**The Non-Mechanical Beam Steering of Light in Reflective Inverse Diffusion**

THESIS

Eric K. Nagamine, 2Lt, USAF

AFIT-ENP-19-M-086

**DEPARTMENT OF THE AIR FORCE**
**AIR UNIVERSITY**

***AIR FORCE INSTITUTE OF TECHNOLOGY***

**Wright-Patterson Air Force Base, Ohio**

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

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

AFIT-ENP-19-M-086

# The Non-Mechanical Beam Steering of Light in Reflective Inverse Diffusion

THESIS

Presented to the Faculty

Department of Engineering Physics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Applied Physics

Eric K. Nagamine, B.S.

2Lt, USAF

February 26, 2019

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENP-19-M-086

# The Non-Mechanical Beam Steering of Light in Reflective Inverse Diffusion

THESIS

Eric K. Nagamine, B.S.
2Lt, USAF

Committee Membership:

Lt Col Kenneth W. Burgi, PhD
Chair

Lt Col Samuel D. Butler, PhD
Member

Michael A. Marciniak, PhD
Member

AFIT-ENP-19-M-086

# Abstract

Wavefront shaping is a technique that uses spatial light modulators to conjugate the phase of light incident on a rough surface, such that the light will refocus after reflection. This refocusing effect is called reflective inverse diffusion. There currently are two different approaches used to achieve reflective inverse diffusion: iterative methods and matrix methods. Iterative methods find one phase mask which allows for reflected light to be focused at a single, specific position, with results that are immediately available and continuously improving. Matrix methods calculate the complex matrix which describes the rough surface and allows for reflected light to be focused at many positions after reflection and at multiple spots simultaneously. However, matrix methods are susceptible to decreased performance in a noisy system, and their results are not immediately available. This thesis provides an alternative to the current paradigm of choosing between iterative methods and matrix methods by showing that beam steering can be integrated into existing iterative methods, giving them the multiple-spot capabilities of matrix methods. For a focal plane system, a genetic algorithm not previously used for reflective inverse diffusion is used to find a phase mask that focuses light to one point. Circular shifts are then applied to the phase mask that create phase tilts at the rough surface and steer the reflected spot at the cost of decreased enhancement with a larger shift.

# Table of Contents

Page

Abstract . . . . . iv

List of Figures . . . . . viii

List of Tables . . . . . x

List of Abbreviations . . . . . xi

I. Introduction . . . . . 1

1.1 Motivation . . . . . 1
1.2 Application of Dual Photography . . . . . 2
1.3 Objective . . . . . 3
1.4 Overview . . . . . 3

II. Background . . . . . 4

2.1 Introduction . . . . . 4
2.2 Fourier Transforms . . . . . 4
2.3 Discrete Fourier Transforms and the Fast Fourier Transform . . . . . 5
2.4 Rayleigh-Sommerfeld Diffraction . . . . . 7
2.5 Non-Mechanical Beam Steering . . . . . 9
2.6 Spatial Light Modulators . . . . . 10

III. Inverse Diffusion Theory . . . . . 12

3.1 Introduction . . . . . 12
3.2 Wavefront Shaping . . . . . 12
Different Approaches to Wavefront Shaping . . . . . 14
3.3 Matrix Calculation Methods . . . . . 15
3.4 Iterative Methods . . . . . 16
3.5 Considerations for Iterative Algorithm Parameters . . . . . 18
Search Space . . . . . 18
Algorithm Speed . . . . . 20
Decorrelation . . . . . 21
3.6 Reflective Inverse Diffusion . . . . . 22
3.7 Summary . . . . . 23

IV. Reflective Inverse Diffusion . . . . . 26

4.1 Introduction . . . . . 26
4.2 Equipment . . . . . 26

Page


4.3 Optical Setups . . . 27
Simulations . . . 27
Laboratory . . . 27
4.4 Mathematical Model . . . 28
Diffraction Based Propagator . . . 28
Channel Size . . . 29
4.5 Genetic Algorithm . . . 30
4.6 Algorithm Performances . . . 32
Simulated Results . . . 32
Experimental Results . . . 36
4.7 Conclusions . . . 38

V. Non-Mechanical Beam Steering in Reflective Inverse Diffusion . . . 40

5.1 Introduction . . . 40
5.2 Creating Phase Tilts at the Scattering Sample . . . 41
5.3 Creating Spot Displacement . . . 43
5.4 Limitations of the Circular Shift . . . 44
Loss of Enhancement . . . 44
Discrete Displacement Size . . . 45
5.5 Assessing Shift Performance . . . 46
Circular Shift Procedure . . . 46
Simulated Results . . . 47
Experimental Results . . . 52
5.6 Conclusions . . . 56

VI. Conclusion . . . 58

6.1 Research Summary . . . 58
6.2 Acknowledgements . . . 58
6.3 Future Work . . . 59
Construction of an RM . . . 59
Phase Mask Interpolation . . . 60
Integration of Measurements and Simulations . . . 60
A Lensless Setup . . . 61

Appendix A. Rayleigh-Sommerfeld Diffraction Propagator in Matlab® . . . 63

Appendix B. Iterative Algorithm Simulations in Matlab® . . . 66

2.1 Continuous Sequential Algorithm . . . 66
2.2 Partitioning Algorithm . . . 68
2.3 Genetic Algorithm . . . 71

Page

Appendix C. Beam Steering Simulations in Matlab® . . . . . 76

Bibliography . . . . . 79

# List of Figures


| Figure | | Page |
|---|---|---|
| 1 | Dual Photography | 2 |
| 2 | Rayleigh Sommerfeld Diffraction | 7 |
| 3 | Reflection from a Blazed Grating | 10 |
| 4 | Reflection from a Liquid Crystal Phased Array | 10 |
| 5 | Design Concept of a Nematic Cell | 11 |
| 6 | Vellekoop and Mosk's Experiment for Transmissive Inverse Diffusion | 13 |
| 7 | Parallel Wavefront Optimization | 16 |
| 8 | The Continuous Sequential Algorithm | 17 |
| 9 | The Partitioning Algorithm | 18 |
| 10 | Conkey's Genetic Algorithm | 19 |
| 11 | Simulated Focal Plane System | 23 |
| 12 | Image Plane System | 24 |
| 13 | Laboratory Focal Plane System Setup | 27 |
| 14 | Diffraction Limited Spot Size | 30 |
| 15 | Genetic Algorithm Breeding Process | 32 |
| 16 | Simulated Performance of Different Reflective Inverse Diffusion Algorithms | 34 |
| 17 | Experimental Performance of Iterative Algorithms | 37 |
| 18 | Decorrelation in the Focal Plane System | 38 |
| 19 | Methods for Moving an SLM Phase Mask | 42 |
| 20 | Diagram of Locations of CCD Channels Chosen for Optimization in Laboratory Experiments | 47 |

Figure Page


21 Simulated Enhancement of Displaced Spot as a Two Dimensional Function of Displacement Size . . . . . 48

22 Simulated Enhancement of Displaced Spot as a Two Dimensional Function of Displacement Size, with Sample not at Focus . . . . . 49

23 Phase Tilts at the Rough Surface . . . . . 51

24 Experimental Enhancement of a Displaced Spot as a Two Dimensional Function of Displacement Size on Polished Nickel . . . . . 53

25 Experimental Enhancement of a Displaced Spot as a Two Dimensional Function of Displacement Size on Zinc Oxide White Paint . . . . . 54

26 Genetic Algorithm Performance in a Lensless System . . . . . 62

# List of Tables


Table Page

1 Predicted and Simulated Minimum Displacement Sizes . . . . . 49

2 Predicted and Tested Minimum Mean Displacement Sizes . . . . . 55

3 Surface Properties of Scattering Samples . . . . . 56

# List of Abbreviations

| Abbreviation | | Page |
|---|---|---|
| DFT | Discrete Fourier Transform | 5 |
| FFT | Fast Fourier Transform | 6 |
| SLM | Spatial Light Modulator | 10 |
| LCOS | Liquid Crystal on Silicon | 11 |
| CCD | Charge-Coupled Device | 13 |
| TM | Transmission Matrix | 15 |
| RM | Reflection Matrix | 15 |
| PWO | Parallel Wavefront Optimization | 15 |
| CSA | Continuous Sequential Algorithm(s) | 16 |
| PA | Partitioning Algorithm(s) | 16 |
| GA | Genetic Algorithm(s) | 17 |
| LUT | Lookup Table | 26 |
| RMS | Root Mean Square | 31 |
| GPU | Graphics Processing Unit | 33 |
| DDDAS | Dynamic Data Driven Application Systems | 57 |

The Non-Mechanical Beam Steering of Light in Reflective Inverse Diffusion

# I. Introduction

## 1.1 Motivation

Optical imaging systems depend upon ballistic light that travels along a direct line of sight from the target being imaged to the imaging system. Imaging cannot be performed if there are obstacles or processes along a path that scatter light and eliminate the straight line of sight. A device which could virtually create a line of sight where it does not naturally exist would allow users to see around corners or past obstacles without physical equipment in that line of sight. In the hands of a trained warfighter, this capability could give a dearly needed tactical edge. Diffuse scattering due to reflection off of a rough surface is one effect that makes standard imaging impossible. When a surface is not optically smooth, it means that the height and slope of the surface vary rapidly compared to the wavelength of incident light. These rapid variations cause reflected light to be scattered, which is a reason why one is unable to see reflections off of rough surfaces. However, using spatial light modulators, a process called wavefront shaping allows for light to be refocused instead of scattered after reflection from a rough surface. This refocusing effect is called reflective inverse diffusion, and is a necessary part of the process that could be used to see around corners.

## 1.2 Application of Dual Photography

The application of reflective inverse diffusion that could allow for seeing around corners is in a technique called Dual Photography [1]. Dual photography is based on Helmholtz reciprocity, which is the concept that a "signal" can be propagated between two points starting at one or the other, and there is a symmetric relationship where the receiving point will receive the same signal. In dual photography, by exploiting Helmholtz reciprocity, it is possible to virtually interchange the position of a camera and a projector, as shown in Figure 1.

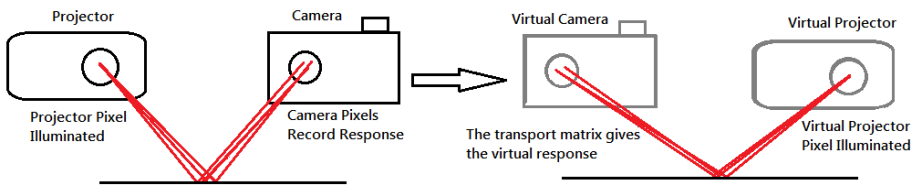



**Figure 1. The concept behind dual photography. The propagation of light between projector and camera is measured, allowing for the positions of the projector and the camera to be virtually interchanged. Hidden scene information, from the point of view of the projector can be uncovered.**

By mapping the propagation of light from each pixel of the projector to each pixel of the camera in a transport matrix, virtual propagation in the reverse direction through the matrix can uncover hidden scene information [2]. In practice, this by itself is not very useful, because if there is a projector available for use at some location, there are few reasons why a camera itself could not be placed there, defeating the need for dual photography. The novel application of reflective inverse diffusion is that it allows for the creation of a focused beam after random scattering, which means that a rough surface could be made into a projector. Therefore, because dual photography allows for the virtual interchange of a projector and a camera, a virtual camera could be created at the rough surface, which should reveal hidden information from the perspective of the rough surface i.e. seeing around corners.

## 1.3 Objective

Currently, wavefront shaping techniques used to achieve reflective inverse diffusion work by finding phase masks that are applied to light incident on a rough surface with an SLM. Ideally, the phase mask conjugates the light so that when it reflects from the rough surface, it refocuses into a single beam. A key limitation of current techniques is that they require a different phase mask for every different direction that the refocused beam is aimed. Imaging applications would naturally require that the refocused beam is scanned across a large area, so this would require the creation of many different phase masks, which is prohibitively slow.

The objective of this research is to demonstrate that in wavefront shaping, one phase mask is capable of being repurposed to refocus light in reflective inverse diffusion in more than one direction. This allows for scanning of the refocused beam across an area without the need to find more than one phase mask.

## 1.4 Overview

Non-specific optics and mathematical principles, which are important to the research are covered as background in Chapter II. The theories behind wavefront shaping methods that create inverse diffusion, including previous work performed at AFIT are included in Chapter III. The process used to achieve reflective inverse diffusion, which includes a new and robust genetic algorithm for wavefront shaping, is covered in Chapter IV. Last, non-mechanical beam steering is covered in Chapter V followed by a conclusion in Chapter VI that provides on overview of the work, and a preview of work to come.

# II. Background

## 2.1 Introduction

This chapter is meant to be a review of mathematics and optics principles which are important for understanding wavefront shaping and reflective inverse diffusion research, but are not specific to it. The properties of analytic Fourier transforms are discussed, followed by an introduction to discrete Fourier transforms as computational tools. Next, the model of Rayleigh-Sommerfeld diffraction for light propagation is discussed. Finally, non-mechanical beam steering, spatial light modulators, and their connection are discussed.

## 2.2 Fourier Transforms

A Fourier transform is a mathematical operation that relates a function defined in position or temporal space to frequency space. The two dimensional Fourier transform and inverse Fourier transform are defined by the Fourier integrals in Equations (1) and (2) respectively.

$$G(f_x, f_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\exp\left[-i2\pi(f_x x + f_y y)\right]dxdy, \tag{1}$$

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(f_x, f_y)\exp\left[i2\pi(f_x x + f_y y)\right]df_x df_y, \tag{2}$$

for a function $g(x,y)$, and its frequency spectra $G(f_x, f_y)$, with independent variables x and y. The Fourier transform has been well studied, so properties of the Fourier transform, such as Fourier transform pairs of common functions, and other properties are known. The convolution theorem of Fourier transforms in Equation (3) makes Fourier transforms a useful tool for computing convolutions [3]. The theorem states

that the Fourier transform of two functions being convolved is equal to the product of the Fourier transforms of the individual functions. In mathematical notation, the theorem is

$$\mathcal{F}[A * B] = \mathcal{F}[A]\mathcal{F}[B], \tag{3}$$

for functions A and B, where $\mathcal{F}$ is the Fourier transform operator, and $*$ is the convolution operator. The shift theorem of Fourier transforms in Equation (4) relates translations of a function in one domain to phase shifts of the Fourier transform.

$$\mathcal{F}[g(x-a, y-b)] = \mathcal{F}[g(x,y)] \exp[-i2\pi(f_x a + f_y b)], \tag{4}$$

where a and b are constants. The shift theorem shows that when a function is translated before undergoing a Fourier transform, the transform remains the same except that a linearly tilted phase shift is applied to it, with a slope proportional to the size of the translation. The Fourier integral theorem in Equation (5) states that an inverse Fourier transform undoes a Fourier transform, and that a Fourier transform undoes an inverse Fourier transform.

$$\mathcal{F}[\mathcal{F}^{-1}[g(x,y)]] = \mathcal{F}^{-1}[\mathcal{F}[g(x,y)]] = g(x,y), \tag{5}$$

where $\mathcal{F}^{-1}$ is the inverse Fourier transform operator. The Fourier integral theorem simplifies operations where repeated Fourier transforms and inverse Fourier transforms are used.

### 2.3 Discrete Fourier Transforms and the Fast Fourier Transform

Real data sets are naturally discrete, and finite. On discrete data, analytic Fourier transforms cannot be used. The discrete Fourier transform (DFT) is the analogous

operation that is used for discrete data, shown in Equation (6). The discrete frequency spectrum $G(f_x, f_y)$ found by the DFT is given by

$$G(f_x, f_y) = \sum_{n=0}^{N_x-1} \sum_{m=0}^{N_y-1} g(n\Delta x, m\Delta y) \exp\left[-i2\pi(f_x n\Delta x + f_y m\Delta y)\right]. \tag{6}$$

where data, $g(x, y)$, is sampled from a two dimensional space space $N_x$ by $N_y$ points wide, taken at at even intervals of $\Delta x$ and $\Delta y$ respectively. Equation (6) can be used to find the DFT of a data set element by element, or simultaneously if matrix algebra is used, but it takes on the order of $N_x^2 N_y^2$ operations to compute the entire spectrum [3]. The relationship between the interval over which the function is sampled and the spacing of the discrete frequency spectrum is given in Equation (7) as

$$\begin{aligned} \Delta f_x &= \frac{1}{(N_x - 1)\Delta x} \\ &= \frac{1}{L_x}, \end{aligned} \tag{7}$$

where $L_x$ is the length of the sampled region along the x-dimension, and the same relation applies to the y dimension. Similarly, the relationship between the spacing between sampled points, and the range of the frequency spectrum is given by Equation (8) as

$$\begin{aligned} \frac{1}{\Delta x} &= (N_x - 1)\Delta f_x \\ &= L_{f_x}, \end{aligned} \tag{8}$$

where $L_{f_x}$ is the length in frequency space of the generated frequency spectrum. The fast Fourier transform (FFT) is a class of algorithms that can reduce the cost of DFTs to an order of $N_x log(N_x) N_y log(N_y)$ operations [4]. All FFT algorithms operate on the principle of breaking down a large DFT recursively into smaller DFTs [4]. For

applications where a DFT is required, such as in diffraction simulations, FFTs are the algorithms that are typically used because of their efficiency.

### 2.4 Rayleigh-Sommerfeld Diffraction

Rayleigh-Sommerfeld diffraction is a model for describing the propagation of light, based on the Huygens-Fresnel principle, which treats every point along a wavefront as a source of secondary spherical wavelets [3]. In Figure 2, the electric field, $U(P_1)$, at a finite aperture, $\Sigma$, propagates to create the field at a point some distance away, $U(P_0)$. The Rayleigh-Sommerfeld Diffraction integral, in Equation (9), describes this

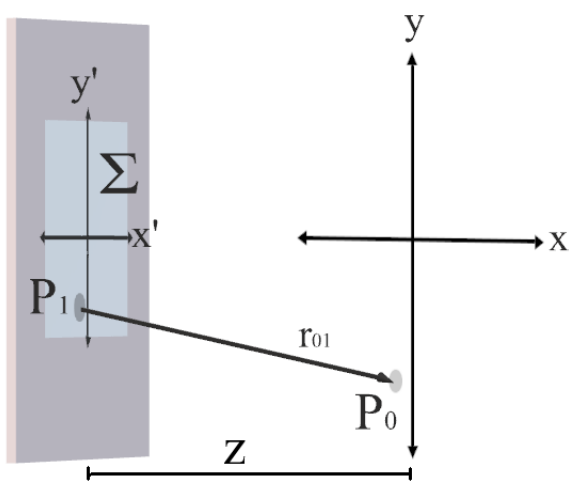



**Figure 2. Given a known field distribution $U(P_1)$ across an aperture $\Sigma$, the Rayleigh-Sommerfeld Diffraction integral calculates the electric field due to secondary wavelets as they propagate to a point $P_0$.**

propagation;

$$U(P_0) = \frac{1}{i\lambda} \iint_{\Sigma} U(P_1) \frac{\exp(ikr_{01})}{r_{01}} \cos(\theta) ds, \tag{9}$$

where $\lambda$ is the wavelength, $k$ is the propagation constant of the wave, $\vec{r}_{01}$ is a vector from the field point to the center of the aperture, and $\theta$ is defined by the angle between the aperture's surface normal and $\vec{r}_{01}$. Rayleigh-Sommerfeld diffraction is valid

so long as $r_{01} \gg \lambda$, and is more accurate than the Fresnel and Fraunhoffer models of diffraction, which are themselves derived from Rayleigh-Sommerfeld diffraction. However, the Rayleigh-Sommerfield diffraction integral is often not analytically solvable [3].

The Rayleigh-Sommerfeld diffraction integral can also be viewed as a convolution of the source field with the spherical wave impulse response, shown in Equation (10).

$$U(P_0) = U(P_1) * h(x, y, z), \tag{10}$$

where $h(x, y, z)$ is the spherical wave impulse response, defined in Equation (11) as

$$h(x, y, z) = \frac{z}{i\lambda} \frac{\exp(ikr_{01})}{r_{01}^2}. \tag{11}$$

Using the convolution theorem of Fourier transforms in Equation (3), the field $U(P_0)$ can be determined without using the Rayleigh-Sommerfeld Diffraction integral. The field is given in Equation (12) as

$$U(P_0) = \frac{z}{i\lambda} \mathcal{F}^{-1}[\mathcal{F}[U(P_1)]H(f_x, f_y, z)], \tag{12}$$

where $H(f_x, f_y, z)$ is called the Rayleigh-Sommerfeld transfer function. $H(f_x, f_y, z)$ is defined by the Fourier transform shown in Equation (13) as

$$\frac{z}{i\lambda} H(f_x, f_y, z) = \mathcal{F}[h(x, y, z)], \tag{13}$$

where the analytic solution to $H(f_x, f_y, z)$ is given by Equation (14) as

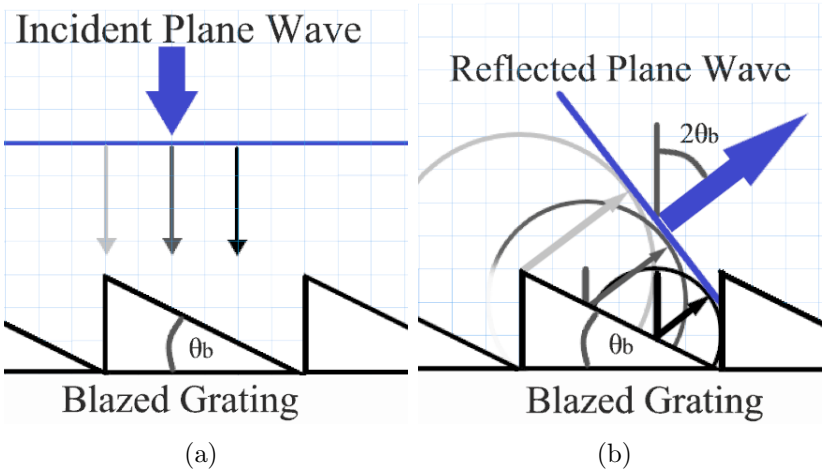
$$H(f_x, f_y, z) = \exp\left(ikz\sqrt{1 - (\lambda f_x)^2 - (\lambda f_y)^2}\right). \tag{14}$$

Therefore, Equation (12) indicates that the propagation of light using the model of Rayleigh-Sommerfeld diffraction can be computed with Fourier transforms. Using FFTs, this process can efficiently model light propagation without the need for point by point integration as in Equation (9). Using a diffraction-based propagation model is important to capture phase-based effects such as phase modulation and non-mechanical beam steering.

### 2.5 Non-Mechanical Beam Steering

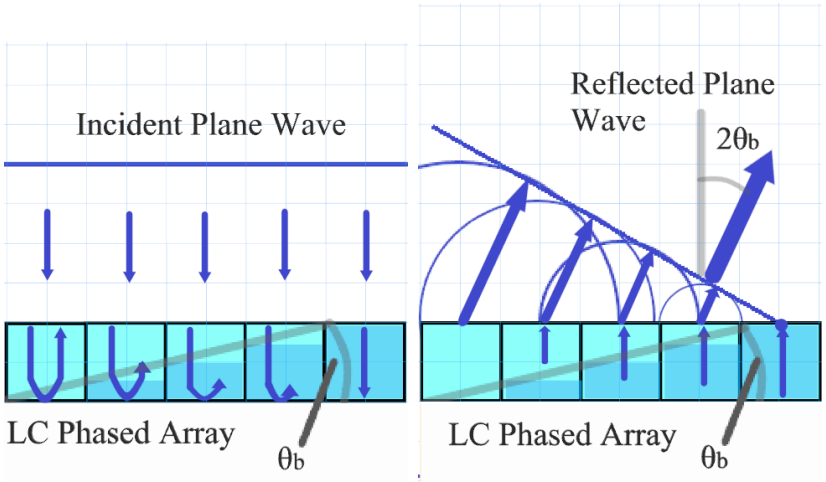
Beam steering refers to the act of changing the direction in which a beam travels. To do so mechanically, the direction that the illumination source is pointing could simply be shifted. If a beam is being reflected off of a mirror, that mirror could be rotated, and the beam would change directions. Non-mechanical beam steering, on the other hand, is the ability to steer a beam without moving a physical component. This was demonstrated using liquid crystals in 1996 [5]. The equipment used was a phased array of liquid crystals, which added a phase tilt to incident light, tilting and redirecting the reflected waves. This was done by applying a saw tooth pattern of phase delay across the array, analogous to a blazed grating.

In Figure 3, a blazed grating is shown, and in Figure 4, a phased array of liquid crystals is shown. For both the blazed grating and the phased array, the relationship between the incident and reflected angle is based on the blaze angle, $\theta_b$. At normal incidence, the reflected wave is at an angle of $2\theta_b$ from the incident wave. The reflected angle in the blazed grating can be explained by geometric optics and is also in agreement with diffractive optics, as shown in Figure 3 (b). The ability of a phased array of liquid crystals to steer a beam is not compatible with geometric optics, and it is only because of the wave properties of light that a beam can be non-mechanically steered, as in Figure 4 (b).




(a) (b)

**Figure 3. Reflection from a blazed grating. An incident plane wave in (a) is reflected from the grating at an angle in (b) based on the law of reflection, and corroborated by illustrated Huygens' wavelets which are shown to reconstruct the reflected plane wave at an angle.**




(a) (b)

**Figure 4. Reflection from a liquid crystal phased array. An incident plane wave in (a) enters the array where the individual liquid crystals apply different amounts of phase delay in the pattern of a sawtooth profile (dark blue represents a delay that doubles the optical path length). In (b), portions of the wavefront return to the surface of the crystals at different times, causing the wavefront to diffract at an angle. Huygens' wavelets are shown to reconstruct the reflected plane wave at an angle.**

## 2.6 Spatial Light Modulators

The spatial light modulator (SLM) is a piece of equipment used in many wavefront shaping applications. SLMs are either reflective, or transmissive devices that

can modulate the amplitude or phase of light in a spatially non-uniform manner across a wavefront. An SLM consists of a small screen of pixels that can be individually controlled to shape a wavefront. The mechanism of this control can be physical translation such as in deformable mirror based SLMs, or the rotation of liquid crystals to retard phase in liquid crystal on silicon (LCOS) SLMs. LCOS SLMs contain nematic cells, which perform variable phase retardation. A nematic cell, shown in Figure 5, consists of crystals suspended in fluid between two alignment layers. The alignment layers are polished such that the liquid crystals, which are long and thin, tend to align with the polish. A nematic cell is birefringent, with the index of refraction depending on the light's polarization with respect to the crystals' orientation [3]. By applying a field to twist the orientation of the crystals, the index of refraction that light sees when propagating through the cell is altered. This effect is used to variably retard the phase of a wave.

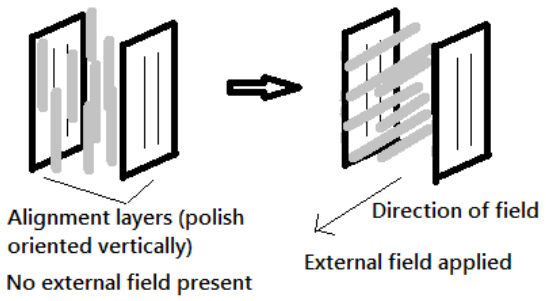



**Figure 5. Design concept of a nematic cell. The liquid crystals (gray) are initially aligned with the polish of the alignment layers, but the application of an electric field twists the orientation of the crystals.**

# III. Inverse Diffusion Theory

## 3.1 Introduction

Prior to computational advances which have allowed for the advance of adaptive optics, diffuse scattering was thought to be a limit to all imaging techniques [6]. Yet, the speckle pattern that forms after diffuse scattering contains some information about the incident light [7, 8]. In fact, diffuse scattering, though incredibly complex, is not random. In a static medium, scattering is deterministic and linear [6]. In the 1990s, the theoretical underpinnings of what is now called inverse diffusion were developed. In a paper by I. Freund, a diffuse scatterer was treated as a complex field with which incident light interferes. He theorized that if one used a number of known reference waves and observed the speckle pattern formed by each, the properties of the scatterer's complex field could be determined. Further, reference waves could be chosen that made the random scatterer act as an optical element such as a lens [8]. In this chapter, the origins of wavefront shaping with transmissive inverse diffusion, and the mathematics which describe the process are discussed. Next, specific details of how wavefront shaping algorithms work are discussed. Two classes of algorithms, called matrix methods, and iterative methods will be covered, followed by a discussion of the parameters used by iterative algorithms and their effects on algorithm performance. Finally, research into reflective inverse diffusion at AFIT is discussed.

## 3.2 Wavefront Shaping

In 2007, Ivo M. Vellekoop and Allard P. Mosk demonstrated that the diffusion of light while propagating through a turbid medium could reversed [9]. Further, the medium could be made to act as a lens, and actually improve the focus of the beam [10]. This process where the diffusion of light was reversed was aptly termed

inverse diffusion. To demonstrate inverse diffusion, they used an SLM divided into $N$ segments to shape a wavefront before sending it through a stationary, transmissive, scattering sample, with a setup as shown in Figure 6. The resulting speckle pattern was then observed on a Charge-Coupled Device (CCD). The algorithm they used to create a focused beam relied on the linearity of the process. The field at some pixel $E_m$ on the CCD was a linear combination of responses to input fields from each of the $N$ segments of the SLM, given by Equation (15).

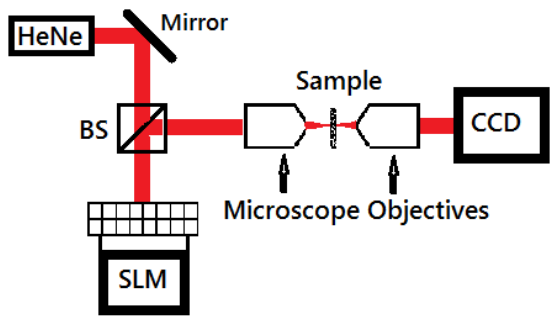



**Figure 6. Design concept for Vellekoop and Mosk's experiment demonstrating wavefront shaping to achieve transmissive inverse diffusion. An expanded beam from a 632.8 nm HeNe reflects off of an SLM, creating a shaped wavefront, which is imaged onto the scattering sample. The resulting speckle pattern is then expanded and propagates to the CCD.**

$$E_m = \sum_{n=1}^{N} t_{mn} A_n e^{i\phi_n} \tag{15}$$

where $t_{mn}$ are the elements of the transmission matrix discussed by Freund, and $A_n$ and $e^{i\phi_n}$ are the amplitude and phase from the $n^{th}$ SLM segment. This algorithm worked by individually cycling the phase of each of the $N$ SLM segments alone until a maximum intensity at the desired spot was achieved. Intensity in this context is defined in Equation (16) as the square modulus of the electric field, so at the $m^{th}$ pixel,

$$I_m = \left| \sum_{n=1}^{N} t_{mn} A_n e^{i\phi_n} \right|^2 . \tag{16}$$

At the end, the segments were set to their optimum positions, and a maximum intensity was achieved at the desired location. The metric used to measure the focused beam was termed enhancement, $\eta$, and is defined in Equation (17) as

$$\eta \equiv \frac{I_{opt}}{\langle I_{ref} \rangle}, \tag{17}$$

which is the ratio of the optimized intensity over the averaged background intensity. Vellekoop and Mosk were able to achieve an enhancement above 1000 in one sample, and they found that the maximum enhancement is proportional to the number of SLM segments used[9]. The general relationship between number of SLM segments and maximum enhancement has the form,

$$\eta_{max} = \alpha(N-1)+1, \tag{18}$$

where $\alpha$ is a constant that depends on the type of modulation used, the intensity distribution over the SLM, and the type of random scattering process [17, 6]. However, the actual variable that usually limits enhancement is called the persistence time, $T_p$, which is a measure of temporal stability of the system [11]. $T_p$ is found by calculating the autocorrelation of the field at the observation plane over a period of time. $T_p$ is the time required for the autocorrelation to reach $\frac{1}{e}$ the initial value. Autocorrelation is the correlation between a signal at an initial point in time and itself at subsequent points in time.

**Different Approaches to Wavefront Shaping.**

Vellekoop and Mosk's initial research paved the way for much of what has been done in the last 10 years in the field of wavefront shaping, but the initial research was not concerned with calculating the transmission matrix, instead it dealt with an

ad-hoc optimization of a single point. Two classes of algorithms now exist for what is called feedback-based wavefront shaping. The first class of algorithms iteratively optimize the wavefront to create a maximum enhancement at a point. The second class of algorithms first measure the complex matrix elements, $t_{mn}$, and thereby allow for the flexibility of focusing light onto more than one point [6]. It is important to understand the difference in results produced by these two methods. The iterative methods find one phase mask which is applied to the SLM and focuses scattered light to one point on the CCD. Matrix methods attempt to map the entire linear relationship between segments on the SLM and points on the CCD. Each row of a reflection (or translation) matrix is its own phase mask which focuses scattered light onto one point on the CCD, whereas each column is the phase profile on the CCD caused by the propagation of one SLM segment.

### 3.3 Matrix Calculation Methods

In order to calculate a transmission matrix (TM or reflection matrix (RM), the contribution from each SLM segment must be measured separately. One of the best methods to measure an RM or a TM is called Parallel Wavefront Optimization (PWO) [12]. A variant of this method was demonstrated by Yoon et al. in 2015, and has also been applied for measurement of a reflection matrix [13, 14]. The method separated SLM pixels into two groups, shown in Figure 7. One group was held static, in order to serve as a reference, while in the other group, the pixels were modulated at unique frequencies $\omega_1$, $\omega_2$, $\omega_3 ... \omega_{N/2}$. A Fourier decomposition of the resulting CCD output allowed for each segment's contribution to each of the CCD's channels to be singled out. PWO requires $4N + 8$ intensity measurements by the CCD to be completed, where again, $N$ is the number of SLM segments.

The advantages of calculating a TM or RM are that in the ideal case, such in-

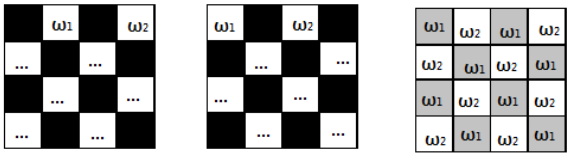



**Figure 7. Yoon's method for determining the transmission matrix, called Parallel Wavefront Optimization. The SLM segments are split into two groups. Initially one group at a time has its segments modulating at different frequencies while the other group is held constant. Last, the entire groups are modulated at two different frequencies.**

formation is reusable and is itself useful because of the information about the diffuse scatterer it contains. Once the matrix is known, arbitrary patterns can be imaged onto the CCD. However, the time required to make the matrix (which must be done first), and the necessity for a static medium are both drawbacks. Before enhancement of a desired location is performed, the whole transmission matrix must be measured. Research has shown that as time goes on, previously optimized segments tend to become suboptimal by the time the process is complete [1].

### 3.4 Iterative Methods

Iterative methods are specially designed to quickly and optimally create enhancement at a desired spot. They are feedback-driven processes that measure enhancement while modulating the phase of SLM segments to increase that enhancement. In this way, they iteratively find phase masks which create the highest enhancement.

One way to find the best phase shifts to apply to SLM segments, is to individually modulate their phases while recording the enhancement of the desired spot. This is the basis for two prominent algorithms called the continuous sequential algorithm (CSA), and the partitioning algorithm (PA) [11]. In the CSA, shown in Figure 8, the SLM segments are ordered, and one by one, sequentially, the phase of each segment is modulated over a discrete range of possible phase levels. The enhancement is recorded as a segment modulates over each phase level, and after the modulation of a segment

is complete, its phase is set to the level which produced the highest enhancement. The PA, shown in Figure 9, is very similar to the CSA. For each iteration, a random partition of the SLM segments is chosen. Like the CSA, those segments are modulated over a discrete range of possible phase levels, while the enhancement is recorded for each phase level, then the optimum phase level is assigned to the partitioned segments.

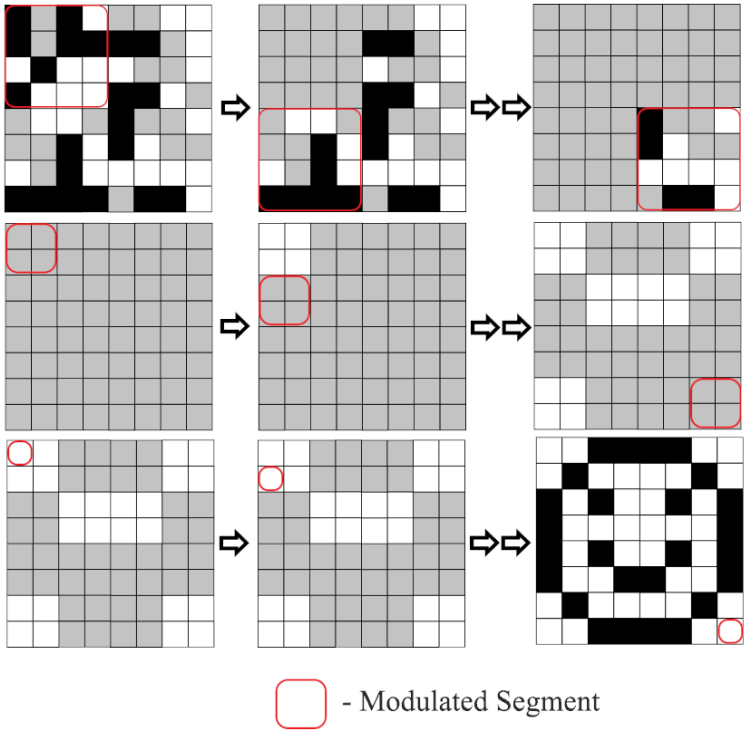



**Figure 8. The Continuous Sequential Algorithm (CSA) with preoptimization. The CSA modulates individual SLM segments through a number of chosen phase levels to determine the best phase level for each segment. Preoptimization starts off with larger segments before working down to the desired segment size**

In principle, the CSA and PA should be able to directly find the optimal phase for each SLM segment, at least to the nearest discrete phase level. In practice, they are limited by noisy testing conditions combined with low signal to noise ratios, which make discerning precise optimal levels difficult for individual or partitioned segments. The application of genetic algorithms (GAs) to iterative wavefront shaping was demonstrated by Conkey et al. for achieving enhancement after light propagation through a turbid medium [15]. These algorithms take more of a black box approach

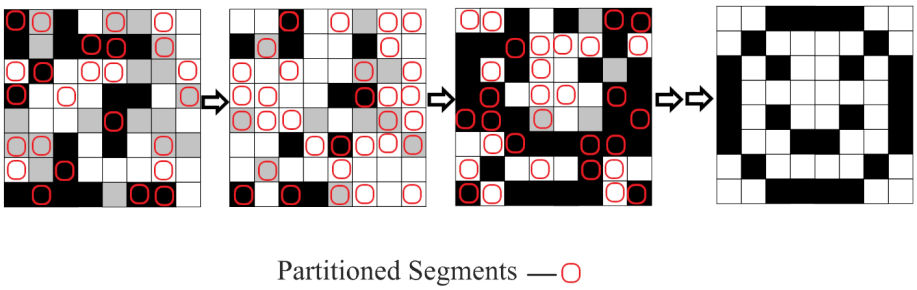



**Figure 9. The Partitioning Algorithm (PA) is an iterative algorithm for inverse diffusion. Half of the SLM segments are chosen at random to have their phases cycled simultaneously through a number of chosen phase levels. The partitioned segments are set to the best phase level then again, half of the segments are again chosen at random, continuing until a desired end goal is achieved.**

to achieve enhancement, because while still feedback driven, their main mechanism of achieving enhancement is not a programmed search for optimal phase values of specific SLM segments, but random chance and a survival of the fittest premise. Conkey's algorithm, shown in Figure 10, works by initially generating a random set of phase masks, then ranking each mask according to fitness, which is defined by the enhancement created at a desired spot. Breeding then occurs, which combines members of the population at random, with a higher chance of breeding given to the highest ranked phase masks. During breeding, selected parent masks are combined to make new composite offspring masks, but also random mutations are added to those offspring. In this way, through a combination of random chance and competition, the algorithm finds phase masks with increasing fitness. Because the GA measures the fitness created by entire phase masks, it is more noise resistant than the CSA or PA, which depend on detecting changes caused by modulations of individual segments.

### 3.5 Considerations for Iterative Algorithm Parameters

**Search Space.**

Iterative algorithms operate in a search space of $N$ variables, where $N$ is the number of SLM segments. The CSA and PA limit the search space by choosing a

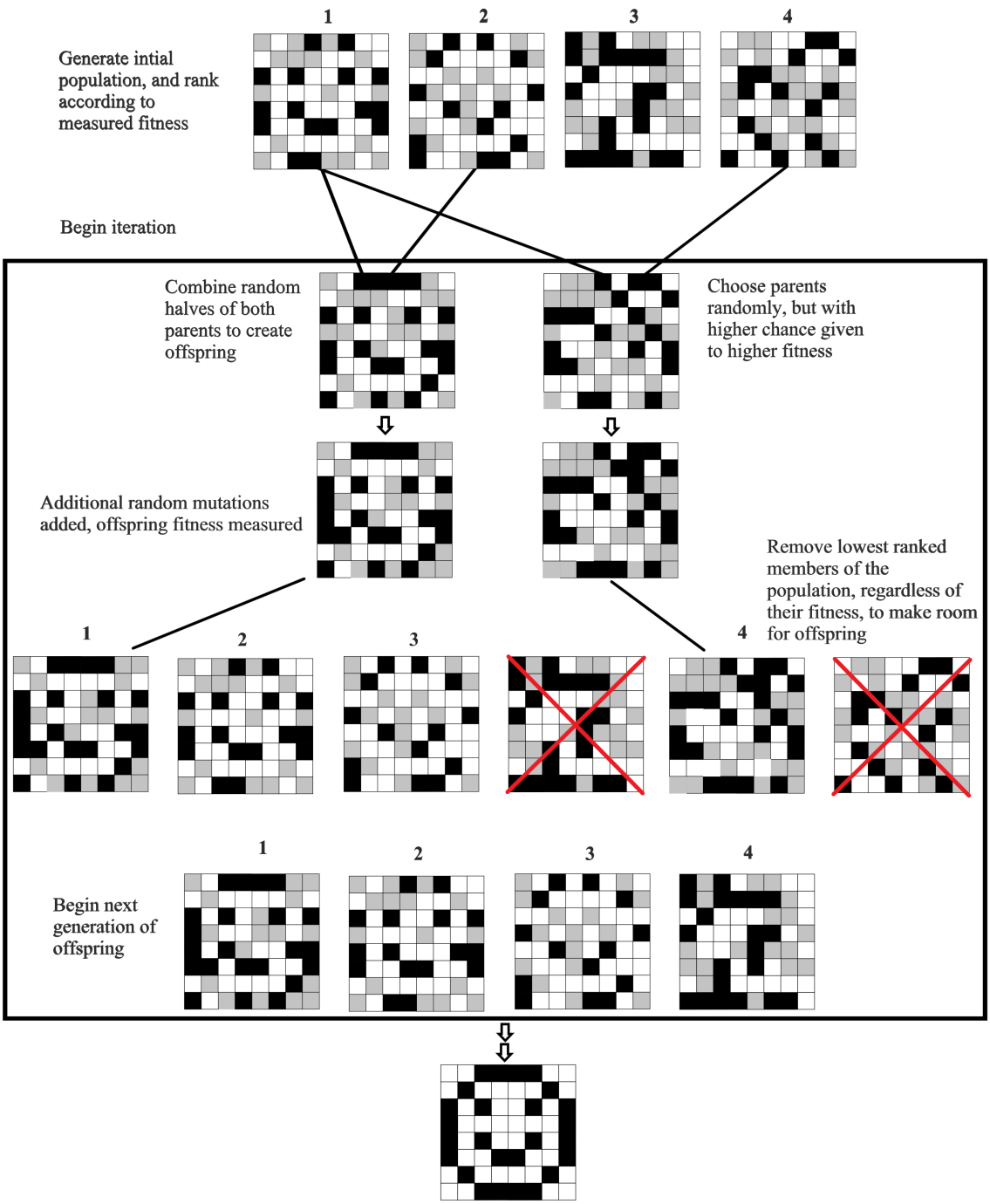



**Figure 10. The Genetic Algorithm (GA) designed by Conkey [15]. The algorithm creates a population of phase masks, then goes through generations where new offspring are bred, and the worst performing phase masks are removed. Through a survival of the fittest principle, the population becomes more fit.**

discrete number of allowed phase shift values between 0 and $2\pi$ that each segment can take. This limitation is not due to a lack of levels that the device can display. There are usually sufficiently many linearly spaced phase levels that the range of possible values could be considered semi-continuous. For example, the Meadowlark Optics Reflective XY series model P512 SLM [16] can take over $16,000$ linearly spaced phase levels. The reason that the CSA and PA must limit the number of allowed phase levels is that each level corresponds to a required intensity measurement by the algorithm for each iteration. Having more levels lengthens the time required for each iteration, $T_i$. Yet, using smaller step sizes between phase levels should increase the accuracy of a phase map, since a segment's optimal phase may not lie directly on a phase level. Evidence of this has been shown by Burgi for reflective inverse diffusion, where using 21 phase levels corresponding to a step size of $\frac{\pi}{10}$ increased the maximum intensity of a spot by 20% when compared to the use of six phase levels corresponding to a step size of of $\frac{2\pi}{5}$ [17]. It is unclear what the exact form of the relationship between phase step size and maximum theoretical enhancement is, but it may safely be asserted that there is a correlation between maximum enhancement and step size. Combining the known linear relationship between $N$ and maximum enhancement, from Equation (18), and the correlation between maximum enhancement and phase step size, the conclusion is that using a larger search space, which allows for more possible phase masks, may increase the maximum enhancement that an algorithm can achieve. However, using a larger search space comes with detriments that sometimes counteract the benefits.

**Algorithm Speed.**

Using a larger search space may cause an algorithm to take longer by either increasing the iteration time or increasing the number of required iterations. Increasing the number of phase levels will increase the iteration time, $T_i$, of the PA and CSA,

because each phase level corresponds to an intensity measurement that must be made. On the other hand, the GA does not depend on testing through a number of phase levels, instead, $T_i$ for the GA depends on the number of offspring created per iteration. Therefore, there is no obvious reason why the GA should limit the number of allowed phase levels that it uses, which is one advantage of the GA. The other aspect of the search space which can be increased is $N$. Increasing $N$ will increase the runtime of the CSA because it takes $N$ segment modulations in sequence to make a complete pass through the SLM. Unlike the CSA, the run times of the PA and GA are not dependent on $N$, in the sense that they do not operate in a sequential manner through all $N$ segments of the SLM. The PA chooses a partition of segments for modulation, and when using a partition size of 50%, regardless of how large $N$ is, after a few dozen iterations, virtually all segments will have been included in a partition, and will have been modulated at least once. Similarly, for the GA, all segments are constantly being modulated due to the random breeding and mutation process. Therefore, both the PA and GA can operate with larger $N$, without a penalty. However, only the GA can operate in a search space with semi-continuous phase levels, giving it access to a higher maximum enhancement without the associated long run time.

**Decorrelation.**

Time cost is not the only detriment caused by long run times or slow iterations. Decorrelation of the system over time places an upper limit on algorithm performance. For the CSA, if $T_p/T_i \gg N$, then the maximum theoretical enhancement is determined by Equation (18) [11]. However, if $T_p/T_i < N$, then time required to make a full pass through the SLM, $NT_i$, is longer than the persistence time of the system. For this reason, using an $N$ which is too large is detrimental when using the CSA. The PA, on the other hand, is not limited by $N$, but assuming $T_p/T_i \ll N$, the maximum

enhancement of the PA, without noise is $\eta = T_p/(2T_i) + 1$ [11]. Conkey's research showed that short persistence times decreased the performance of the GA, though the exact relationship is not known [15].

### 3.6 Reflective Inverse Diffusion

Reflective inverse diffusion is theoretically very similar to transmissive inverse diffusion, with one key difference. The diffuse scattering process is reflection from a rough surface, as opposed to propagation through a turbid media. The process is still deterministic and linear, and thus, most of the mathematics which describe the transmissive case apply to the reflective case as well. The most significant mathematical difference is that when using phase-only modulation, the theoretical enhancement for reflective inverse diffusion is

$$\eta_{max} = N, \tag{19}$$

whereas amplitude and phase modulation is required to make $\eta_{max} = N$ for transmissive inverse diffusion [17, 6]. While transmissive inverse diffusion originated with a need for medical imaging, reflective inverse diffusion was motivated by applications which could use reflected light for imaging, such as Dual Photography [2] or using a rough surface as a scattering mirror [18]. The CSA for Reflective Inverse Diffusion was demonstrated by Burgi et al. in 2016 [17], and later, he demonstrated PWO for measuring the reflection matrix [1, 14].

The optical setups used for transmissive inverse diffusion depend on optical elements before and after the interaction with the scattering sample. The applications for reflective inverse diffusion involve being able to refocus light after random scattering. If a lens could be placed near the scattering sample, or in a position so as to focus randomly scattered light, there is no reason why the laser source itself could not simply be positioned there in the first place. This would defeat the purpose of

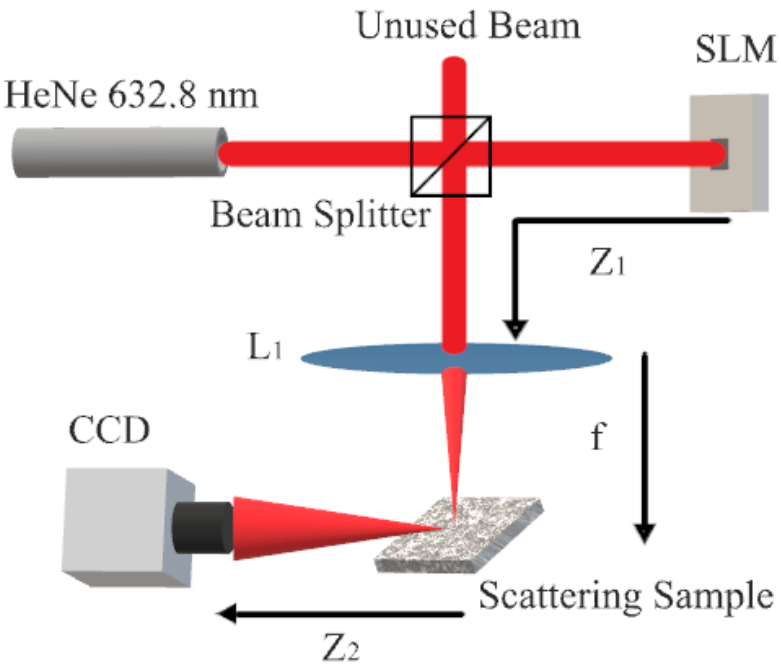



**Figure 11. Simplified diagram of focal plane setup used in reflective inverse diffusion simulations, and experimentation done by Burgi [19] [17] [14]. Polarized, collimated light is expanded and normally incident on the SLM. A beam splitter separates out modulated light reflected from the incident beam and that modulated light is then focused by lens $L_1$ onto the scattering sample and reflected to the CCD.**

reflective inverse diffusion.

Currently, two different systems for achieving reflective inverse diffusion are called focal plane systems, and image plane systems. Focal plane systems focus collimated, modulated light onto the surface of the scattering sample, as shown in Figure 11. Image plane systems, on the other hand, create a demagnified image of SLM on the scattering sample, as shown in Figure 12.

### 3.7 Summary

Wavefront shaping is a technique used for controlling light after diffuse scattering events. The goal of wavefront shaping is to achieve an effect called inverse diffusion, which occurs when the phase of light going through a scattering event is conjugated such that the scattering event focuses that light. The technique is based on the

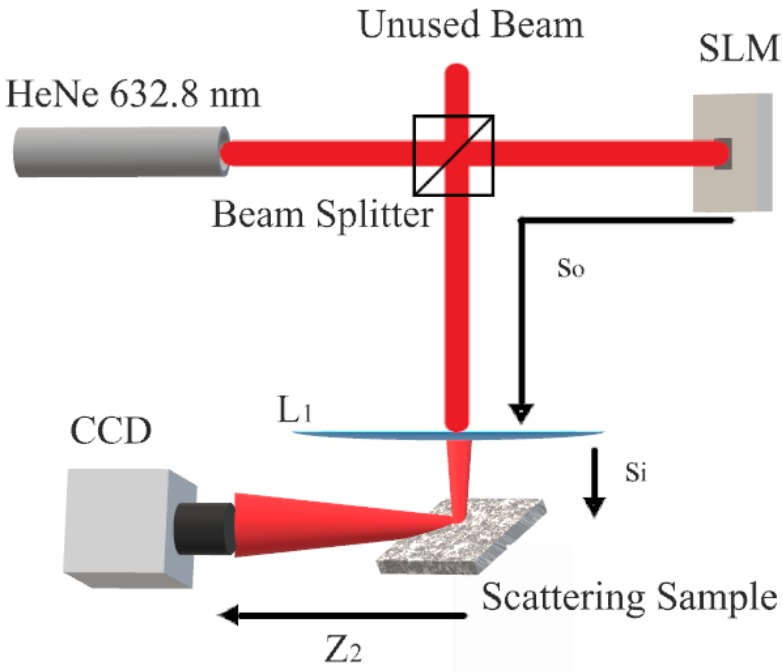



**Figure 12. Image plane setup used in reflective inverse diffusion simulations, and experimentation done by Burgi [1, 19, 14]. Polarized, collimated light is expanded and normally incident on the SLM. A beamsplitter separates out modulated light reflected from the incident beam, and the SLM is imaged by lens $L_1$ onto the scattering sample and reflected to the CCD.**

principle that light scatter is a linear, deterministic process. Wavefront shaping uses SLMs to apply the required phase shifts to the wavefronts. Initially, inverse diffusion was shown on light transmitted through turbid media, and later it was demonstrated on light reflected off of rough surfaces. Feedback-based wavefront shaping is a branch of algorithms which includes both matrix methods and iterative algorithms. These methods operate by sending modulated light through a scattering interaction and observing the scattered speckle field for changes with a detector. Matrix methods attempt to map the linear relationship of light from the SLM onto the detector, which means that information from the RM or TM can be used to focus light to any point on the detector. Iterative methods work to focus light on only point on the camera, but they produce better results than matrix methods. The genetic algorithm developed by Conkey is arguably the best iterative algorithm because it has been

shown to outperform other algorithms in noisy conditions, and it can search through a larger number of possible phase masks than other types of algorithms. Finally, reflective inverse diffusion has a potential application to dual photography, which could finally lead to the coveted goal of seeing around corners.

# IV. Reflective Inverse Diffusion

## 4.1 Introduction

Reflective inverse diffusion is the effect that is achieved by wavefront shaping algorithms that describes the refocusing of light reflected off of a rough surface. Research of reflective inverse diffusion at AFIT has been motivated by applications to Dual Photography, and a related technique called Indirect Photography [1]. Recent research into reflective inverse diffusion has favored PWO as the wavefront shaping algorithm due to the RM that it provides, which gives multiple-spot capabilities. However, the average enhancement that phase masks found by PWO create is very low compared to enhancements achieved by iterative methods. The use of a genetic algorithm, an iterative algorithm, is discussed in this chapter. First, the equipment used is described, followed by details of the experimental and simulated optical setups. Next, the mathematical model describing the propagation of light in the system is introduced. Finally, the settings and operation of the genetic algorithm are discussed, followed by a report of the genetic algorithm's performance.

## 4.2 Equipment

The SLM used was a Meadowlark Optics Reflective XY series model P512 SLM, which has $512 \times 512$ square pixels, with a pixel pitch of $15\mu m$ for a total effective area of $7.68mm \times 7.68mm$. Each pixel is capable of 16384 distinct phase levels between 0 and $2\pi$. A manufacturer provided lookup table (LUT) mapped inputs for each pixel to the correct output phase for each pixel. The SLM was controlled by libraries provided by Meadowlark Optics. The camera used was a Thorlabs model 4070M-GE-TE, which has $2048 \times 2048$ square pixels, with a pixel pitch of $7.4\mu m$. The camera was also controlled by desktop computer in Matlab®, via a Gigabit Ethernet connection,

using libraries provided by Thorlabs.

## 4.3 Optical Setups

**Simulations.**

The simulated environment matched the focal plane system setup used by Burgi in the PWO simulations, shown in Figure 11, with $Z_1 = 15$ cm, $f = 500$ mm, and $Z_2 = 50$ cm [19].

**Laboratory.**

The laboratory setup was a modified focal plane setup, simplified by removing the beamsplitter, and adjusting the angle of the SLM, shown in Figure 13. The CCD was

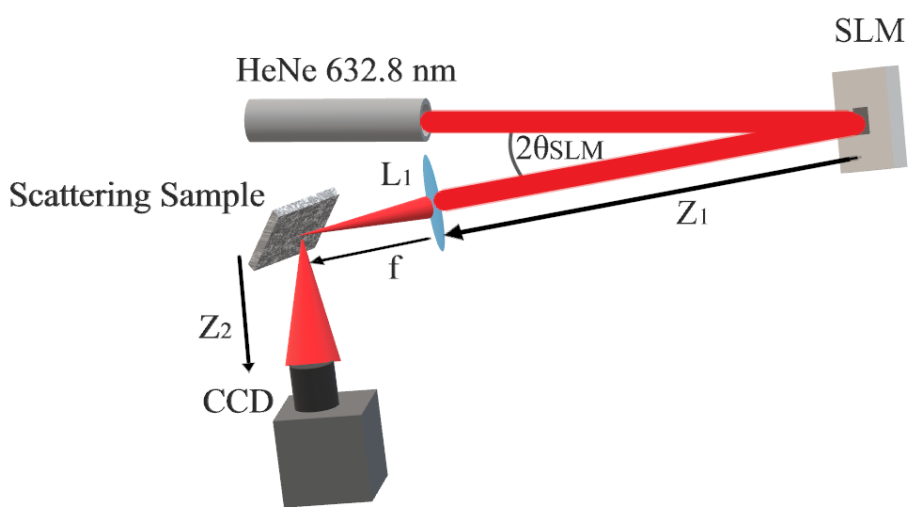



**Figure 13. Simplified focal plane setup used in reflective inverse diffusion experiments. Polarized, collimated light is expanded and incident on the SLM at a slight angle. Light reflected from the SLM is focused by lens $L_1$ onto the scattering sample and reflected to the CCD. In the lab, the incident angle of the beam onto the SLM, denoted $\theta_{SLM}$ was $3.98° \pm 0.01°$, while $Z_1 = 54 \pm 0.5$ cm, $Z_2 = 40 \pm 0.5$ cm, $f = 400$ mm.**

placed at $90° \pm 1°$ with respect to the light incident on the scattering sample, in the specular region of the reflection from the scattering sample, at an angle of $45° \pm 1°$ from the surface normal of the scattering sample. Additionally, a 400 mm lens was

used, and new propagation distances $Z_1$ and $Z_2$ of $54 \pm .5$ and $40 \pm .5$ cm were used.

## 4.4 Mathematical Model

### Diffraction Based Propagator.

To model the propagation of the light, a diffraction-based propagator was used. The propagator was written by Burgi [1], based on Voelz's *Computational Fourier Optics* [20]. The simulation of the focal plane system begins at the SLM, where the electric field is assumed to be a plane wave with unity amplitude, at normal incidence on the SLM. The light propagates through free space before being reflected, then propagated to a lens. The reflection is ignored in the simulation. The field just before the lens, $U_{L_1}$ is given in Equation (20). Equation (20) is based on Equation (12), where the source field is the field at the SLM, $U_{SLM}$.

$$U_{L_1}(x, y) = \mathcal{F}^{-1}[\mathcal{F}[U_{SLM}(x, y)]H(f_x, f_y, Z_1)], \tag{20}$$

where the leading constant phase term from Equation (12) is neglected because it does not impact the wavefront's time-averaged value. Next, the light is focused down to a diffraction-limited spot on the sample by a lens. The lens is large enough that all light reflected off of the SLM passes through it, and the effects of a pupil function would be negligible. This step is modeled by a Fourier transform with a scaled coordinate transformation, due to the Fourier transform properties of lenses [3]. The field at the sample plane, $U^-_{sample}$, prior to interaction with the surface is given by Equation (21).

$$U^-_{sample}(u, v) = e^{(i\frac{k}{2f}(u^2+v^2))}\mathcal{F}[U_{L_1}(x, y)] \tag{21}$$

where the new coordinates $u$ and $v$ are related to the spatial frequencies $f_x$ and $f_y$ of $U_{L_1}(x, y)$, $(f_x = u/(\lambda f), f_y = v/(\lambda f))$. Additionally, a constant phase term has

again been removed. Next, the interaction of the sample and the light is modeled as random phase shifts being applied at each spatial location in the modeled coordinate space. The field at the sample just after reflection, $U^{+}_{sample}(u,v)$, is given by Equation (22).

$$U^{+}_{sample}(u,v) = U^{-}_{sample}(u,v)e^{i\theta(u,v)}, \tag{22}$$

where $\theta(u,v)$ represents the spatially varying phase shifts of scattering sample. Finally, once reflected, the light propagates to the CCD in the observation plane. Rayleigh-Sommerfeld propagation is again used to model this propagation. Finally, putting all steps of the propagation together, the Fourier transform of the lens in Equation (21) cancels the inverse Fourier transform in Equation (20). The field at the observation plane, $U_{obs}$ is given by Equation (23):

$$\begin{aligned} U_{obs}(u,v) &= \mathcal{F}^{-1}\left[\mathcal{F}\left\{e^{(i\frac{k}{2f}(u^2+v^2))}e^{i\theta(u,v)}\mathcal{F}[\mathcal{F}^{-1}[\mathcal{F}[U_{SLM}(x,y)]H(f_x,f_y,Z_1)]]\right\}H(f_u,f_v,Z_2)\right] \\ &= \mathcal{F}^{-1}\left[\mathcal{F}\left\{e^{(i\frac{k}{2f}(u^2+v^2))}e^{i\theta(u,v)}\mathcal{F}[U_{SLM}(x,y)]H(f_x,f_y,Z_1)\right\}H(f_u,f_v,Z_2)\right], \end{aligned} \tag{23}$$

where $e^{(i\frac{k}{2f}(u^2+v^2))}e^{i\theta(u,v)}\mathcal{F}[U_{SLM}(x,y)]H(f_x,f_y,Z_1)$ is the field at the sample, after reflection.

**Channel Size.**

Choosing a proper size for CCD channels is important for being able to achieve the highest possible enhancement. Ideally, the channel should be approximately the same size as the diffraction-limited spot. The spot created after a beam with diameter $D$ is focused, shown in Figure 14, will have an approximate radius $q_1$ at the focus of that beam, given by Equation (24) [21],

$$q_1 \approx \frac{2\lambda f}{D}. \tag{24}$$

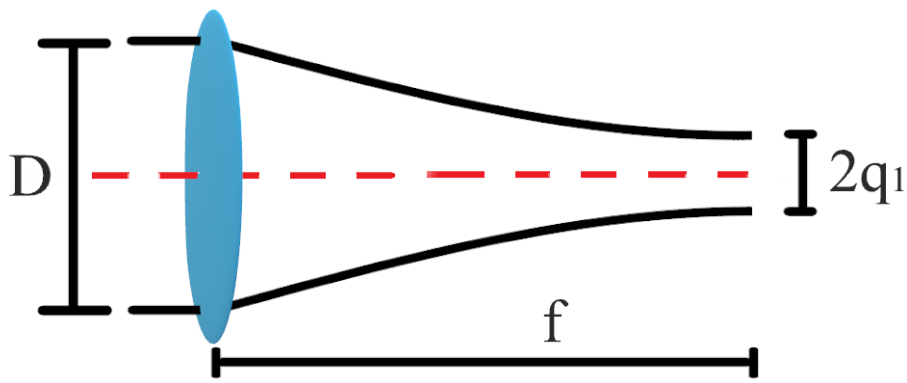



**Figure 14. The diffraction limited spot size of a focused beam.**

Under ideal circumstances, properly conjugated light will have a converging wavefront when reflected off of a rough surface. That is, the rough surface acts as a lens. In the focal plane system, this means that the beam is focused twice, once by lens $L_1$, and a second time by the scattering sample. To calculate the spot size, the proper initial beam diameter to use is that of the SLM segments, rather than the entire size of the reflected beam off of the SLM [17]. The predicted spot size, $q_2$ at the observation plane is given by Equation (25).

$$q_2 \approx \frac{Z_2}{f}\frac{D_{SLM}}{2\sqrt{N}}, \tag{25}$$

where $D_{SLM}$ is the diameter of the SLM. In practice, it was found that the best channel size was half of what is predicted in Equation (25) [1].

### 4.5 Genetic Algorithm

The algorithm used for iterative optimization was a genetic algorithm, adapted from work by Conkey et al. for transmissive inverse diffusion. A population of $P$ solutions for the SLM phase mask is initially randomly generated. Each solution is

ranked from best to worst in terms of fitness, which is chosen to be the enhancement $\eta$ achieved at a desired CCD channel. To calculate enhancement, the average intensity within the channel is divided by the root mean square (RMS) intensity of the background channels. The solutions themselves are discretized phase masks which are applied to the SLM. Every new generation, $G$ offspring are created by breeding $G$ pairs of parents from the population. Selection of solutions for breeding is probability based, and solutions with higher fitness have a higher chance of being chosen. The algorithm allows for a solution to be a parent in more than one pair, and a solution is allowed breed with itself. Breeding, shown in Figure 15, creates offspring phase masks by assigning half of the offspring's segments the values of one parent, and the remaining half the values of the other parent. To do this, a unique binary template is created for each pair of parents. The template fills half of the elements of the offspring with one parent's corresponding array elements and the remaining empty elements are filled by the other parent's corresponding array elements. Next, a random mutation of a proportion of the offspring elements occurs. This proportion is tapered as iteration is continued so that fewer segments are being randomly mutated as the solutions become more fine-tuned and large mutations would likely hurt fitness. The mutation rate, $R$ is given by Equation (26).

$$R = (R_0 - R_{end}) \exp(-n/d) + R_{end}, \tag{26}$$

for initial mutation rate $R_0$, final mutation rate $R_{end}$, generation number $n$, and decay factor $d$.

Finally, the fitness of each offspring solution is measured and the offspring are inserted into the population, giving the population P+G members. Before the next generation of offspring are produced, the bottom G members of the population in terms of fitness are removed, returning the population to P members. For the sim-

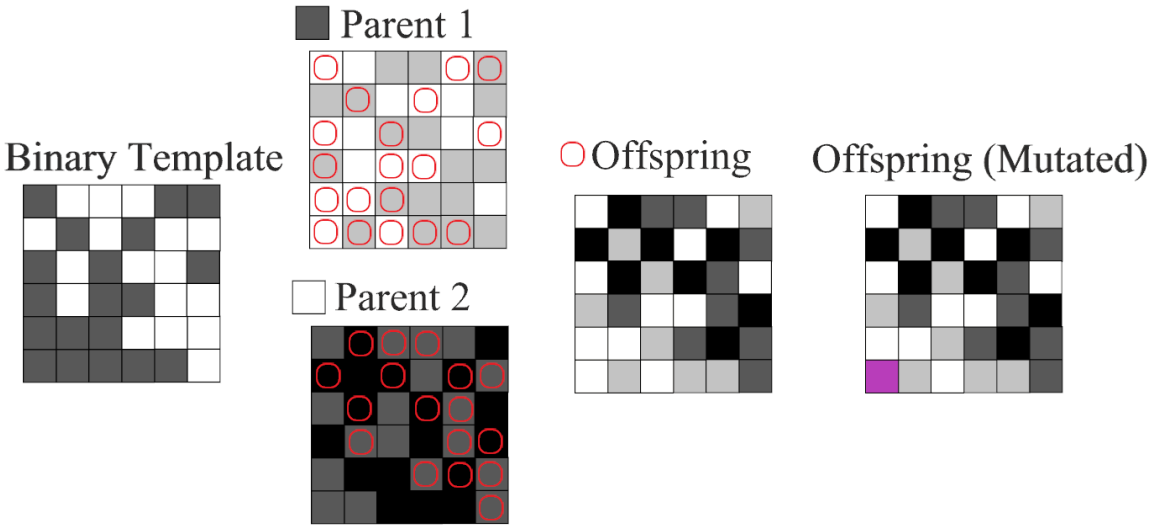



**Figure 15. The breeding process. Two phase masks from the population are chosen to be parents with higher ranked phase masks more likely to be chosen. A random binary template combines half of the elements from Parent 1 with half of the elements from Parent 2 to create an offspring, which then undergoes an additional random mutation. Each new offspring is generated in this way.**

ulations, $P = 30$, $G = 15$, $R_0 = 0.1$, $R_{end} = 0.01$, $d = 1364/3$. The values of these parameters were based on the values used in Conkey's paper [15]. The algorithm was put through 1364 generations, that number being chosen because it is the number of segments that must be optimized for the CSA to make a complete pass through an SLM divided into $32 \times 32$ segments along with preoptimization.

## 4.6 Algorithm Performances

### Simulated Results.

The simulation settings were identical to those used by Burgi, when testing for reduntant phase information in RMs [19]. The electric field is assumed to be a plane wave with unity amplitude when incident upon the SLM, therefore the field at the SLM is represented by a $512 \times 512$ element matrix, with complex valued elements of magnitude one. The reason for choosing $512 \times 512$ is that is the number and size of the SLM pixels. Each pixel is a square $15\mu m$ pixel, which makes the sampled area

7.68 mm $\times$ 7.68 mm. Prior to propagation, zero padding was used to double the dimensions of the sampled area at the SLM, to 15.36 mm $\times$ 15.36 mm at $1024 \times 1024$ points. From Equation (7) and the coordinate scaling of lens-based optical Fourier transforms, the spacing of points on the reflector sample plane and on observation plane is given by Equation (27).

$$\begin{aligned} \Delta u &= \frac{\lambda f}{L_x} \\ \Delta v &= \frac{\lambda f}{L_y}, \end{aligned} \tag{27}$$

where $L_x$ and $L_y$ are both $15.36mm$, $\lambda = 633nm$, $f = 500mm$, making $\Delta u = \Delta v = 20.61\mu m$. This made the total lengths of the simulated sample and observation planes $L_u = L_v = 21.1mm$. The sample was modeled as a unit amplitude field with uniformly distributed phase shifts. Decorrelation was not included in the simulations. The simulations were run in Matlab® using a graphics processing unit (GPU)with the GPU computing toolbox, which itself uses the NVIDIA CUFFT library which is optimized for computing FFTs on a GPU. Commented code for the propagation is available in Appendix A. The GA and PA were implemented with $N = 16384$, or $128 \times 128$ segments, each four pixels, or $60\mu m$ wide, and a channel size of $2 \times 2$ points, or $41.22\mu m$ wide. The CSA was implemented with $N = 1024$, or $32 \times 32$ segments, each sixteen pixels, or $240\mu m$ wide and a channel size of $8 \times 8$ points, or $164.88\mu m$ wide. For the PA and CSA, 15 phase levels for each segment were used, and for the GA, 15 offspring were generated each generation, making the number of required intensity measurements for one iteration of the PA and CSA equal to the required intensity measurements for a generation of the GA. The CSA was run with one complete pass through the SLM divided into $32 \times 32$ segments, but with preoptimization, which iterates through the SLM at $2 \times 2$, $4 \times 4$, $8 \times 8$, and $16 \times 16$ segments, in total, this required 1364 segment optimizations. The PA and GA were

iterated 1364 times in order to match the number of required intensity measurements of the CSA with preoptimization. Noise was simulated with a random $20\mu m$ (1 pixel) vibration, as well as a random 5% error in the observed intensity at the observation plane, and an error of $\frac{\pi}{4}$ of the phase for each SLM pixel. Additional time-dependent decorrelation was not accounted for in the simulation. Commented code for the iterative algorithm simulations is available in Appendix B.

Figure 16 shows iterative algorithm performance, averaged over five trials, compared with the simulated performance of PWO with $N = 1024$ under identical experimental parameters [14]. Overall, the range of intensity measurements was not

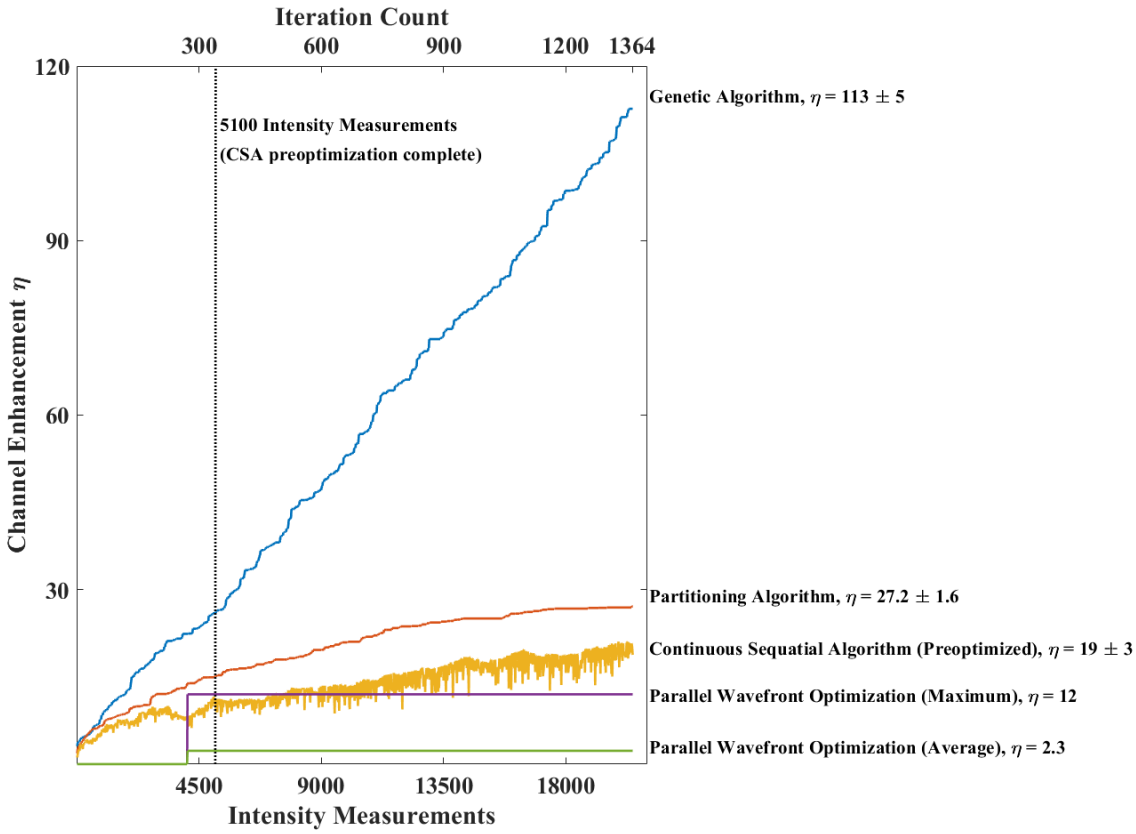



**Figure 16. Simulated performance of different reflective inverse diffusion algorithms.**

sufficient for any of the algorithms to approach the maximum theoretical enhancement of $N$ predicted by Equation (19), which was 16364 for the PA and GA, and 1024 for the other methods. The GA immediately outperformed the other iterative

algorithms. Throughout the range of 1364 generations, the increase of enhancement with the GA followed a linear trend, despite the mutation rate $R$, changing from 10% to 1% over that time. The rate of increase of enhancement with the PA tapers off despite no changes to the parameters of the algorithm itself as iterations continue. Conkey's simulated results showed that for the transmissive case, the enhancement created by the CSA exhibited parabolic behavior with iteration, but at high noise levels, the algorithm had trouble optimizing [15]. At 5100 intensity measurements, preoptimization of the CSA finishes, and the $32 \times 32$ segment optimization begins. There is an immediate increase in the amplitude of noise fluctuations. It appears that the noise level is sufficiently high to flatten the parabolic behavior, as was seen in Conkey's experiments. The reason that the CSA has visible fluctuations while the PA and GA monotonically increase is that it has no backtrack mechanism. Momentary noise fluctuations cause unoptimal phase levels to appear to be better than the true optimal levels. Without a backtrack mechanism, the algorithm cannot undo a loss of enhancement caused by an unoptimal choice, so losses of enhancement remain until the algorithm makes an entire pass through the SLM and has a chance to re-optimize the unoptimized segment once more. PWO optimized to 1024 observation plane channels simultaneously, giving a large quantity of information [14]. However, at the 4104 intensity measurements required by PWO, the average channel enhancement of $\eta = 2.3$ was outperformed by all iterative methods, which only optimized to one channel. Interestingly, the channel with the maximum enhancement created by PWO outperformed the CSA at an equal number of intensity measurements. Unfortunately, the enhancements created by each mask from PWO are not known until measured, unlike the iterative algorithms which always have enhancement available. Therefore, it is possible that PWO could create a better phase mask than the CSA for a specific channel, but on average it will not. Further, it takes additional intensity

measurements to determine the enhancement created by each mask that PWO finds, which is not shown in Figure 16.

**Experimental Results.**

The GA, PA, and CSA were tested on a 600 grit polished nickel sample, as well as a white, zinc oxide paint sample on glass. The optical setup was covered during testing to prevent ambient light from being picked up by the camera and impacting test results. Additionally, the laboratory lights were turned off during testing. No vibration or other noise was intentionally added to the test environment. As in the simulation, $N = 16384$ was used for the GA and PA, and $N = 1024$ was used for the CSA. For the CSA and PA, 15 phase levels were used, and 15 offspring per iteration were used by the GA. The time required to measure 1364 iterations for the algorithms was approximately 100 minutes. Prior to beginning measurements, the equipment was allowed to warm up for 1 hour.

Figure 17 shows the performances of the different algorithms averaged over five trials. For the PA, and GA, the results show a small period of rapid increase, after which the behaviors of the PA and GA follow the trends seen in simulation. The small jump was more pronounced on the polished nickel than in the paint sample. The CSA was unable to achieve any lasting increase of enhancement over the starting value for both samples. Higher final enhancement is achieved by the GA and CSA with the brushed nickel sample than with the paint sample. The final enhancement achieved by the PA is also higher for the brushed nickel, but the results are within the margin of error for both samples. PWO was not tested. However, for reference, in a similar experiment, using the focal plane setup in Figure 11, the average and maximum enhancement for the 600 grit sample using PWO were $\eta = 1.3$, and $\eta = 6.0$ respectively [14]. It is not clear why the CSA was unable to achieve any increase of

enhancement, while the GA and PA were. The PA, which is conceptually very similar to the CSA, tested the same 15 phase levels that the CSA used, and was able to enhance the tested channel. Even during preoptimization, when simulation showed lower noise amplitude, the CSA did not achieve any increases in the lab.

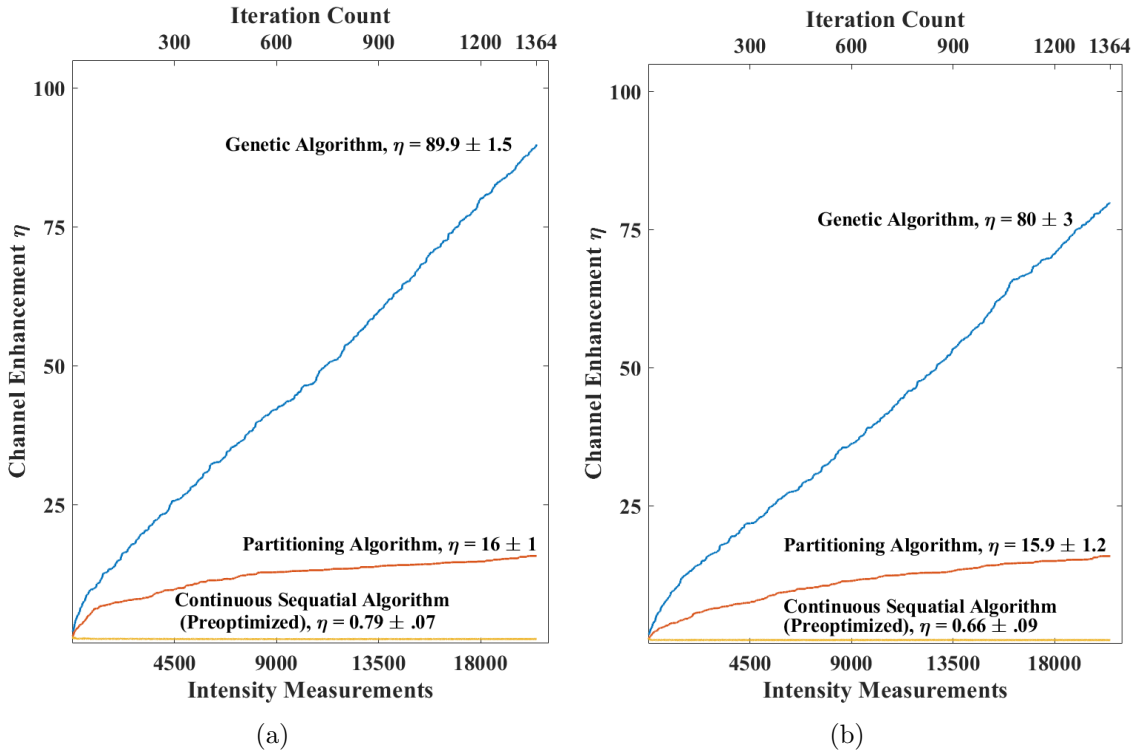
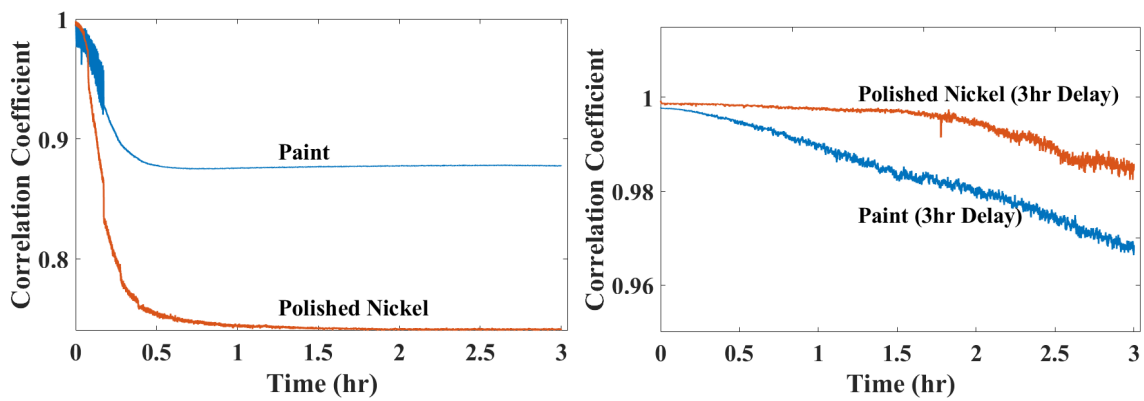



**Figure 17. Enhancement achieved using different techniques for reflective inverse diffusion. In (a), the performance on 600 grit polished nickel is shown. In (b) the performance on zinc oxide white paint is shown.**

The system's tendency to decorrelate with time was also investigated. The autocorrelation of the samples is shown in Figures 18 (a) and (b). A randomized SLM phase mask with $N = 16384$ was written to the SLM, and autocorrelation of the resulting speckle intensity on the CCD was calculated over time. Prior to each intensity measurement required for the autocorrelation calculation, the same SLM phase mask was rewritten to the SLM. In (a) the correlation coefficient is computed beginning at the time when the CCD, the SLM, and the laser are powered on. The correlation coefficient decreases rapidly within the first hour that the equipment is powered on.

After the first hour, the rate of decrease levels off to a comparatively slow pace. In (b), the system has been allowed to warm up for 3 hours before autocorrelation is calculated. The persistence time of the system, which is the time required for the correlation coefficient to reach $\frac{1}{e}$ of the initial value, depends on both the scattering sample and how long the system has been allowed to warm up. The length of time tested was not sufficient to determine the actual persistence times, $T_p$, for either of the materials. However, relating the autocorrelation results to the iterative algorithm tests, $T_p$ must be much longer than the 100 minutes required by each algorithm for 1364 iterations. This means that decorrelation did not significantly impact the performance of the iterative algorithms.




(a) (b)

**Figure 18. The autocorrelation of the field at the observation plane over time in the focal plane system using both scattering samples. In (a) the autocorrelation is computed starting at the time when all of the equipment is powered on. In (b) the autocorrelation is computed starting after the equipment has warmed up for 3 hours.**

## 4.7 Conclusions

In simulation, and in a controlled laboratory setting, iterative algorithms outperformed the average channel enhancement found by PWO. This was to be expected, given the requirement that matrix methods have for a static medium. Overall, the simulated results for the iterative algorithms predicted the general trends shown by

the experimental results, with one large exception- the CSA was unable to perform in the laboratory, likely due to a high level of noise. Nevertheless, the other algorithms did perform. The GA was the best iterative algorithm over the range of intensity measurements chosen. Impressively, the enhancement from the GA showed linear growth over the entire range of measurement, indicating the potential for even further increase of enhancement, though Conkey's research showed that the rate of increase of enhancement of the genetic algorithm does round off eventually [15].

Going forward, the GA has been shown to be well-suited for realistic laboratory conditions, more so than other existing algorithms. However, further design testing of the GA could still find design choices that would improve it. Based on the fact that the core aspect of the GA is the breeding process, suggested test areas would involve changing the processes for selection of parents, the selection of which segments each parent provides the offspring, and the random mutation rate. In the next chapter, which deals with non-mechanical beam steering, the refocused beams that the genetic algorithm creates are the beams which are used in the testing of non-mechanical beam steering.

# V. Non-Mechanical Beam Steering in Reflective Inverse Diffusion

## 5.1 Introduction

In recent research, simulation has shown that there is redundant phase information in reflection matrices [19]. Specifically, phase masks required to focus light onto different channels in the observation plane share similarities. This is significant, because prior to this research, it was believed that unique phase masks were required to focus to different points on the observation plane. The research showed this redundant information by using PWO, which finds phase masks required to focus light to every channel in the observation plane. When phase masks used to focus light to adjacent channels on the observation plane were propagated to the rough sample in simulation, the difference in the phases of the propagated waves was only a phase tilt. This suggests that a phase mask could be decomposed into an underlying phase front which conjugates the rough surface to create a focused beam and phase tilts that determine where that beam goes. Phase tilts are the mechanism behind non-mechanical beam steering. The simulations were applied to both the image plane, and focal plane systems shown in Figures 11 and 12. The implications of this research were that a reflection matrix could be constructed out of a single phase mask, which also means that the ability to focus to one point on the observation plane gives the ability to focus to other points as well. However, the reusability of a phase mask depends on being able to create phase tilts at the scattering sample.

In this chapter, the process for performing non-mechanical beam steering of a refocused beam in reflective diffusion is discussed. First, the creation of phase tilts at the scattering sample is discussed, followed by a discussion of the required shifts of the SLM phase mask. Next, limitations of the circular shift, which is used to

shift the SLM phase mask, are discussed. Finally, the methods and results of the non-mechanical beam steering procedure are reported.

## 5.2 Creating Phase Tilts at the Scattering Sample

Making phase tilts at the scattering sample depends on the optical setup being used. The image plane system is a special case because the SLM is projected onto the scattering sample. This means phase tilts at the SLM are directly projected to the scattering sample [19], which in turn steers the enhanced spot in the observation plane. For a focal plane system, the same relationship does not apply. However, the propagation of light from the SLM to the scattering sample involves a Fourier transform relationship. We recall the shift property of Fourier transforms in Equation (4) indicates that a translation in one domain corresponds to a linear phase tilt in the other. The Fourier transform occurs between the plane just prior to the lens, to the surface of the scattering sample. A translation in the position of the field incident on the lens, $U_{L_1}$, would create a phase tilt at the spot on the scattering sample, without moving that spot. Applying Equations (4) and (5) to Equation (20), Equation (28) shows that a translation of the field in the plane of the SLM, $U_{SLM}$ would also translate the field at the lens, $U_{L_1}$.

$$U_{L_1}(x,y) = \mathcal{F}^{-1}[\mathcal{F}[U_{SLM}(x,y)]H(f_x,f_y,Z_1)],$$

$$\mathcal{F}[U_{L_1}(x,y)]e^{-i2\pi(f_x a+f_y b)} = (\mathcal{F}[U_{SLM}(x,y)]H(f_x,f_y,Z_1))e^{-i2\pi(f_x a+f_y b)},$$

$$\mathcal{F}[U_{L_1}(x-a,y-b)] = \mathcal{F}[U_{SLM}(x-a,y-b)]H(f_x,f_y,Z_1),$$

$$U_{L_1}(x-a,y-b) = \mathcal{F}^{-1}[\mathcal{F}[U_{SLM}(x-a,y-b)]H(f_x,f_y,Z_1)], \tag{28}$$

for shifts $a$ and $b$. This means that a translation of $U_{SLM}$ is required in order to

create a shift in $U_{L_1}$, which will then create a phase tilt at the reflector. Beam steering could not be used to translate $U_{L_1}$, because although it could move where the beam is incident on the lens, it would also introduce a tilt into the field, which, again, by the shift theorem, would change the position of the focused spot on the scattering sample. If the light incident on the scattering sample changes position, the interaction as seen in Equation (22) will change, and any phase masks will be invalidated [19].

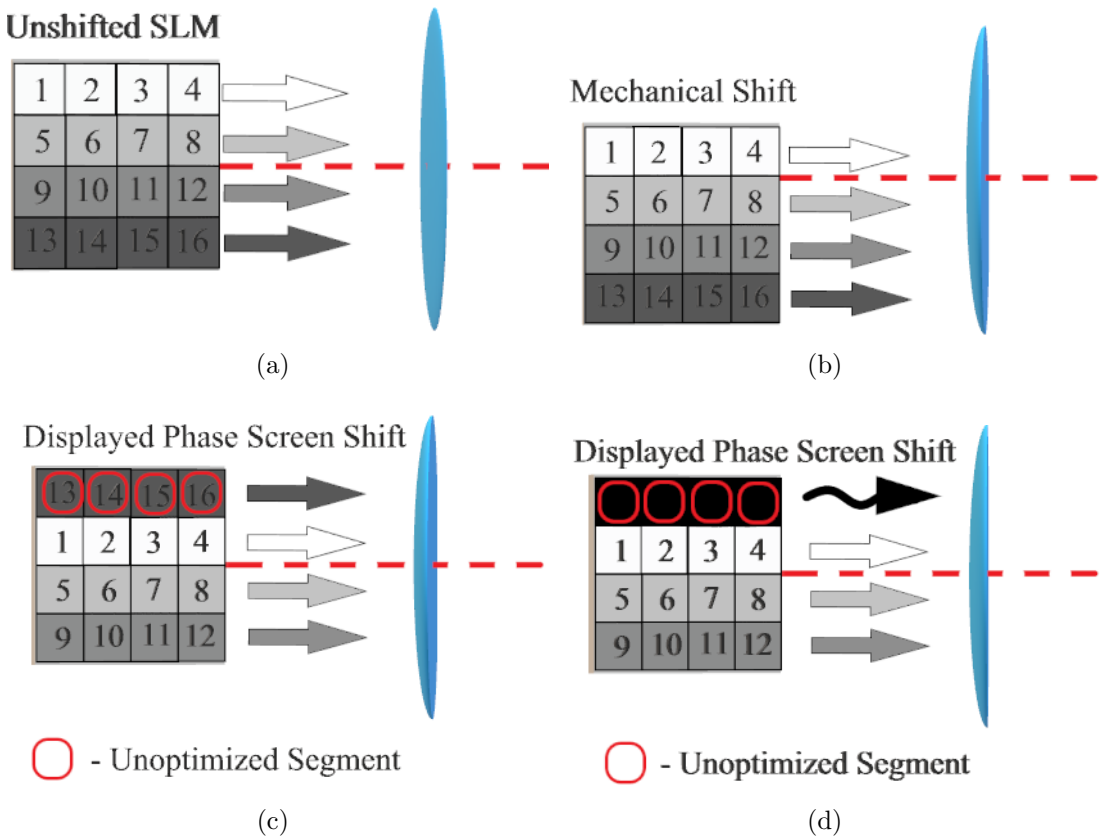



**Figure 19. Demonstration of different methods to create a shift in the field incident on the lens. In (a) an unshifted SLM and the generated wavefront are shown. In (b), a mechanical shift physically moves the SLM, and the wavefront. In (c) and (d) the phase mask displayed by the SLM is shifted, which also shifts the wavefront with reduced efficiency. In (c) the phase mask shift is a circular shift. In (d) the phase mask shift is a linear shift, and vacated pixels are randomized.**

### 5.3 Creating Spot Displacement

It has been established that a translation of $U_{L_1}$ would require a translation of the field at the plane of the SLM, $U_{SLM}$. This would require that the entire phase mask generated by the SLM, represented by Figure 19 (a), be shifted in position and still be illuminated. However, the SLM only modulates light where it has illuminated pixels, and all illuminated pixels should already be used in the phase mask. Physically translating the SLM using mechanical means would work to shift $U_{SLM}$, assuming the laser source is large enough to completely illuminate the SLM after translation, as seen in Figure 19 (b). Yet, there is a simpler solution that does not require any additional equipment. Electronically, the phase mask displayed by the SLM can be shifted, but because the SLM is finite, not all of the screen can be displayed. Additionally, this means that only discrete shifts are allowed. When the phase mask on the SLM is shifted, it leaves behind vacated pixels which are unoptimized, as seen in Figure 19 (c) and (d). There is no way to know what phase values to apply to vacated pixels so that they are optimized to create enhancement at the new shifted spot. Yet, the light which reflects off of them still contributes to the field at the observation plane, so some consideration of what phase values to apply to them is needed. Vacated pixels could simply be randomized, however they would then spread out the light reflected off of them as background speckle in the observation plane. A better option would be to somehow control the light that is reflected off of the vacated pixels so that it does not add to the background. A circular shift of the SLM phase mask achieves this purpose by reusing the portion of the phase mask which is shifted off of the surface of the SLM, as seen in 19 (c). Elements 1-12 are shifted 1 position down, while elements 13-16, with nowhere to go, are shifted three positions up to the vacated positions. The field from elements 1-12 will propagate to the lens and will have a phase tilt after being Fourier transformed; this will steer the chosen optimized

spot in the observation plane. Elements 13-16 are still technically optimized to focus to a spot, but they have a different shift, which means when the field from elements 13-16 propagates to the lens and is Fourier transformed, it will have a different tilt than elements 1-12, and will be steered in another direction, and will not contribute to background speckle. The downside of the circular shift is that though the energy does not become randomly scattered as background, that energy is focused to a second, undesired spot.

## 5.4 Limitations of the Circular Shift

**Loss of Enhancement.**

As mentioned in the previous section, using circular shifts creates vacated pixels which are not optimized to the new, displaced spot position, but instead, to a secondary, undesired spot position. As the shift becomes larger, fewer and fewer pixels will actually be optimized to the desired spot, and instead, the energy will be directed toward the undesired spot. Neglecting the contribution of a secondary, unwanted spot to the background intensity, if it is assumed that on average, each SLM pixel contributes the same amount to the spot enhancement, then the shifted spot enhancement will be proportional to the number of still-optimized pixels. Equation (29) gives the upper limit of enhancement that can be created after a shift.

$$\eta = \eta_{max} \frac{N_{optimized}}{N} \tag{29}$$

where $\eta_{max}$ is the enhancement of the undisplaced spot, and $N_{optimized}$ is the number of segments, or fractions thereof that are optimized to the displaced spot.

**Discrete Displacement Size.**

One limitation of non-mechanical translation of phase maps is that only discrete pixel shifts are allowed. This means that only discrete phase tilts can be created, and an enhanced spot can only be displaced to certain positions in the observation plane. Assuming normal incidence and the focal plane system shown in Figure 11, the shift theorem from Equation (4) indicates that a shift of the phase map of $\Delta y$ will produce a phase tilt with a slope, $m_v$, given in Equation (30) as,

$$\begin{aligned} m_v &= -\frac{2\pi}{\lambda f}\Delta y \qquad \left(\frac{radians}{m}\right) \\ &= -\frac{\Delta y}{f} \qquad (unitless)\,. \end{aligned} \tag{30}$$

The angle of this tilt, which shall be denoted $\theta_v$ is given in Equation (31) as

$$\theta_v = \tan^{-1}(m_v). \tag{31}$$

Using the small angle approximation for the displacements on observation plane, the displacement of the spot, $\Delta v$ is given by Equation (32) as

$$\Delta v = Z_2\theta_v, \tag{32}$$

where $\theta_v$ is in radians, and $Z_2$ is the distance from the scattering sample to the observation plane. In Figure 13, the SLM is rotated with respect to the incident light in order to send the reflected light to the lens $L_1$, but this means that the surface of the SLM is also rotated with respect to the surface of $L_1$. For this reason, a correction factor can be applied for horizontal shifts, which is along the $x$ axis, whereas vertical shifts are along the $y$ axis, which is also the axis about which the SLM is rotated. The

vertical axis of the SLM is parallel to the vertical axis of the lens, so no projection is necessary, but for the horizontal axis, a projection factor must be used, which modifies Equation (30) and the subsequent equations appropriately. The slope of the phase tilt when projection is accounted is given by Equation (33).

$$m_u = -\frac{\Delta x}{f}\cos(\theta_{SLM}), \tag{33}$$

where $\theta_{SLM}$ is the angle between the plane of the lens, and the surface of the SLM.

## 5.5 Assessing Shift Performance

### Circular Shift Procedure.

Phase maps were generated using the genetic algorithm described in Chapter IV. The form in which the phase masks are sent to the SLM is as a $512 \times 512$ matrix. The desired circular shift was applied to the phase mask matrix, then the shifted mask was written to the SLM. The displacement of the spot in the observation plane was dependent on the size of the circular shift of the phase mask and did not correspond to the division of the observed area into channels. In order to accurately track the position of the spot, an approximate displacement per shift was determined. Within a square window two channels wide, centered on the estimated spot position, the centroid of the intensity distribution was calculated. The new shifted channel of the spot was centered on the calculated centroid. The intensity inside the shifted channel was then used in the calculation of enhancement. The commented code for the beam steering simulations is available in Appendix C.

The SLM matrix was shifted in increments of 16 positions. In simulation, the SLM was shifted up to 256 positions in any direction. At that point, the number of vacated pixels was equal to the number of optimized pixels, which created a secondary

spot of equal enhancement as the primary, desired spot. In experiment, shifts from five channels in the observation plane were tested. Four of the channels were at the centers of each quadrant of the observation plane, and one channel was at the center of the observation plane, as shown in Figure 20. This allowed for testing shifts multiple times with different phase maps, without having to adjust the area of incidence of the light onto the scattering sample. Shifts were limited to 208 positions in any direction in laboratory experimentation, because full shifts to 256 positions caused the spot to be displaced entirely off the screen of the CCD for the four corner channels. In the laboratory experimentation, both the polished nickel, and the white paint samples were tested.

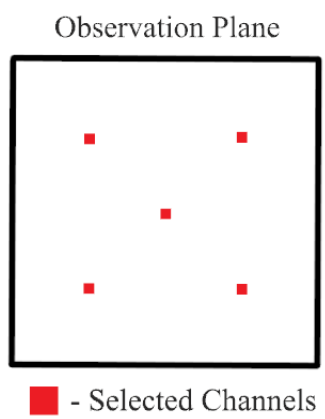



**Figure 20. Diagram of the locations of the five channels chosen for optimization in laboratory experiments.**

### Simulated Results.

Figure 21 shows the displaced spot enhancement, normalized by the maximum enhancement, $\frac{\eta}{\eta_{max}}$ of the simulated focal plane system. The normalization allows for better comparison between samples and simulation, where the $\eta_{max}$ is different. A contour line at 0.2857 is included, because in the redundant phase information paper, $\eta > 100$ was reported with displacements greater than $4mm$, for $\eta_{max} = 350$, which corresponds to $\frac{100}{350} = 0.2857$ [19]. The sampled area does not reach far

enough along the $u$ or $v$ axes to show $4mm$, but the distance along the diagonals is greater than $4mm$. It is worth noting that the decrease in enhancement as a function of displacement is not circularly symmetrical. The contours extend along the axes further than along the diagonals. Considering Equation (29), the number of optimized segments determines the upper limit of enhancement. Therefore, considering only movements along the axes are allowed, the Cartesian distance of a displacement is not what determines the loss of enhancement, but rather the number of shifts of the SLM in each direction in order to make that displacement.

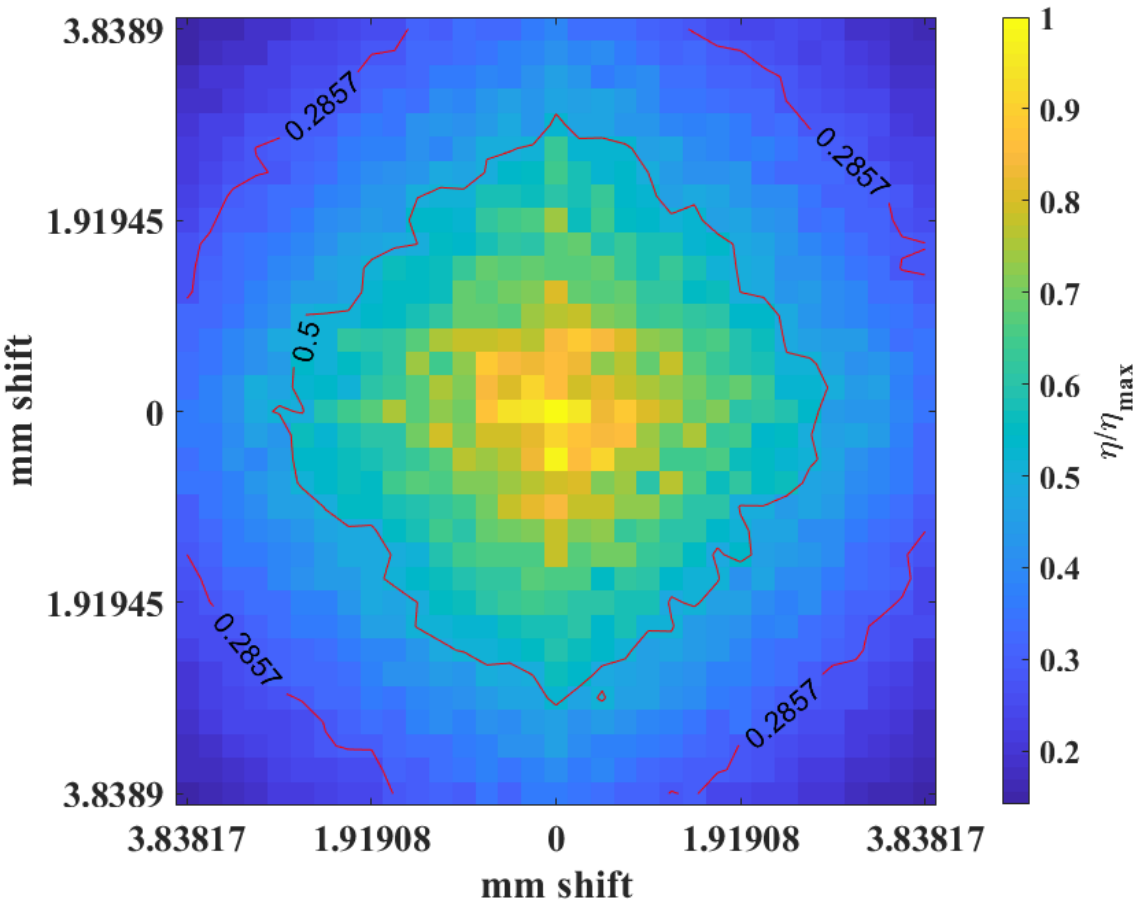



**Figure 21. Simulated enhancement of displaced spot as a two dimensional function of displacement size after a circular shift of the SLM phase map, of focal plane system shown in Figure 11. This demonstrates the simulated efficiency of using a single phase map to create focused spots at different positions in the observation plane. Results were averaged over 50 trials. The maximum value of $\eta$ prior to normalization was $113 \pm 5$.**

**Table 1. Predicted and Simulated Minimum Displacement Sizes**

| Displacement Type | Equation (32) Predicted | Experimental Displacement |
|---|---|---|
| $\Delta u$ | $15.00\mu m$ | $14.99\mu m \pm 0.03\mu m$ |
| $\Delta v$ | $15.00\mu m$ | $15.00\mu m \pm 0.03\mu m$ |

In Table 1, the minimum possible displacements along each axis are shown alongside the minimum shifts predicted by Equation (32). Because there was normal incidence on the SLM in the simulated system, the predicted $\Delta u$, and $\Delta v$ were the same. The simulated results matched the expected results from Equation (32).

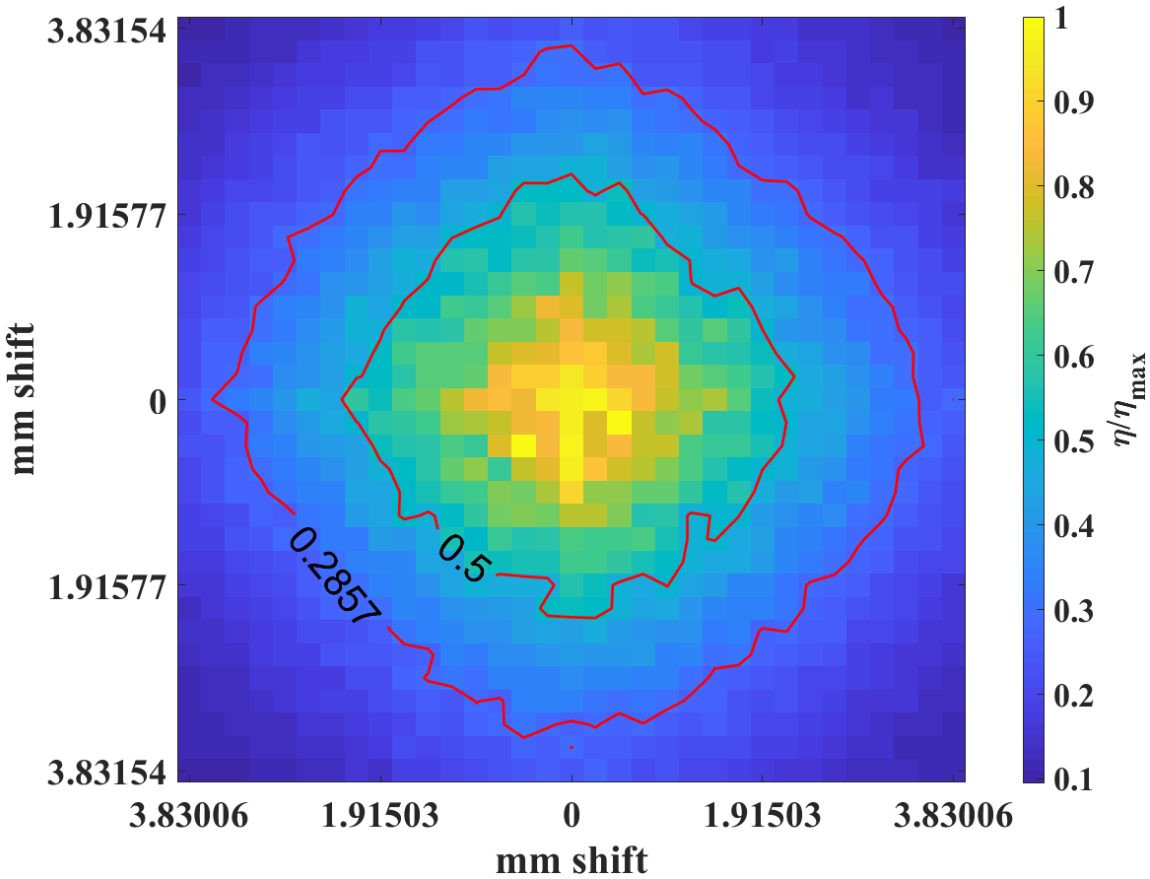



**Figure 22. Simulated enhancement of displaced spot as a two dimensional function of displacement size after a circular shift of the SLM phase map, of focal plane system shown in Figure 11. The sample is placed 5mm past the focal plane of lens. The decreased efficiency compared to Figure 21 demonstrates the impact of incorrect lens placements, which result in decrease shift performance. Results were averaged over 50 trials. The maximum value of $\eta$ prior to normalization was $109 \pm 4$.**

The effect of an error in the placement of a sample, such that the sample is not precisely at the focus of the lens, was also investigated in simulation. This was done by slightly modifying the propagator developed in section 4.3. Instead of modeling the propagation of light from the lens to the sample as a Fourier transform, as shown in Equation (21), it was modeled as a Fourier transform followed by a short Rayleigh-Sommerfeld propagation. This modification changes the form of the field incident on the sample, $U^{-}_{sample}$, which is now given by Equation (34), as

$$U^{-}_{sample}(u,v) = \mathcal{F}^{-1}[\mathcal{F}[e^{(i\frac{k}{2f}(u^2+v^2))}\mathcal{F}[U_{L_1}(x,y)]]H(f_u,f_v,\Delta Z)], \tag{34}$$

where $\Delta Z$ is the distance the sample is displaced from the focus of the lens. Figure 22 shows the displaced spot enhancement, normalized by the maximum enhancement, $\frac{\eta}{\eta_{max}}$ of the simulated focal plane system, when the sample is placed $5mm$ past the focal plane of the lens. We note that the maximum enhancement of the non-ideal system is 109$\pm$ whereas the maximum enhancement of the ideal system is $113 \pm 5$. These values both fall within the common margins of error, so there was no statistically significant difference in the maximum enhancement achieved. In Figure 22, the contour lines denoting values of 0.5, and 0.2857 are receded toward the center when compared to Figure 21. The value of .2857 is achieved at approximately $3.6mm$ with on-axis displacements. The cause of this decreased shift efficiency is the breakdown of the Fourier transform relationship between the SLM phase mask and the field incident on the sample. The phase angle of a linear phase tilt would cycle by linearly increasing from $-\pi$ to $\pi$, then instantaneously dropping back to $-\pi$. Figure 23 (a) shows a phase tilt at the rough surface as phase angles. The phase tilts are created by an 8-pixel horizontal shift of the SLM phase mask when $\Delta Z = 0$. The phase angles are extracted by calculating the difference between the unshifted field phase and the shifted field phase at the sample. Figure 23 (b) also shows a phase tilt due to an

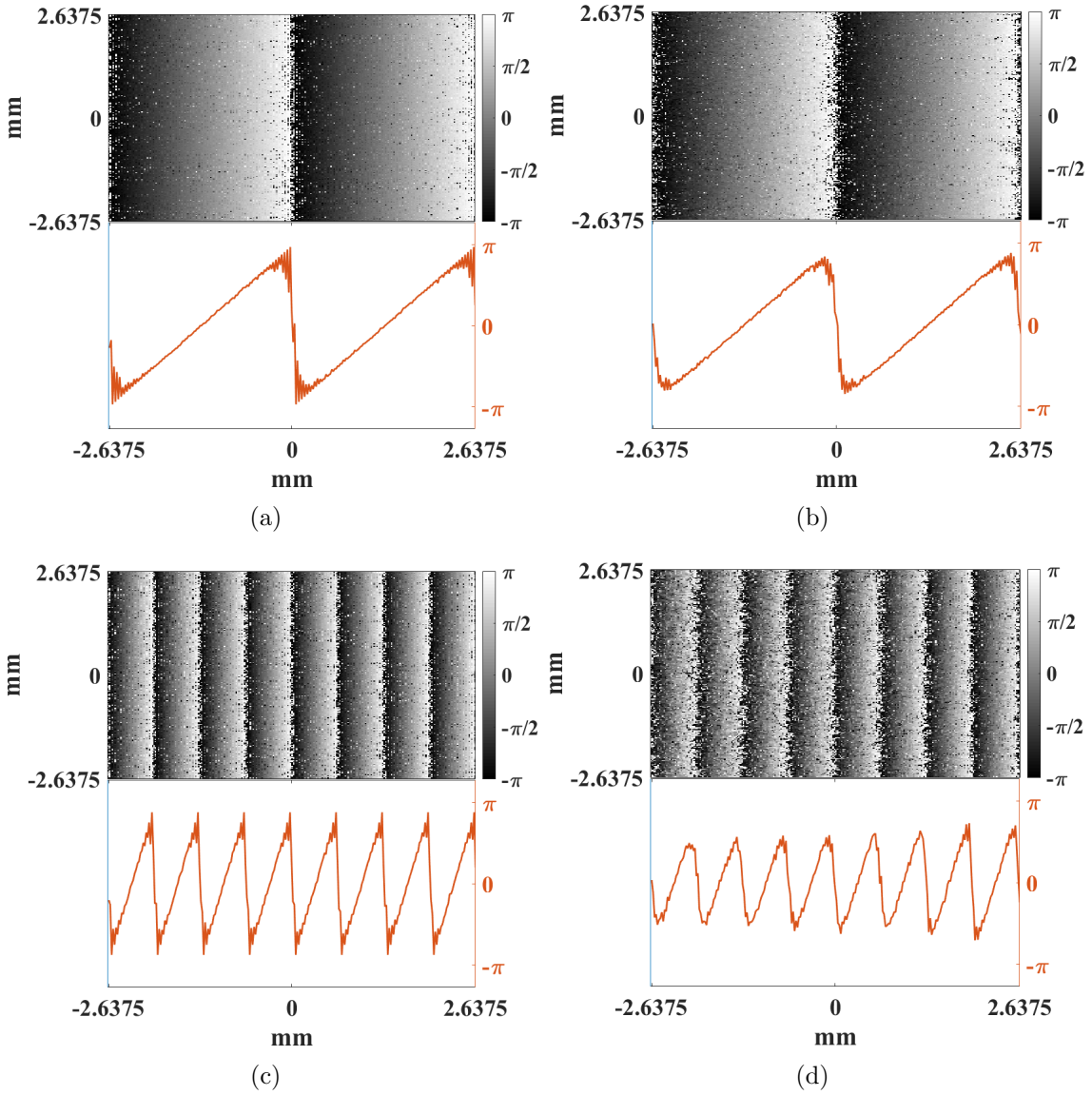



**Figure 23. Difference between phase angles of the electric field at the rough surface. In (a) and (b), the difference is between an unshifted phase mask and a phase mask horizontally shifted by eight pixels on the SLM for a sample (a) at the lens' focal plane and (b) $5mm$ past the lens' focal plane. In (c) and (d), the difference is between an unshifted phase mask and a phase mask horizontally shifted by 32 pixels on the SLM for a sample (c) at the lens' focal plane and (d) $5mm$ past the lens' focal plane. Beneath each phase map, in orange, is a vertically-average phase profile. A comparison of (a) and (b) to (c) and (d) shows that increased shift size decreases the range of the phase angle fluctuations from the ideal $-\pi$ to $\pi$. A comparison of (a) and (c) to (b) and (d) shows that rounding of the phase profile corners occurs when the sample is not at the focus.**

8-pixel horizontal shift of the SLM phase mask, when $\Delta Z = 5mm$. Figures 23 (c) and (d) both show the extracted phase tilts for a 32-pixel horizontal shift of the SLM phase mask for $\Delta Z = 0$, and $\Delta Z = 5mm$ respectively. As expected, the larger shift of the SLM phase mask creates steeper phase tilts at the sample when compared to (a) and (b). Figure 23 shows two trends which are degrading the ideal form of the linear phase tilt. The first trend is that increasing the shift size appears to decrease the range of the phase angle cycle. In Figures 23 (a) and (b), which were created by 8-pixel shifts, the range of the phase values almost covers the possible range of $-\pi$ to $\pi$, whereas Figures 23 (c) and (d) show visibly decreased ranges. The second trend is that moving the sample from the focus of the lens causes rounding of the corners of the phase profile, as seen in Figures 23 (b) and (d). However, the rounding of the corners also appears to reduce ringing effects at those corners. This is to be expected because ringing is created when a Fourier series attempts to recreate a discontinuity, or sharp jump in a signal, and is lacking necessary high-frequency components.

**Experimental Results.**

Figures 24 and 25 show the normalized experimental enhancements of the polished nickel and paint samples respectively, normalized, and as functions of displacement size. These results validate the simulations which show that a single phase mask could be reused to focus to multiple points, for a focal plane system.

The nickel sample experiences a rather uniform and gradual decrease of enhancement as displacement increases. The distance at which the enhancement decreases by 50% along the $u$ axis is approximately $1.39mm \pm 0.13mm$. Along the $v$ axis, the distance for 50% decrease is approximately $1.42mm \pm 0.12mm$. Then, the distance at which enhancement decreases by 75% along the $u$ axis is at approximately $2.5mm \pm 0.3mm$, and along the $v$ axis, it is approximately $2.7mm \pm 0.12mm$. The

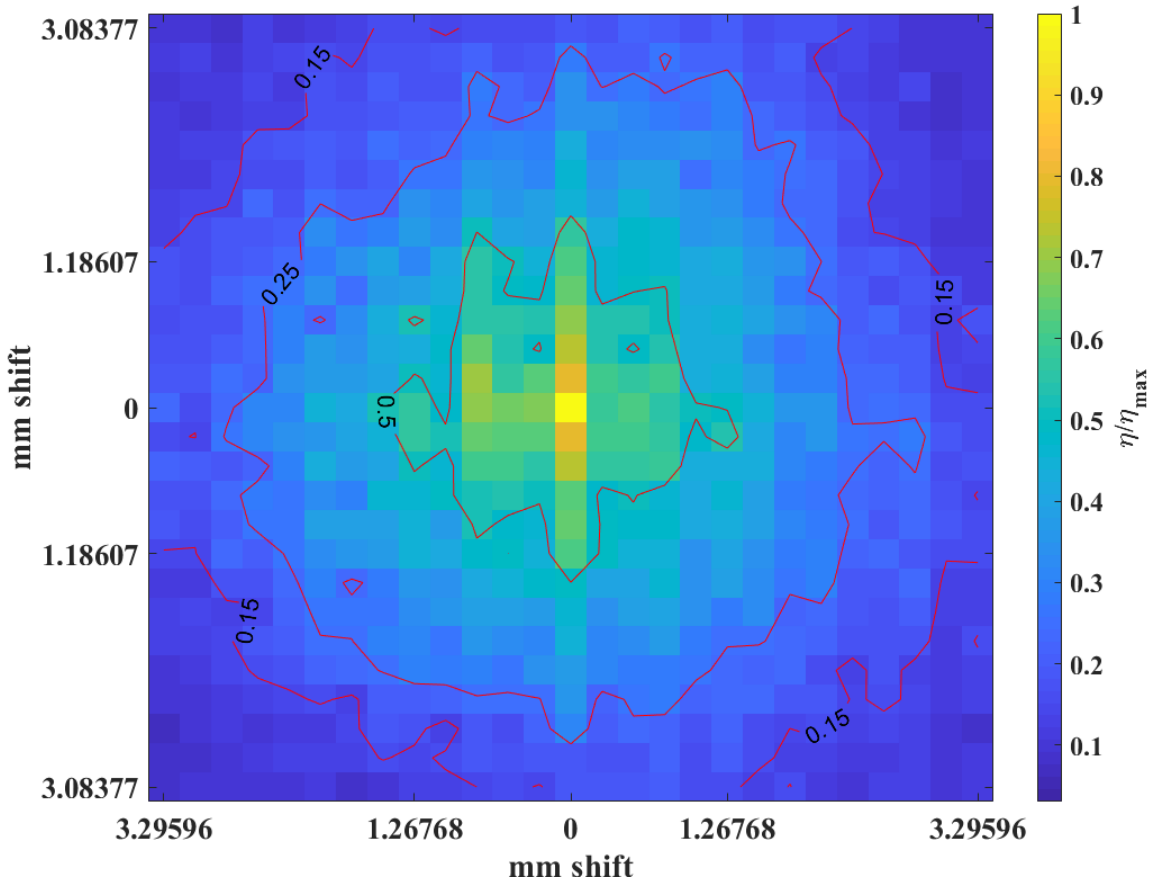



**Figure 24. Experimental enhancement of displaced spot as a two dimensional function of displacement size after a circular shift of the SLM phase map, of focal plane system shown in Figure 13 for 600 grit polished nickel. Results were averaged over five trials. The maximum value of $\eta$ prior to normalization was $89.9 \pm 1.5$.**

paint sample dropped rapidly to 50% enhancement at $.23mm \pm 0.11mm$ along the $u$ axis, and by $.24mm \pm 0.12mm$ along the $v$ axis. However, the subsequent decrease by 75% occurs at $1.6mm \pm 0.11$ along the $u$ axis, and at $1.07mm \pm 0.12mm$ along the $v$ axis. Aside from the rapid drop off to 50%, the paint sample is also asymmetrical between the axes. Overall, enhancement drops with displacement of the refocused spot faster in experiment than in in the simulations. This could be explained in part by a larger error in the placement of the sample with respect to the focal plane of the lens than was accounted for in the simulations. The reason that this is possible is because in simulation, the light is normally incident on the rough surface. In

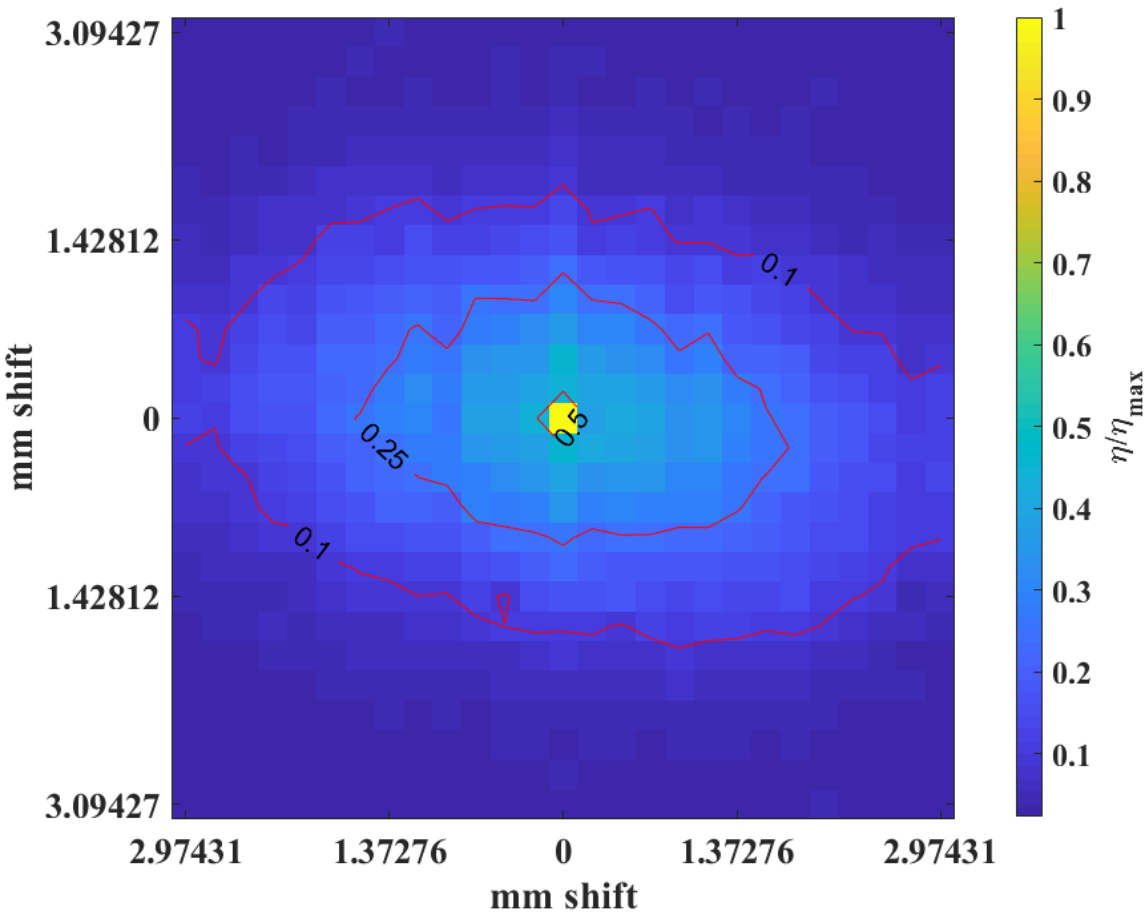



**Figure 25. Experimental enhancement of displaced spot as a two dimensional function of displacement size after a circular shift of the SLM phase map, of focal plane system shown in 13 zinc oxide paint, on glass. Results were averaged over five trials. The maximum value of $\eta$ prior to normalization was $80 \pm 3$.**

experiment, the light was obliquely incident on the scattering sample at a 45° angle from the surface normal, with a horizontal plane of incidence. Because of the oblique incidence, the distance that different parts of the wavefront have to travel in order to contact the surface are different. It is also likely that the oblique incidence of the light is related to the horizontally elongated shape of the contours on the white paint sample. Other than surface height and slope variations, the main difference between the nickel sample and the paint sample is that the nickle sample is a surface scatterer while the paint sample is more of a bulk scatterer.

Table 2 shows the experimental and expected displacements based on Equation

(32). The displacement size along the $v$ axis, which is the axis parallel to the SLM's y axis falls within the predicted range of Equation (32). The displacement along the $u$ axis, conversely does not fall within the range predicted by Equation (32). It is difficult to draw conclusions from this result, because for one sample, the value is over the estimate and for the other sample, the value is under the estimate; otherwise, the testing parameters were the same. Some uncertainty about the process of creating displacement may be attributable to fact that the $u$ axis is within the plane of incidence of the light. However, a displacement of the sample from the focal plane of the lens may degrade the form of the phase tilts, which was shown in Figure 23. By degrading the phase tilts, the relationship in Equation (33) may not be valid.

Table 3 shows various surface properties of the two scattering samples tested based on profilometer data. Looking at the samples independently, both correlation length metrics, $l_c$, and $l_{\lambda/2}$, are positively correlated. Roughness and slope are also positively correlated. Roughness and slope together are negatively correlated to both correlation length metrics. Because all four properties are somewhat redundant, only the roughness of the samples will be considered for analysis. Now, comparing Table 2 to Table 3, one possible trend is a link between smaller mean displacements and lower roughness, for a given sample. The nickel sample created smaller displacements along the $v$ axis, where it had a smaller roughness, and likewise, the paint sample created smaller displacements along the $u$ axis, where it had smaller roughness. This result seems counterintuitive, and additional testing would be prudent before any

**Table 2. Predicted and Tested Minimum Mean Displacement Sizes**

| Displacement Type | Equation (32) Predicted | Tested Displacement |
|---|---|---|
| Nickel $\Delta u$ | $15.0\mu m \pm 0.4\mu m$ | $15.85\mu m \pm 0.01\mu m$ |
| Nickel $\Delta v$ | $15.0\mu m \pm 0.4\mu m$ | $14.83\mu m \pm 0.01\mu m$ |
| Paint $\Delta u$ | $15.0\mu m \pm 0.4\mu m$ | $14.3\mu m \pm 0.01\mu m$ |
| Paint $\Delta v$ | $15.0\mu m \pm 0.4\mu m$ | $14.88\mu m \pm 0.01\mu m$ |

**Table 3. Surface properties of scattering samples, from a profilometer. Roughness is the standard deviation of surface height. $l_c$ is the distance required to shift the autocorrelation of the surface height profile by $1/e$. Slope is the RMS surface slope. $l_{\lambda/2}$ is the distance required to create a height change of $\lambda/2$ assuming the surface has a linear slope with the value of the RMS slope.**

| Sample, axis | Roughness | Correlation, $l_c$ | Slope | Correlation $l_{\lambda/2}$ |
|---|---|---|---|---|
| 600-grit Nickel, $u$ | $0.50\mu m$ | $12.25\mu m$ | 0.086 | $3.69\mu m$ |
| 600-grit Nickel, $v$ | $0.40\mu m$ | $19.75\mu m$ | 0.046 | $6.82\mu m$ |
| White Paint on Glass, $u$ | $0.67\mu m$ | $22.25\mu m$ | 0.080 | $3.96\mu m$ |
| White Paint on Glass, $v$ | $0.75\mu m$ | $17.75\mu m$ | 0.092 | $3.45\mu m$ |
| 600-grit Nickel, average | $0.45\mu m$ | $16\mu m$ | 0.066 | $4.79\mu m$ |
| White Paint on Glass, average | $0.71\mu m$ | $20\mu m$ | 0.086 | $3.69\mu m$ |

claims about the cause should be made. With respect to Figures 24 and 25, the nickel sample has a lower roughness, which could be linked to the slower falloff of the enhancement with displacement, however, the surface properties do not explain why the paint sample's enhancement drops off so quickly initially, or why the shape of the contours are not symmetric about the axes.

## 5.6 Conclusions

True non-mechanical beam steering while performing reflective inverse diffusion was demonstrated for a focal plane system. This validates Burgi's conclusions from the simulations which showed that redundant phase information is contained within the RM of PWO. The ability to beam steer means that there is no longer a need to completely restart iterative algorithms when attempting to focus to different channels in the observation plane. However, the effectiveness of using circular shifts for beam steering is dependent on a precise Fourier transform relationship that requires accurate placement of a lens with respect to a rough surface. Additionally, the circular shift itself is not the ideal solution for beam steering because it leaves vacated pixels unoptimized. The use of the circular shift was meant to show that there is

information contained in a single phase mask that allows for refocusing to more than one position. As a quick first approximation, the circular shift can be used for beam steering without requiring any additional calculations or measurements, but a more complete solution could incorporate simulations and measurements in real time. Using the concept of Dynamic Data Driven Application Systems (DDDAS) [22], the same GPUs which can run simulations of reflective inverse diffusion could be used to run smart algorithms which analyze measured data in real time to find optimal phase masks faster than simple feedback schemes currently allow. Ideally, a DDDAS integrated algorithm would be able to give predicts the modified phase masks required to focus to different points better than a circular shift. Finally, one result that was not fully explained was the behavior of the paint sample compared to the nickel sample. Because the surface height and slope, from Table 3 did not explain the discrepancy, the best explanation for this effect is that nickel is a surface scatterer, and paint is a bulk scatterer. It is possible that the non-uniform bulk scatter in the paint sample degrades the Fourier transform relationship between the field at the SLM and the surface. The shape of the contours for the paint sample are likely related to the oblique incidence of the light on the sample, but is not clear what mechanism cause the horizontal axis to support displacements better than the vertical axis.

# VI. Conclusion

## 6.1 Research Summary

The stated objective of this research was to demonstrate that in wavefront shaping, one phase mask is capable of being repurposed to refocus light in reflective inverse diffusion in more than one direction.

This research investigated several iterative algorithms for wavefront shaping, and has demonstrated the effectiveness of Conkey's genetic algorithm (GA) for use in reflective inverse diffusion. The GA is a robust algorithm that can perform under noisy conditions. Even under controlled laboratory conditions which limited noise, the GA outperformed the other existing iterative algorithms. Next, the ability to beam steer a single enhanced spot created by a single phase mask was demonstrated. This was done by applying circular shifts to the SLM phase mask.

The combined impact of these two rather tangential topics is that a new process whereby one (or perhaps several, if need be) continually improving phase maps can be found through iterative means, which can at any time steer their refocused beams to scan across an area. Alternatively, a single phase map could be used to construct an entire reflection matrix (RM).

## 6.2 Acknowledgements

This research was largely a follow-on study of the redundant phase information found in RMs by Burgi [19]. Additionally, this research builds on the foundation in reflective inverse diffusion research he previously laid, which includes results, but also technical expertise, practical proficiency, and even Matlab® code.

## 6.3 Future Work

This research has improved one of the methods used for achieving reflective inverse diffusion, which is an essential operation in the process of seeing around corners. The other main operation in the process of seeing around corners will be using the light in reflective inverse diffusion for dual photography.

However, before reflective inverse diffusion is applied to dual photography, wavefront shaping techniques must still be tested and improved. First, the proposed process of creating an RM using circular shifts in a focal plane system on a single phase map must still be tested, and as discussed in Chapters IV and V, there are areas of future work within the performance of the GA itself, and in improvements that could be made to the type of shift that is applied to the SLM in order to beam steer.

**Construction of an RM.**

The first continuation of this research could be the construction of reflection matrices. This would not be without difficulty, because the range over which circular shifts create an enhanced spot with an acceptable level of enhancement did not fill the entire surface of the CCD. To remedy this problem, it may be necessary to find a patchwork pattern of phase maps, each mapping to a separate location on the CCD such as in Figure 20. These patchwork phase maps could be optimized simultaneously in some kind of alternating fashion, essentially in parallel, or they could be optimized individually, in a sequence. Design and testing of actual methods to create RMs, which answer these questions is needed.

**Phase Mask Interpolation.**

Given the need to develop an RM, a technique which interpolates phase masks based on multiple known phase masks could be attempted. This method would work in the focal plane system, because phase masks to adjacent channels have been shown to contain redundant phase information. This is true for the phase map for each channel, and is therefore also true for non-adjacent phase maps. The concept of phase mask interpolation would use this principle so that if two phase masks to non-adjacent channels were known, the phase mask of a channel in between could be found by combining portions of the known masks. In terms of the circular shift, which was discussed in Chapter V, knowledge of multiple close by phase masks could provide the information about what values to assign to vacated pixels, when using a shift. In theory, phase mask interpolation would be superior to circular shifts, because it would provide the necessary phase information to use in the vacated pixels.

**Integration of Measurements and Simulations.**

Though Conkey's genetic algorithm is currently the best iterative algorithm tested for reflective inverse diffusion, improvements must still be made to increase the speed of the algorithm. Currently, the genetic algorithm, like other iterative algorithms, uses a very simple feedback system that requires no computational power. However, ample computational power is available and is not being used. As mentioned in Chapter V, the integration of measurements and simulations in real time could allow for algorithms that can predict optimal phase masks. Specifically, a combination of simulated propagation of the SLM to the scattering sample, and back-propagation of the CCD's intensity map to the scattering sample would give information about the electric field before and after reflection from the scattering sample, and a phase retrieval algorithm could be used to make guesses about the surface height properties

of the scattering sample in real time.

**A Lensless Setup.**

An important finding from Chapter V was that the efficiency of using circular shifts depended on a precise placement of the sample at the focal plane. Though the lens creates a Fourier transform relationship that allows for easy non-mechanical beam steering, it may also be a liability due to the need for precise lens-sample placement, and the limited range of frequency components that can be created with the Fourier transform of a finite signal (as evidenced by ringing effects in Figure 23). Figure 26 shows the results of preliminary testing of a lensless system. The genetic algorithm was used with the 600-grit polished nickel sample in the optical setup from Figure 13, only with the lens removed from the system. SLM segments two pixels, or $30\mu m$ wide were modulated, focusing to a CCD channel two pixels, or $14.8\mu m$ wide. Figure 26 demonstrates that it is possible to use a lensless system to create reflective inverse diffusion, and future work could explore the merits of a lensless system compared to the focal plane and imaging plane systems.

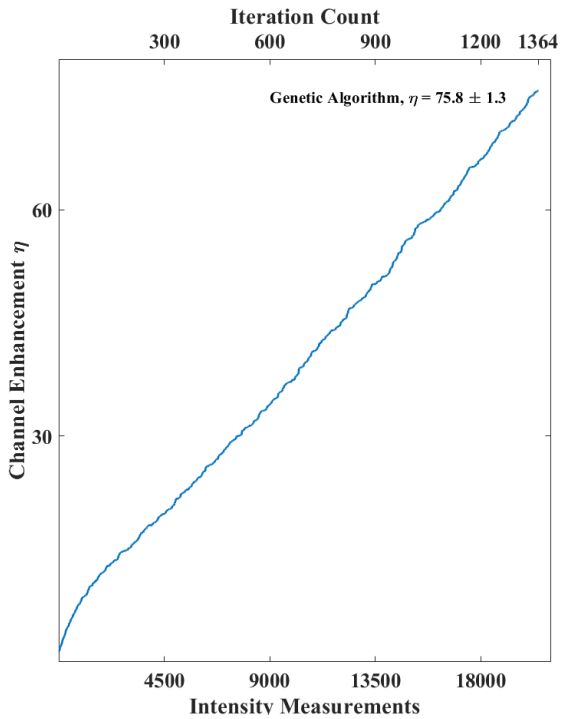



**Figure 26. Experimental genetic algorithm performance in a lensless system, using the 600 grit polished nickel sample. The optical setup was identical to the setup used in Figure 13, except that the lens was removed from the system.**

# Appendix A. Rayleigh-Sommerfeld Diffraction Propagator in Matlab®

The following code models the propagation of light from the SLM to the observation plane, as described in Section 4.3 for the system seen in Figure 11.

## SLM propagation function

```
function [Uobs,Uminus,Uplus,dx3] = PropagateSLM(input,z1,focus,z2,sample,a,recalc)
persistent prevZ1 prevF prevZ2 prevM prevN prevA pad
persistent x3 chirp dx1
persistent Hslm2lens Hsample2ccd
if nargin<7
recalc=0;
elseif nargin<6
a=2;
recalc=0;
elseif nargin<5
sample=1;
a=2;
recalc=0;
end
[m,n]=size(input);
M=a*m;
[msample,nsample]=size(sample);
if msample~=1 || nsample~=1
if a*m~=msample && a*n~=nsample
error('sample incorrect size')
end
end

if isempty(prevZ1) %|| isempty(prevZ2) || isempty(prevM) || isempty(prevN)
recalc=1;
prevZ1=0;
prevF=0;
prevZ2=0;
prevM=0;
prevN=0;
prevA=0;
elseif z1~=prevZ1 || focus~=prevF || z2~=prevZ2 || m~=prevM || n~=prevN || a~=prevA
recalc=1;
```

```
end
if recalc
display('recalculating');

lambda=633*10^-9;       % HeNe wavelength
k=2*pi/lambda;          % Propagation number
D=0.00768;              % BNS SLM dimension
L1=a*D;                 % Length of SLM plane (including zero pad)
% SLM plane coordinates
dx1=L1/M;                       % SLM pixel size
x1=-L1/2:dx1:L1/2-dx1;
[X1,Y1]=meshgrid(x1);           % SLM pixel coordinates

% propagation Transfer Function - SLM to Lens
fx2=(-M/2:M/2-1)*1/L1;          % range of sampled frequencies
[FX2,FY2]=meshgrid(fx2);        % Sampled frequency space coordinates
Hslm2lens=exp(1i*k*z1*sqrt(1-(lambda*FX2).^2-(lambda*FY2).^2)); %Rayleigh-Sommerfeld Transfer function

% sample plane
dx3=lambda*focus/L1;
L3=lambda*focus/dx1;   % Length of sample plane
x3=-L3/2:dx3:L3/2-dx3;
[X3,Y3]=meshgrid(x3);  %sample plane coordinate grid

% Fraunhofer chirp
chirp=exp(1i*k/(2*focus)*(X3.^2+Y3.^2));

% propagation Transfer Function - sample to CCD
fx4=(-M/2:M/2-1)*1/L3;
[FX4,FY4]=meshgrid(fx4);
Hsample2ccd=exp(1i*k*z2*sqrt(1-(lambda*FX4).^2-(lambda*FY4).^2));


prevZ1=z1;
prevF=focus;
prevZ2=z2;
prevM=m;
prevN=n;
prevA=a;
pad=gather((M-m)/2);
```

```
end
u1=padarray(input,[pad,pad]); % SLM field
Uminus=chirp.*fftshift(fft2(fftshift(u1)))*dx1^2.*Hslm2lens;
% Field just prior to the sample

Uplus=sample.*Uminus;
% Field after interaction with sample

Uobs=fftshift(ifft2(fft2(fftshift(Uplus)).*fftshift(Hsample2ccd)));
% Field at Observation plane
end
```

# Appendix B. Iterative Algorithm Simulations in Matlab®

The following code was used to simulate the iterative algorithms for reflective inverse diffusion, as discussed in Chapter IV. The intensity at the observation plane was calculated by the propagator in Appendix A.

## 2.1 Continuous Sequential Algorithm

### Propagation Parameters.

```
lambda=gpuArray(633*10^-9);   % HeNe wavelength
k=gpuArray(2*pi/lambda);      % Propagation number
slmpix=gpuArray(512);         % Number of physical SLM pixels
a=gpuArray(2);                % zeropad factor
IObsdim=gpuArray(2048);       % Dimension of camera pixels
D=gpuArray(0.00768);  % SLM side length;
dx1=gpuArray(D/slmpix);     %SLM delta x
x1=gpuArray(((-slmpix/2):(slmpix/2-1))*dx1);     % SLM coordinate array
flens=gpuArray(0.5);      % positive lens focal length    flens=0.2;
z1=gpuArray(0.15);         % distance from SLM to the lens   z1=0.3;
z2=gpuArray(0.5);         % distance from reflector to CCD   z2=0.4;
dx2=lambda*flens/(a*D);              % Sample pixel size
x2=((-a*slmpix/2):(a*slmpix/2-1))*dx2; % Sample coordinate array
```

### SLM Settings & CCD Settings.

```
SLMdimension=gpuArray(32);
SLMchannels=SLMdimension^2;

CCDdimension=gpuArray(128);
CCDchannels=CCDdimension^2;
```

### Noise Settings.

```
JitterSpace=gpuArray(1); % size of vibrations in elements shifted
SampleDimension=slmpix*a; % Dimension of scattering sample
CCDerror=gpuArray(0.05); % intensity measurement error in percent
SLMerror=gpuArray(pi/4); % SLM error in radian
```

### Determine Observation plane windowing.

```
%chooses the window over which to calculate the background intensity
s1=gpuArray(129);s2=gpuArray(1024-128);s3=s1;s4=s2;
```

## Iterative Optimization to Test Channel.

```
SLM1=ExpImage(2*pi*rand(slmpix,'gpuArray'),[slmpix slmpix]); % Random initial guess for SLM phase

%CCD Channels
TestChannel=(CCDdimension+1)*CCDdimension/2;% Center
%TestChannel=(CCDdimension+1)*CCDdimension/4*3;% bottom right
%TestChannel=(CCDdimension+1)*CCDdimension/4+CCDdimension/2;% bottom left
%TestChannel=(CCDdimension+1)*CCDdimension/4;% top left
%TestChannel=(CCDdimension+1)*CCDdimension/4*3-CCDdimension/2;% top right
[ChannelMask,BackgroundMask]=ChannelMasks(TestChannel,CCDdimension,(1024));

% Phase Levels
PhaseSpacing=gpuArray(15); % Number of phase levels
PhaseValue=2*pi/PhaseSpacing*((-PhaseSpacing/2):(PhaseSpacing/2-1)); % values of the phase levels
kmax=log(SLMchannels)/log(4); % The number of times the CSA will run through the SLM Counting preoptimization
Performance=zeros(1,1364,'gpuArray');
```

## Generate a random sample.

```
Jsample=exp(1i*2*pi*rand(SampleDimension+2*JitterSpace,'gpuArray'));
```

## Iterations Begin.

```
for k=1:kmax
N=2^(2*k);
for n=1:N
ChannelEnhance=zeros(length(PhaseValue),1,'gpuArray');
SLMmask=ChannelMasks(n,sqrt(N),slmpix);
for p=1:length(PhaseValue)
SLM1(SLMmask)=PhaseValue(p);
SLMnoise=SLMerror*rand(size(SLM1),'gpuArray')-SLMerror/2;

% simulation optics table vibration
% Uniform Distribution
Jx=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);
Jy=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);

% Apply the jitter to the sample
JstartX=JitterSpace+Jx+1;JstopX=JitterSpace+Jx+SampleDimension;
JstartY=JitterSpace+Jy+1;JstopY=JitterSpace+Jy+SampleDimension;
sample=Jsample(JstartX:JstopX,JstartY:JstopY);
```

```
% Calculate Intensity at observation plane, with added error
Uobs=10^6*PropagateSLM(gpuArray(exp(1i*SLM1+SLMnoise)),z1,flens,z2,sample,a);
CCDnoise=-CCDerror*rand(size(Uobs),'gpuArray')+1;
Iobs=CCDnoise.*abs(Uobs).^2;

% Use only the portion of the Observation plane where there is a
% speckle field
WindowIobs=Iobs(s1:s2,s1:s2);
WindowsBmask=BackgroundMask(s1:s2,s1:s2);

%Calculate the channel enhancement for the current phase value
ChannelEnhance(p)=sqrt(WeightedIntensity2(Iobs(ChannelMask),2048/CCDdimension))/
sqrt(mean(WindowIobs(WindowsBmask).^2));

clear Iobs CCDnoise JstartY JstartX Jy Jx SLMnoise WindowIobs WindowBmask
end
[~,idx]=max(ChannelEnhance); % Identify the best phase level
SLM1(SLMmask)=PhaseValue(idx);  % Set SLM segment to best phase level

Performance(n+sum(2.^(2*(1:k-1))))=max(ChannelEnhance); % Keep track of the algorithm performance
end
end
```

## 2.2 Partitioning Algorithm

### Propagation Parameters.

```
lambda=gpuArray(633*10^-9);   % HeNe wavelength
k=gpuArray(2*pi/lambda);      % Propagation number
slmpix=gpuArray(512);         % Number of physical SLM pixels
a=gpuArray(2);                % zeropad factor
IObsdim=gpuArray(2048);       % Dimension of camera pixels
D=gpuArray(0.00768);  % SLM side length;
dx1=gpuArray(D/slmpix);     %SLM delta x
x1=gpuArray(((-slmpix/2):(slmpix/2-1))*dx1);     % SLM coordinate array
flens=gpuArray(0.5);      % positive lens focal length    flens=0.2;
z1=gpuArray(0.15);         % distance from SLM to the lens   z1=0.3;
z2=gpuArray(0.5);         % distance from reflector to CCD   z2=0.4;
dx2=lambda*flens/(a*D);              % Sample pixel size
```

```
x2=((-a*slmpix/2):(a*slmpix/2-1))*dx2; % Sample coordinate array
```

### SLM Settings & CCD Settings.

```
SLMdimension=gpuArray(128);
SLMchannels=SLMdimension^2;
CCDdimension=gpuArray(512);
CCDchannels=CCDdimension^2;
```

### Noise Settings.

```
JitterSpace=gpuArray(1); % size of vibrations in elements shifted
SampleDimension=slmpix*a; % Dimension of scattering sample
CCDerror=gpuArray(0.05); % intensity measurement error in percent
SLMerror=gpuArray(pi/4); % SLM error in radian
```

### Determine Observation plane windowing.

```
%chooses the window over which to calculate the background intensity
s1=gpuArray(129);s2=gpuArray(1024-128);s3=s1;s4=s2;
```

### Iterative Optimization to TestChannel.

```
SLM1=ExpImage(2*pi*rand(slmpix,'gpuArray'),[slmpix slmpix]); % Random initial guess for SLM phase
PrevSLM=SLM1;
PrevEnhancement=gpuArray(0);

%CCD Channels
TestChannel=(CCDdimension+1)*CCDdimension/2;% Center
%TestChannel=(CCDdimension+1)*CCDdimension/4*3;% bottom right
%TestChannel=(CCDdimension+1)*CCDdimension/4+CCDdimension/2;% bottom left
%TestChannel=(CCDdimension+1)*CCDdimension/4;% top left
%TestChannel=(CCDdimension+1)*CCDdimension/4*3-CCDdimension/2;% top right
[ChannelMask,BackgroundMask]=ChannelMasks(TestChannel,CCDdimension,(1024)); %s2-s1+1

% Phase Levels
PhaseSpacing=gpuArray(15); % Number of phase levels
PhaseValue=2*pi/PhaseSpacing*((-PhaseSpacing/2):(PhaseSpacing/2-1)); % values of the phase levels
```

### Generate a random sample.

```
Jsample=exp(1i*2*pi*rand(SampleDimension+2*JitterSpace,'gpuArray'));
```

### Iterations begin.

```
K=1364; % Number of iterations
Performance=zeros(1,K,'gpuArray');
for n=1:K
ChannelEnhance=zeros(length(PhaseValue),1,'gpuArray');
SLMmask=rand(SLMdimension,'gpuArray')<rand(1)/2;

 % Creates index mask to access the elements of the SLM to be modulated
SLMmask=gpuArray(imresize(gather(SLMmask),gather(slmpix/SLMdimension),'nearest'));

for p=1:length(PhaseValue)
SLM1(SLMmask)=PhaseValue(p);
SLMnoise=SLMerror*rand(size(SLM1),'gpuArray')-SLMerror/2;

% simulation optics table vibration
% Uniform Distribution
Jx=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);
Jy=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);

% Apply the jitter to the sample
JstartX=JitterSpace+Jx+1;JstopX=JitterSpace+Jx+SampleDimension;
JstartY=JitterSpace+Jy+1;JstopY=JitterSpace+Jy+SampleDimension;
sample=Jsample(JstartX:JstopX,JstartY:JstopY);

% Calculate Intensity at observation plane, with added error
Uobs=10^6*PropagateSLM(gpuArray(exp(1i*SLM1+SLMnoise)),z1,flens,z2,sample,a);
CCDnoise=-CCDerror*rand(size(Uobs),'gpuArray')+1;
Iobs=CCDnoise.*abs(Uobs).^2;

% Use only the portion of the Observation plane where there is a
% speckle field
WindowIobs=Iobs(s1:s2,s1:s2);
WindowsBmask=BackgroundMask(s1:s2,s1:s2);
%Calculate the channel enhancement for the current phase value
ChannelEnhance(p)=sqrt(WeightedIntensity2(Iobs(ChannelMask),2048/CCDdimension))/
sqrt(mean(WindowIobs(WindowsBmask).^2));

clear Iobs CCDnoise JstartY JstartX Jy Jx SLMnoise WindowIobs WindowBmask
end
```

```
% Backtrack mechanism- does not implement changes, unless there is an
% improvement of enhancement
if max(ChannelEnhance)>PrevEnhancement
[~,idx]=max(ChannelEnhance);
SLM1(SLMmask)=PhaseValue(idx);
PrevSLM=SLM1;
PrevEnhancement=max(ChannelEnhance);
else
SLM1=PrevSLM;
end

Performance(n)=PrevEnhancement; % Keep track of performance
end
```

## 2.3 Genetic Algorithm

### Propagation Parameters.

```
lambda=gpuArray(633*10^-9);   % HeNe wavelength
k=gpuArray(2*pi/lambda);      % Propagation number
slmpix=gpuArray(512);         % Number of physical SLM pixels
a=gpuArray(2);                % zeropad factor
IObsdim=gpuArray(2048);       % Dimension of camera pixels
D=gpuArray(0.00768);  % SLM side length;
dx1=gpuArray(D/slmpix);     %SLM delta x
x1=gpuArray(((-slmpix/2):(slmpix/2-1))*dx1);     % SLM coordinate array
flens=gpuArray(0.5);      % positive lens focal length    flens=0.2;
z1=gpuArray(0.15);         % distance from SLM to the lens   z1=0.3;
z2=gpuArray(0.5);         % distance from reflector to CCD   z2=0.4;
dx2=lambda*flens/(a*D);              % Sample pixel size
x2=((-a*slmpix/2):(a*slmpix/2-1))*dx2; % Sample coordinate array
```

### SLM Settings & CCD Settings.

```
SLMdimension=gpuArray(128);
SLMchannels=SLMdimension^2;
CCDdimension=gpuArray(512);
CCDchannels=CCDdimension^2;
```

## Noise Settings.

```
JitterSpace=gpuArray(1); % size of vibrations in elements shifted
SampleDimension=slmpix*a; % Dimension of scattering sample
CCDerror=gpuArray(0.05); % intensity measurement error in percent
SLMerror=gpuArray(pi/4); % SLM error in radian
```

## Determine Observation plane windowing.

```
%chooses the window over which to calculate the background intensity
s1=gpuArray(129);s2=gpuArray(1024-128);s3=s1;s4=s2;
```

## Optimization to test channel.

```
TestChannel=(CCDdimension+1)*CCDdimension/2;% Center
%TestChannel=(CCDdimension+1)*CCDdimension/4*3;% bottom right
%TestChannel=(CCDdimension+1)*CCDdimension/4+CCDdimension/2;% bottom left
%TestChannel=(CCDdimension+1)*CCDdimension/4;% top left
%TestChannel=(CCDdimension+1)*CCDdimension/4*3-CCDdimension/2;% top right
[ChannelMask,BackgroundMask]=ChannelMasks(TestChannel,CCDdimension,(1024)); %s2-s1+1
```

## GA Parameters:.

```
P=gpuArray(30); % Population size
G=gpuArray(15); % Number of offspring each generation
R0=gpuArray(.1); % Initial mutation rate - fraction of "mutated" segments in offspring
Rend=gpuArray(.01); % Final Mutation rate
K=gpuArray(1364);
Performance=zeros(1,gather(K));
df=K/3; % Decay Factor
```

## Generate initial population.

Comment out if initial population already exists

```
Population_SLM=2*pi*rand(SLMdimension,SLMdimension,P,'gpuArray');
Population_SLM=imresize(gather(Population_SLM),gather(slmpix/SLMdimension),'nearest'); %Units are radians
Jsample=exp(1i*2*pi*rand(SampleDimension+2*JitterSpace,'gpuArray'));  % uniform distribution

\subsection*{Measure Initial population}
Fitness=zeros(1,P,'gpuArray');
for i=1:P
```

```
% SLM phase error
SLMnoise=SLMerror*rand(size(Population_SLM(:,:,i)),'gpuArray')-SLMerror/2;

% simulation optics table vibration
% Uniform distributed Jitter
Jx=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);
Jy=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);

% Apply the jitter to the sample
JstartX=JitterSpace+Jx+1;JstopX=JitterSpace+Jx+SampleDimension;
JstartY=JitterSpace+Jy+1;JstopY=JitterSpace+Jy+SampleDimension;
sample=Jsample(JstartX:JstopX,JstartY:JstopY);

% Calculate Intensity at observation plane, with added error
Uobs=10^6*PropagateSLM(gpuArray(exp(1i*Population_SLM(:,:,i)+SLMnoise)),z1,flens,z2,sample,a);
CCDnoise=-CCDerror*rand(size(Uobs),'gpuArray')+1;
Iobs=CCDnoise.*abs(Uobs).^2;

% Use only the portion of the Observation plane where there is a
% speckle field
WindowIobs=Iobs(s1:s2,s1:s2);
WindowsBmask=BackgroundMask(s1:s2,s1:s2);

%Calculate fitness of current population member
Fitness(i)=sqrt(WeightedIntensity2(Iobs(ChannelMask),slmpix*a/CCDdimension))
/sqrt(mean(WindowIobs(WindowsBmask).^2));

clear Iobs CCDnoise sample JstartY JstartX Jy Jx SLMnoise WindowIobs WindowBmask
end

% Sort the SLM screens by fitness
[Fitness,fitorder]=sort(Fitness,'descend');
Population_SLM(:,:,:)=Population_SLM(:,:,fitorder);
```

## Iterations Begin.

```
for n=1:K % n is the generation number
Offspring_SLM=zeros(slmpix,slmpix,G,'gpuArray');
for i=1:G
% Chose parents - Higher rank has higher weighted chance
```

```
ParentIndices=randsample(P,2,true,(P:-1:1).^X); % Linear Weights

% Generate binary template
Template=rand(SLMdimension,'gpuArray');
Template=imresize(gather(Template),gather(slmpix/SLMdimension),'nearest');

Template=Template>.5;
Offspring_SLM(:,:,i)=Population_SLM(:,:,ParentIndices(1)).*Template
+Population_SLM(:,:,ParentIndices(2)).*(1-Template);

% Generate mutation template
Template=rand(SLMdimension,'gpuArray');

Template=imresize(gather(Template),gather(slmpix/SLMdimension),'nearest');
Template=Template<(R0-Rend)*exp(-n/df)+Rend;
Mutation=2*pi*rand(SLMdimension,'gpuArray');
Mutation=imresize(gather(Mutation),gather(slmpix/SLMdimension),'nearest');
% in radians

%Mutate offspring
Offspring_SLM(:,:,i)=Offspring_SLM(:,:,i).*(1-Template)+Mutation.*(Template);
end

% measure cost
Offspring_Fitness=zeros(1,G,'gpuArray');
for i=1:G
% SLM phase error
SLMnoise=SLMerror*rand(size(Offspring_SLM(:,:,i)),'gpuArray')-SLMerror/2;

% simulation optics table vibration
% Uniform Distribution
Jx=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);
Jy=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);

% Apply the jitter to the sample
JstartX=JitterSpace+Jx+1;JstopX=JitterSpace+Jx+SampleDimension;
JstartY=JitterSpace+Jy+1;JstopY=JitterSpace+Jy+SampleDimension;
sample=Jsample(JstartX:JstopX,JstartY:JstopY);

% Calculate Intensity at observation plane, with added error
```

```
Uobs=10^6*PropagateSLM(gpuArray(exp(1i*Offspring_SLM(:,:,i)+SLMnoise)),z1,flens,z2,sample,a);
CCDnoise=-CCDerror*rand(size(Uobs),'gpuArray')+1;
Iobs=CCDnoise.*abs(Uobs).^2;

% Use only the portion of the Observation plane where there is a
% speckle field
WindowIobs=Iobs(s1:s2,s1:s2);
WindowsBmask=BackgroundMask(s1:s2,s1:s2);

%Calculate current offspring fitness
Offspring_Fitness(i)=sqrt(WeightedIntensity2(Iobs(ChannelMask),2048/
CCDdimension))/sqrt(mean(WindowIobs(WindowsBmask).^2));

clear Iobs CCDnoise JstartY JstartX Jy Jx SLMnoise WindowIobs WindowBmask
end

% Replace lowest ranked masks with new generation of G offspring
Population_SLM(:,:,P-G+1:end)=gather(Offspring_SLM);
Fitness(P-G+1:end)=Offspring_Fitness;
clear Offspring_SLM

% Sort new population of phase screens SLM by fitness
[Fitness,fitorder]=sort(Fitness,'descend');
Population_SLM(:,:,:)=Population_SLM(:,:,fitorder);
Performance(n)=gather(max(Fitness)); % Keep track of the algorithm performance

clear sample
end
```

# Appendix C. Beam Steering Simulations in Matlab®

The following code was used to simulate the beam steering process, as discussed in Chapter V. The intensity at the observation plane was calculated by the propagator in Appendix A.

## Initialization.

```
% Initialize matrices to store information
EnhancementMatrix=zeros(33);
Xmatrix=EnhancementMatrix;
Ymatrix=EnhancementMatrix;
PhiMatrix=EnhancementMatrix;
dxMatrix=EnhancementMatrix;
dyMatrix=EnhancementMatrix;
increment=16; % SLM pixel shift size
m=0;

for i = -256:increment:256  % Range of shifts
for j= -256:increment:256
```

## Measure Intensity at observation plane.

```
% SLM phase error
SLMnoise=SLMerror*rand(size(Population_SLM(:,:,1)),'gpuArray')-SLMerror/2;

% simulation optics table vibration
%Uniform Distribution
Jx=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1);
Jy=randi(2*JitterSpace+1,'gpuArray')-(JitterSpace+1)

% Apply the jitter to the sample
JstartX=JitterSpace+Jx+1;JstopX=JitterSpace+Jx+SampleDimension;
JstartY=JitterSpace+Jy+1;JstopY=JitterSpace+Jy+SampleDimension;
sample=Jsample(JstartX:JstopX,JstartY:JstopY);

% Calculate Intensity at observation plane, with added error
Uobs=10^6*PropagateSLM(indexshift(gpuArray(exp(1i*Population_SLM(:,:,1)+SLMnoise)),i,j),z1,flens,z2,sample,a);
CCDnoise=-CCDerror*rand(size(Uobs),'gpuArray')+1;
Iobs=CCDnoise.*abs(Uobs).^2;
```

## Determine where the shifted spot has gone.

```
% Location not known a priori, requiring estimates
ShiftedChannelMask=indexshift(ChannelMask,round(-i*.725),round(-j*.725));
% Rough estimation for the shift location- Hard coded in, will differ for various setups

% Integration area around the estimated shift location
BallparkMask=ExpandChannel(ShiftedChannelMask,3,CCDdimension,length(Iobs));
IntegrationArea=Iobs.*BallparkMask;  % Zero out all other elements outside integration area for measurement
[~,A]=max(IntegrationArea(:));
maxx=round(A/(length(Iobs)))+1; % x coordinate of the maximum within the ballpark area of the peak
maxy=rem(A,(length(Iobs))); % y coordinate of the maximum within the ballpark area of the peak

% Create new integration mask centered on the peak pixel within the
% integration area
[MaxShiftMask,~]=ChannelMasks2(maxy,maxx,CCDdimension,length(Iobs)); % Mask centered on maximum value
MaxShiftMask=ExpandChannel(MaxShiftMask,1.5,CCDdimension,length(Iobs));

% Calculate the centroid, and other beam parameters of the spot...
% Ref ISO11146-1
CPUIobs=gather(Iobs);

% calculate x,y coordinates of centroid, as well as spot size/rotation
[x,y,~,~,~,phi,dx,dy]=ISOintegrals(CPUIobs.*MaxShiftMask,1,length(Iobs),1,length(Iobs));

% Round the centroid coordinates to nearest pixels
x=ceil(x);
y=ceil(y);

% Create a channel mask for the shifted channel, centered at the
% calculated centroid
[FinalMask,FinalBackgroundMask]=ChannelMasks2(y,x,CCDdimension,length(Iobs));

% Final mask on the newly calculated x and y coordinates of the spot
```

## Calculate Enhancement of Shifted spot.

```
% Use only the portion of the Observation plane where there is a
% speckle field
WindowIobs=CPUIobs(s1:s2,s1:s2);
```

```
WindowsBmask=FinalBackgroundMask(s1:s2,s1:s2);

% Store the calculated values
index1=i/increment+ceil(length(EnhancementMatrix)/2);
index2=j/increment+ceil(length(EnhancementMatrix)/2);
EnhancementMatrix(index1,index2)=(sqrt(mean(CPUIobs(FinalMask)).^2))/sqrt(mean(WindowIobs(WindowsBmask).^2));
Xmatrix(index1,index2)=x;
Ymatrix(index1,index2)=y;
PhiMatrix(index1,index2)=phi;
dxMatrix(index1,index2)=dx;
dyMatrix(index1,index2)=dy;
m=m+1;
end
end
```

# Bibliography


1. K. W. Burgi, "Reflective Matrix Method for Controlling Light after Reflection from a Diffuse Scattering Surface", Dissertation, Air Force Institute of Technology (2016).

2. P. Sen, B. Chen, G. Garg, S. R. Marschner, M. Horowitz, M. Levoy, and H. P. Lensch, "Dual Photography," *Proc. SIGGRAPH* , 745-755 (2005).

3. J. Goodman, *Introduction to Fourier Optics, 4ed.*, W.H Freeman and Company, New York, NY (2017).

4. C. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, Society for Industrial and Applied Mathematics, (1992).

5. D.P. Resler, D.S. Hobbs, R.C. Sharp, L.J. Friedman, and T.A. Dorschner "High-efficiency liquid-crystal optical phased-array beam steering" *Optics Letters* **21**(9), 689-691, (1996).

6. I. M. Vellekoop, "Feedback-based wavefront shaping," *Opt. Express* **23**, 12189-12206 (2015).

7. A. P. Mosk, A. Lagendijk, G. Lerosey, and M. Fink, "Controlling waves in space and time for imaging in complex media," *Nature Photonics* **6**, 283-292, (2012).

8. I. Freund, "Looking through walls and around corners," *Physica A: Statistical Mechanics and its Applications* **168**, 49-65 (1990).

9. I. M. Vellekoop, and A.P. Mosk, "Focusing coherent light through opaque strongly scattering media," *Optics Letters* **32**(16), 2390-2311, (2007).

10. I. M. Vellekoop, A. Legendijk, and A. P. Mosk, "Exploiting disorder for perfect focusing," *Nature Photonics* **4**, 320-322 (2010).

11. I. M. Vellekoop, and A.P. Mosk, "Phase control algorithms for focusing light through turbid media" *Optics Communications* **281**(11), 3071-3080, (2008).

12. M. Cui "Parallel wavefront optimization method for focusing light through random scattering media," *Optics Letters* **36**, 870-872 (2011).

13. J. Yoon, K. Lee, J. Park, and Y. Park "Measuring optical transmission matrices by wavefront shaping," *Opt. Express* **23**, 10158-10167 (2015).

14. K. Burgi, M. Marciniak, M. Oxley, and S. Nauyoks, "Measuring the Reflection Matrix of a Rough Surface," *Appl. Sci.* **7**, 568 (2016).

15. D. B. Conkey, A. N. Brown, A. M. Caravaca-Aguire, and R. Piestun, "Genetic algorithm optimization for focusing through turbid media in noisy environments," *Opt. Express* **20**, 4840-4849 (2012).

16. *User Manual XY Phase Series Spatial Light Modulator with 16-bit PCIe Controller* Rev. 4.10, Meadowlark Optics, Frederick, CO.

17. K. Burgi, J. Ullom, M. Marciniak, and M. Oxley, "Reflective Inverse Diffusion," *Appl. Sci.* **6**, 370 (2016).

18. O. Katz, E. Small, and Y. Silberberg, "Looking around corners and through thin turbid layers in real time with scattered light," *Nature Photonics* **6**, 549-553, (2012).

19. K. W. Burgi, M.A. Marciniak, S.E. Nauyoks, and M.E. Oxley, "Exploiting redundant phase information of a reflection matrix," *Proc. Spie* **10347**, (2017).

20. D.G. Voelz, *Computational Fourier Optics: A MATLAB Tutorial*, (SPIE Press, Bellingham, Washington, 2011).

21. J.T. Verdeyen, *Laser Electronics, 3ed.*, Prentice Hall, Upper Saddle River, NJ (2000).

22. F. Darema "Dynamic Data Driven Applications Systems: New Capabilities for Application Simulations and Measurement," *Computational Science – ICCS 2005*, 610-615 (2005).