap(gray), title('Amplitude before the lens');
subplot(1, 2, 2), imagesc(abs(u_detection(3*Ny/8+1:5*Ny/8, ...
3*Nx/8+1:5*Nx/8))), axis equal, colormap(gray), ...
title('Amplitude in the detection plane (central portion)');
drawnow;

end

figure, imagesc(source), axis equal, colormap(gray), ...
title('Source aperture');
figure, imagesc(angle(object)), axis equal, colormap(gray), ...
title('Object phase');
figure, imagesc(abs(u1).^2), axis equal, colormap(gray), ...
title('Intensity in the object plane (represents SLM area)');
figure, imagesc(I_detection(3*Ny/8+1:5*Ny/8, 3*Nx/8+1:5*Nx/8)),  ...
axis equal, colormap(gray), ...
title('Total intensity (central portion)');
```

Appendix B

Simulation Results of Our Process under Incoherent Light

We present here the results of running our process on the exact same conditions as the Gerchberg–Saxton for incoherent light. We used an SLM that is 1,024 μm wide (128 pixels), with a computational window size of 256×256 for the propagation of incoherent light. The image we tested was Image F, which was the letter F. This image was spatially filtered for both the Gerchberg–Saxton and our process so that the propagation remains in the computational window. To reduce computation time, we allowed our algorithm only 800 basis elements and 17 guesses per basis element. The incoherent light was propagated with 25 point sources chosen by stratified random sampling (the unit circle was divided into 25 equal-area region and a point source was randomly chosen from each region). For incoherent light at 1000 μm and above, we chose our initial basis elements from the sinc() function (which contains no information about the image itself) because the results of the initial incoherent propagation were not trustworthy, but the convergence was completed as our procedure intended. Figure 5.2 uses the values here up to a source diameter of 48 μm.

Correlation Values for Image F Reconstructions using Incoherent Light

Source Diameter (μm)	Correlation
0	0.996
8	0.996
16	0.996
24	0.995
32	0.995
40	0.994
48	0.993
100	0.992
200	0.995
400	0.995
1,000	0.990
2,000	0.990
4,000	0.989
10,000	0.990
20,000	0.989
40,000	0.989
100,000	0.991

Table B.1: This table lists the raw data of the final correlations of different image reconstructions of the letter F from our process using incoherent light. The runs for source diameters of 400 μm and above are less trustworthy, as discussed in Section 5.3.

Bibliography

[1] Bradley Atcheson, Wolfgang Heidrich, and Ivo Ihrke. An evaluation of optical flow algorithms for background oriented schlieren imaging. *Experiments in Fluids*, 46(3):467–476, 2009.

[2] Peter Török and Fu-Jen Kao. *Optical imaging and microscopy: techniques and advanced systems*, volume 87. Springer-Verlag, 2003.

[3] Robert A. Gonsalves. Phase retrieval and diversity in adaptive optics. *Optical Engineering*, 21(5):215829–215829–, 1982.

[4] Chien-Hung Lu, Christopher Barsi, Matthew O Williams, J Nathan Kutz, and Jason W Fleischer. Phase retrieval using nonlinear diversity. *Applied optics*, 52(10):D92–D96, 2013.

[5] R. W. Gerchberg and W. O. Saxton. A practical algorithm for the determination of the phase from image and diffraction plane pictures. *Optik*, 35:237–246, 1972.

[6] Frits Zernike. Phase contrast, a new method for the microscopic observation of transparent objects part i. *Physica*, 9:686–698, 1942.

[7] A. Korpel, D. Mehrl, and H. H. Lin. Schlieren imaging of sound fields. In *IEEE 1987 Ultrasonics Symposium*, pages 515–518, Oct 1987.

[8] James R. Fienup. Phase retrieval algorithms: a comparison. *Applied optics*, 21(15):2758–2769, 1982.

[9] M. G. L. Gustaffson. Surpassing the lateral resolution limit by a factor of two using structured illumination microscopy. *Journal of Microscopy*, 198:82–87, 2000.

[10] J. Mertz. Optical sectioning microscopy with planar or structured illumination. *Nature Methods*, 8:811–819, 2011.

[11] M. J. Booth. Adaptive optics in microscopy. *Phil. Trans. of the Royal Soc.*, 365:2829–2843, 2007.

[12] David Jeffrey Griffiths. *Introduction to electrodynamics*, volume 3. Addison Wesley, 1999.

[13] Joseph W Goodman et al. *Introduction to Fourier optics*. Roberts & Company, third edition, 2005.

[14] Morner. A representation of the recursive gerchberg-saxton algorithm. Wikipedia. http://en.wikipedia.org/wiki/File:GS-diagram.png.

[15] J. R. Fienup and C. C. Wackerman. Phase-retrieval stagnation problems and solutions. *JOSA A*, 3(11):1897–1907, 1986.

[16] Jerome Mertz. *Introduction to optical microscopy*. Roberts, 2010.

[17] Jonathan C Petruccelli, Lei Tian, and George Barbastathis. The transport of intensity equation for optical path length recovery using partially coherent illumination. *Optics express*, 21(12):14430–14441, 2013.

[18] Chang Chang and Takashi Nakamura. Partially coherent image formation theory for x-ray microscopy. *Microscopy: Science, Technology, Applications and Education*, 3:1897–1904, 2010.

[19] A. Qu, A. Goy, and J. W. Fleischer. Phase retrieval using optimized conjugated illumination. This figure was adapted from our submission to the Computational Optical Sensing and Imaging conference, 2015.

[20] C. Pollock and M. Lipson. *Integrated Photonics*. Springer US, 2013.