Design and Simulation of Three-Dimensional Hologram Emitting Phased Arrays in the Near Field

by

Jerry Zhou

B.S. Electrical and Computer Engineering,
University of Illinois (2013)

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2015

Signature of Author...
Department of Electrical Engineering and Computer Science
August 28, 2015

Certified by...
Michael R. Watts
Associate Professor
Thesis Supervisor

Accepted by...
Leslie A. Kolodziejski
Chair of the Committee on Graduate Students

Design and Simulation of Three-Dimensional Hologram Emitting Phased Arrays in the Near Field

by

Jerry Zhou

Submitted to the Department of Electrical Engineering and Computer Science
on August 28, 2015, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Electrical Engineering

Abstract

Silicon photonics technology is gaining attention in both research and industry because of its potential to revolutionize optical components and systems while utilizing the well-established silicon semiconductor fabrication processes. Similarly, three-dimensional holography technology is emerging to build the foundation for three-dimensional displays, interfaces, and many other applications. This work attempts to combine these two technologies to generate a method to design a phased array that will emit three-dimensional holograms and investigate the scalability of the phased arrays.

This thesis will cover the design and functionality of optical phased arrays and how they have been previously used to emit patterns in the far field. It will then explore three-dimensional hologram theory to prompt its incorporation into a phased array design. In order to accomplish three-dimensional holograms, this design technique takes advantage of the fact that the near field allows for multiple planes of focus, as opposed to the single focal pattern of the far field. A three-dimensional hologram can then be generated by breaking it up into planes and having the phased array recreate them by emitting the patterns that come into focus at different distances from the array. The method to do this involves back propagating desired output planes using Fresnel Diffraction and superimposing the back propagated electric fields to generate the required phased array parameters to emit such a field.

This work will then explore simulation results required to design the system and various techniques to improve the process and output quality. The proposed design increases the size of the array by four times our previous designs, requiring three-dimensional Finite-Difference Time-Domain (FDTD) simulations to provide a wider range of coupling. These additional coupler simulations also prove to be critical as this proposed design alters both the amplitude and phase of each antenna in the array, which can be controlled by these couplers. We demonstrate the flexibility of the design by designing and simulating multiple output patterns as well as at varying wavelengths.

Thesis Supervisor: Michael R. Watts
Title: Associate Professor

Acknowledgments

None of this work could be possible without the support and guidance of the great people around me. These are people that I could always count on for advice in research related topics, as well as for things outside that realm.

First, and most importantly, I’d like to thank my advisor Professor Michael R. Watts for everything that he has done for me. From the beginning, he believed in me and brought me into his group to work on exciting and ground breaking technologies. He put me in a position to make an impact in this growing field. Just the opportunity to work on these projects has meant so much to me and opened my eyes to the tremendous work being done in this field. His organized and passionate approach has kept me motivated and moving forward in the work that I do. He gave me research and career advice that I will carry with me wherever I go.

JP Laine and Draper Laboratory has supported me in a multitude of ways. Not only have they funded my research and academic progress, but JP has provided me with a lot of useful insight. He helped me understand the application space for our field in order to give context and meaning to our work. He made sure I always felt welcome at Draper and gave me useful advice on how to approach my career.

I’d like to thank Jie Sun for believing in me to carry on part of his work in phased arrays that this thesis is based on. He was there to guide me along the way and answer my many questions. His success really inspired me to try to make something that could live up to the great work that he accomplished.

I would also like to thank Ami Yaacobi for all of the help that he has provided. In this thesis alone, he was critical to some of the techniques that we implemented and in general, he had many great ideas on how to approach various problems. Along similar lines, I’d like to thank Erman Timurdogan and Zhan Su for their insightfulness into how to approach my work. They always had new ideas on things to try and ways to improve that were essential to this work. Their consistent passion and hard work also motivated me every day to try to match their enthusiasm.

I’d like to thank Chris Poulton for his help in this project. Not only did he provide technical direction, but he was instrumental in coming up with effective ways to strengthen the work overall. He helped me figure out what types of simulations I should show and what kinds of calculations I should make to bolster the presentation of the work. My officemates Purnawirman and Manan Raval have kept me going in both a research sense and outside of it. Our constant conversations about everything have kept me sane and moving forward in my life. In general, the entire Photonics Microsystems Group has been extremely supportive and helpful. They have been very welcoming and each is a wealth of knowledge. The people in this group really helped make my time at MIT enjoyable.

Finally, I can’t stress how appreciative I am of my parents Stanley Zhou and Lan Hu. Their story and what they have accomplished through hardship has always inspired me to be the best that I can be. They have been supportive throughout my journey and have always found ways to push me in the right direction. I can’t imagine being where or who I am today without them.

Contents

List of Figures

Chapter 1

Introduction

1.1. Silicon Photonics

The field of silicon photonics is a combination of two technologies that have seen a lot of success through their technological advancements. It builds upon the huge success of silicon and the semiconductor industry that has proven to have the ability to create smaller and smaller chips and feature sizes while also having a cost efficient method to mass produce these devices. Photonics, on the other hand, has shown the ability to harness the power of light and apply it to various laser, communication, and imaging applications. The combination of these two fields takes advantage of these technologies and allows for the fabrication of various photonic devices within a small chip sized form factor while utilizing the already well researched silicon fabrication technologies. One key application that silicon photonics is designed to address is within the field of optical communication. There is an increasing demand for faster and faster data communication within computer technology and utilizing light and photonics would be a viable solution for this need [1, 2].

As this is still a relatively new field, there have been great strides recently to demonstrate various silicon photonic components. These are the building blocks for the structures that can then be used to address the various applications for silicon photonics. By building up a set of verified components, a variety of structures can be created by using this library of interchangeable pieces. We will go through a few key components shown in Fig. 1.1-1 to give context to the field and this work.

Due to its ability to properly route different channels of light within a fiber, the optical add drop multiplexer (OADM) is a key component to optical networks. One of the key elements within this structure is the add-drop filter, which allows a signal to have different wavelength components added and dropped. A microring resonator filter designed using metal and silicon heaters to deal with temperature fluctuations has been proven to operate with a low insertion loss (0.05 dB) and a wide free-spectral range (35 nm) [3], two key parameters to ensure reliable operation. Insertion loss is how much of the signal power is lost when inserting, in this case, the filter into a transmission line, and obviously this must be minimized to prevent degradation of the signal as it is transmitted. This loss is predominately determined by the quality factor of the resonator, which is a measure of total energy stored over the energy lost in a cycle. Thus to achieve a lower loss, a larger quality factor is required and is realized in that work by minimizing loss to silicon heaters by doping the heaters in patches rather than a full coverage. The free-spectral range (FSR), on the other hand, is the measure of the distance between resonance peaks and a large FSR is desirable because it broadens the optical frequency range in which the device operates. Here, a larger FSR is primarily achievable by having a smaller ring resonator radius, and that work shows a small enough device that minimizes losses and is capable of operation over a 35 nm (4 THz) FSR.

When using high index contrast materials for light confinement as is done in silicon photonics, there is a need to control the polarization of the light since it affects propagation rate and coupling strength. This is due to the fact that boundary conditions for TE and TM modes become more critical to device operation as index contrast increases. Here, the TE mode equates to having no electric field in the direction of propagation and is useful because it is isolated entirely into magnetic fields in the direction of propagation (TM is the exact opposite). Polarization control is important since the light signal is often randomly polarized, but control can be accomplished by using a system of polarization rotators and splitters. Polarization splitters are important because they have the ability to split a signal into separate TE and TM modes. Once split, polarization rotators can rotate a wave’s polarization from TM to TE, and vice versa, providing for the ability to achieve a single polarization state within a photonic circuit. A successful demonstration of a polarization rotator designed using an approximated twist with two waveguide cores has been demonstrated with no significant wavelength sensitivity over the 1.45-1.75 μm band [4]. By introducing a twist in a waveguide, the axis and, consequently, the polarization states of the modes are rotated. Similarly, a polarization splitter designed utilizing two waveguides has been demonstrated to successfully split a signal into separate TE and TM outputs [5]. This is accomplished by utilizing separated vertical and horizontal waveguides designed so that boundary conditions of Gauss's law cause maximum modal confinement for each polarization state to one of the two waveguides. These two works were then combined to demonstrate the first add-drop filter from polarization-sensitive microring resonators [6].

In order to take advantage of the improved data rates of optical communication, a method to convert electrical signals to optical signals is required. The device that accomplishes this conversion is called an optical modulator, and allows electronic control of amplitude or phase of a signal. One way to design these modulators is the Mach-Zehnder modulator, which utilizes a Mach-Zehnder interferometer to control phase through changing refractive index with an electric field and amplitude control through changing applied voltage on a coupler. A design for a Mach-Zehnder modulator has demonstrated a $V_\pi L$ of 1 V·cm and wide open eye diagrams [7]. This $V_\pi L$ value tells us that the device only requires 5V applied to achieve a π phase shift in a 2 mm structure. An alternate design is the microdisk modulator, which has shown low power consumption (3 fJ/bit) and high speed (12.5 Gb/s) in a small form factor (3.5 μm diameter) [8]. These benefits are accomplished because this design maximizes overlap of the depletion region and optical mode, which minimizes the power required to shift phase. By minimizing contact resistance and therefore loss, as well as further optimizing optical mode overlap, an even lower power modulator (0.9fJ/bit) with high speed (25 Gb/s) has shown promise for future expansion of communications [9].

These examples of components can eventually be combined to create much larger systems to tackle the many aspects of optical communication. One such system is the optical broadcast network, which outlines how optical signals are distributed among various ports. Commonly, a system that provides links that allow all ports to communicate with each other is implemented, however this isn't ideal for larger sized system as the power requirements scale with N^2, where N represents the number of ports. An alternative solution is a system that groups certain "users" together and provides the same all-to-all information links within each group. A topology using a wavelength-selective optical drop filter network with tunable resonant filters [10] has proven to be a successful implementation of this type of network as it has shown low power variation (0.11dB), low loss (1.1 dB), and error-free operation for 10Gbit/s data rates [11]. Here, the microring resonators act as filters by tapping off transmission power for a specific wavelength while not picking up power from the other wavelengths. The filters have the flexibility to be thermally tuned to ensure an equal power distribution among all of the ports and are also designed with the small radius design to provide a large FSR and wide optical bandwidth.

These small form factor components have also created the possibility to design systems that were previously large and bulky and instead create them on-chip. For example, the first on-chip interferometer developed in silicon has been successfully demonstrated and brings with it a smaller size and associated cost compared to its table-top counterpart [12]. Interferometers operate by measuring an interference pattern generated by two beams to calculate phase difference, which can then be used to calculate properties such as path length change or change in refractive index. That work realized that it was possible to recreate this measurement system, commonly set up using expensive table-top optics and equipment, on a smaller and cheaper scale using silicon chips. Size and cost were reduced by replacing many of the bulky optics with silicon photonic components on chips, such as utilizing silicon waveguides to maintain spatial separation, and utilizing the CMOS process technology for large-scale manufacturing. This is a big step as interferometers have been essential in measuring displacement, LIDAR, and semiconductor processes and the ability to bring it on-chip allows for more portable and versatile measurement applications.

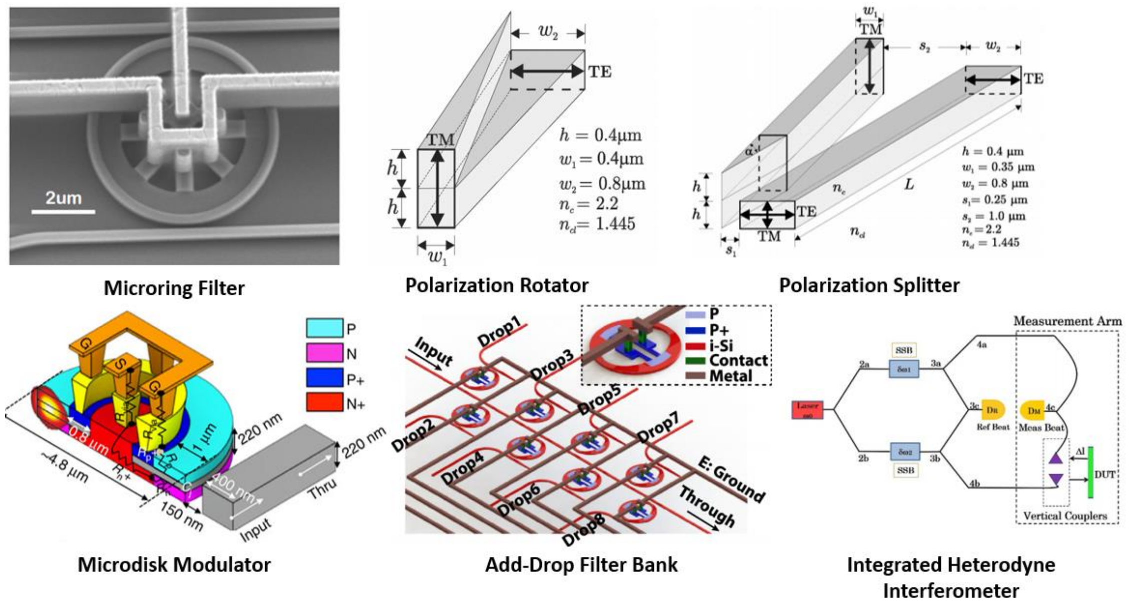

Fig. 1.1-1 Examples of some silicon photonic components and devices

1.2. Optical Phased Arrays

The optical phased array is yet another example showcasing the potential of combining silicon technology and photonics to generate densely integrated systems on small and cheap to produce chips. It consists of an array of antennas with individual phase control in order to emit a controlled radiation pattern. This emission control is useful as it allows for the generation of varying patterns as well as steerable beams. These devices have received a great deal of research attention due to their potential in fields such as communications, imaging, and detection.

Fig. 1.2-1 shows a schematic of the operation of a 1-D phased array. At the very bottom we see the electromagnetic power source, which is traditionally a laser for optical phased arrays. This power is then distributed in a way to achieve the desired amplitudes for each antenna. Each antenna is spaced an equal distance d apart and is connected to a phase shifter so that it emits a wave

$$\mathrm{A}_i e^{j\phi_i} \tag{1.2.1}$$

Where A_i is the previously mentioned amplitude of the antenna and ϕ_i is the phase relative to the 0th antenna. The figure shows that the emissions of the antennas will form a beam with an angle θ to the normal of the plane of the antennas that is dependent on the phase delays of the antennas. By adding a delay, we see that when the emitted waves are in phase, they form the aforementioned beam plane. Basically, we can imagine adding a phase delay to being similar to adding a delay in when the equiphase front is emitted so a delay would allow the previous antenna's wave to be further ahead as seen in Fig. 1.2-1. No phase delay would allow the equiphase front of each emitted field to be identical in shape and size and thus generate a flat beam plane that is parallel to the plane of the antennas. If we add an equal amount of phase delay $\Delta\phi$ to each subsequent antenna, we have the relation

$$\Delta\phi = \frac{2\pi}{\lambda} d \sin\theta_0 \tag{1.2.2}$$

to achieve a beam with angle θ_0 to the normal of the antenna plane. Tuning amplitude allows for control of the beam shape and also suppression of side lobes. Traditionally, the amplitude is tapered similarly to aperture antennas in order to reduce side lobes. The constructive and destructive interference caused by the emission of multiple antennas also provides the capability to generate bright and dark spots. For a 1-D array this will be limited in terms of where the bright spots will be located, but if we expand the array to 2-D, then there is much more freedom in where the bright spots are located, as we will see later.

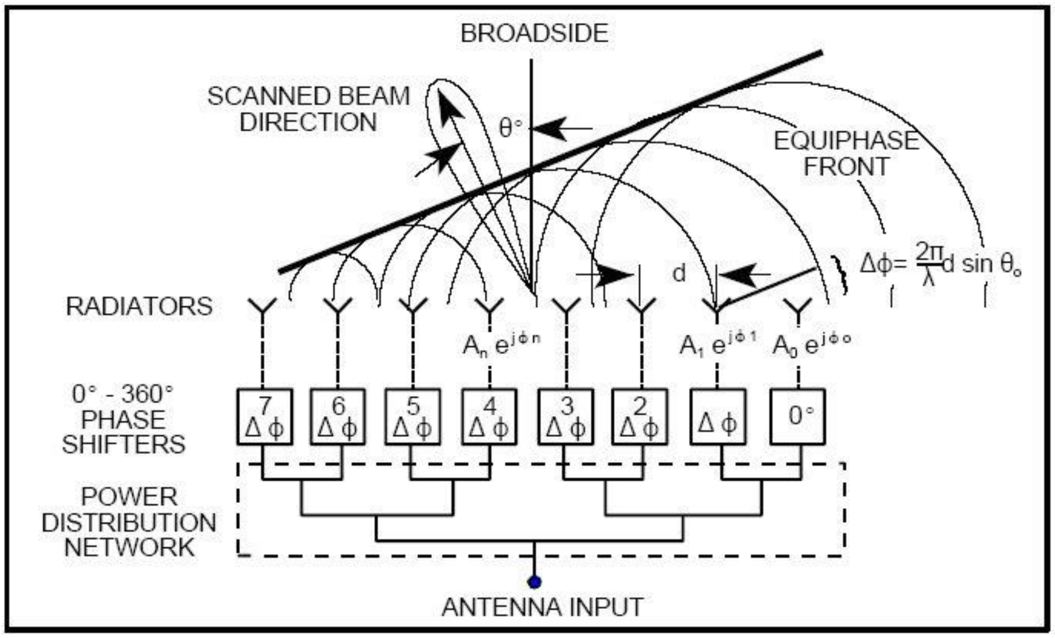

Fig. 1.2-1 1-D phased array schematic

Phased arrays on their own are not a relatively new concept as radio wave phased arrays have been well studied and used in broadcasting, radar, and various communication aspects [13]. The first published demonstration of multiple antennas transmitting in a common direction goes as far back as 1909 and was shown by Nobel laureate Karl Ferdinand Braun in a Nobel lecture [14]. He demonstrated the ability to generate an emitted field with a common direction by adding phase delays to antennas. As this idea evolved, it became known as a phased array due to the control of the emitted field through the relative phases of the antennas.

By utilizing the flexibility of emitted beams, phased arrays have been found to be useful in broadcast systems. They have commonly been employed at communication broadcast stations to boost transmission distances by taking advantage of having high gain directional antennas along with multidirectional emission through antenna phase modification [15]. Furthermore, antennas with scanning capabilities can be packed more tightly than inherently omnidirectional antennas with the same interference, which is crucial towards supporting a larger density of mobile users. It has also been shown that phased arrays can be used to overcome the attenuation due to rain on broadcast systems in the 21 GHz bands by utilizing various combinations of beam concentrations [16]. Here, it is shown that the flexibility of beam shapes allows for emission of concentrated beams to cover areas of heavy rain and a broader beam for less weather intensive regions to compensate for rain-attenuation without much additional satellite power.

Phased arrays have most commonly been used for radar applications due to its ability to quickly and accurately steer beams through the aforementioned phase control [17]. In the military realm, phased arrays have been employed in order to provide a single system to simultaneously track and detect objects in both air and surface [18] as well as targeting systems [19] and sonar [20]. Beyond military applications, this beam scanning that phased arrays allow also provides solutions to improving weather and aircraft traffic monitoring since it has the ability to provide updated scans at intervals of at most one minute [21]. This is critical to providing immediate alerts to incoming storms or managing air traffic safety.

Radar phased arrays, however, usually suffer from the fact that the systems require complex mechanical systems to steer beams, which generally leads to large, expensive overall systems. This severely limits the application space as these systems require high-cost and highly precise machinery in order to operate. In order to remove these limiting factors, optical phased arrays have been proposed in order to achieve high performance devices that come at a much smaller cost and complexity due to the elimination of required mechanical devices and smaller optical wavelength [22]. Here, the distinction is that optical phased arrays operate at optical wavelengths, including infrared, visible, and near ultraviolet, while the previously discussed phased arrays operate at much longer wavelengths. The shorter wavelength requirement allows optical phased arrays to potentially have significantly more elements in devices compared to the previously discussed radiofrequency phased arrays. These optical phased arrays are optimal for steering monochromatic laser beams and provide the same beam steering ability and signal amplification shown in radar phased arrays [23]. It proves useful as it opens up applications in laser communication and imaging and can be produced at a much larger scale due to the significant cost reduction and more compact size.

There have been many demonstrations of optical phased arrays utilizing liquid-crystal [24], 1-D arrays for beam steering [25], and 2-D arrays for beam steering [26]. These have been limited, however, to smaller scale works, which doesn't take advantage of the small form factor to produce larger integrated devices. This changed with the demonstration of a large-scale optical phased array that emits arbitrary far field patterns [27]. The work showcases the ability to design a 64x64 antenna phased array that is able to emit an arbitrary pattern by designing it to output the "MIT" logo in the far field. The Gerchberg-Saxton algorithm shown in Fig. 1.2-2 is an iterative algorithm used to generate the required antenna phases while keeping all 4096 antennas equally balanced in power. $|AF^k(\theta,\phi)|$ represents the amplitude of the desired far field output and $\Phi^k(\theta,\phi)$ is the test phase at the kth iteration. These are combined to give us $AF^k(\theta,\phi)$, which can be inverse Fourier transformed to give us $w^k{}_{mn}$, which represents the antenna response that generates $AF^k(\theta,\phi)$ in the far field. In order to simulate an even distribution of power, the amplitudes of $w^k{}_{mn}$ are set to 1 and the new field is propagated into the far field using another Fourier transform to generate $AF^{*k}(\theta,\phi)$. The phase $\Phi^{*k}(\theta,\phi)$ is then set to be the new test phase at the next iteration. Sufficient iterations will cause the final far field pattern $AF^{*k}(\theta,\phi)$ to match the desired output $|AF^k(\theta,\phi)|$ and the phases are used to tune the phased array antennas.

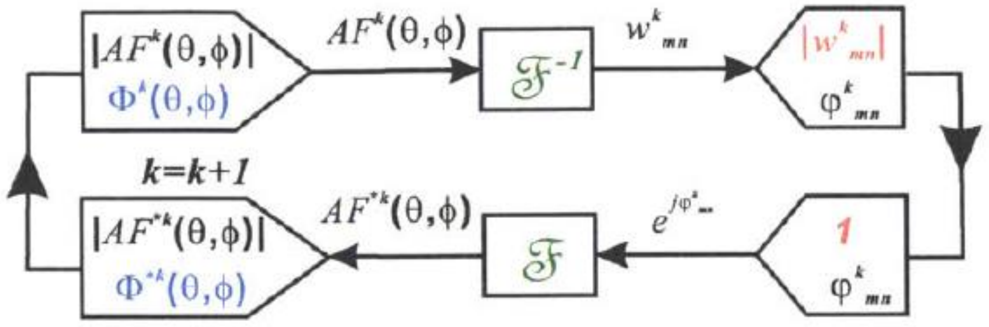

Fig. 1.2-2 Gerchberg-Saxton algorithm flow

Fig. 1.2-3 shows the 64x64 phased array and the inset shows an individual antenna and coupler. A laser is coupled into a waveguide that runs along the left side of the array. Row waveguides then couple power from this waveguide, and that power is then coupled to the individual antennas within the rows. The amount of power coupled both into the rows and the antennas is designed to provide equal power to all antennas to utilize the calculated phases from the Gerchberg-Saxton algorithm. Evanescent couplers control how much power is distributed and a more detailed explanation of how this and phase control works will be explained in a later chapter.

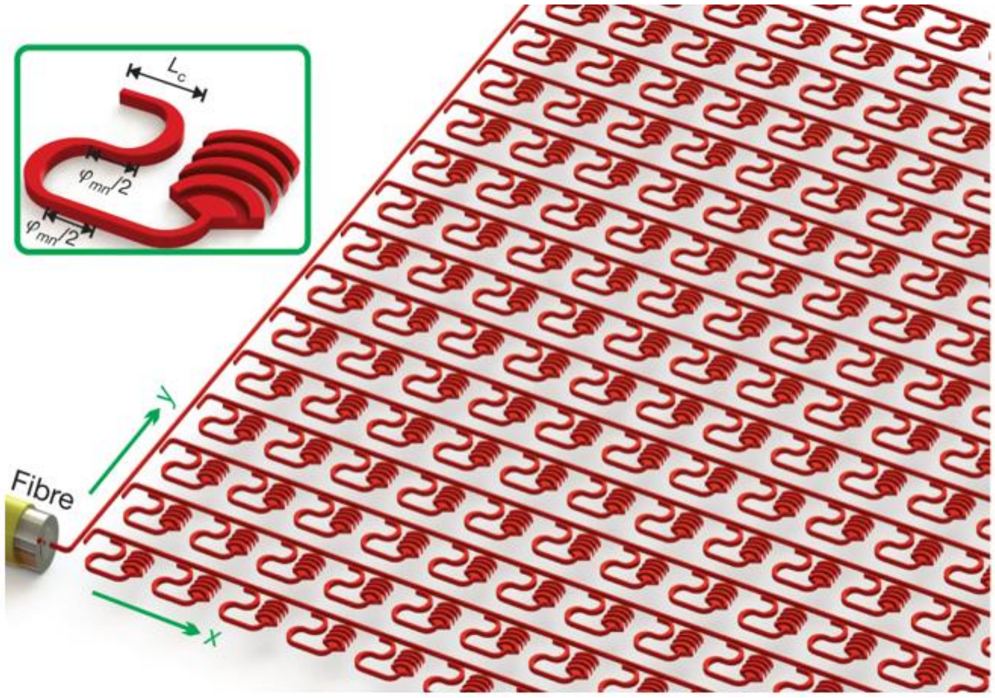

Fig. 1.2-3 Schematic of 64x64 phased array

Antennas are important to this design in order to achieve highly accurate output patterns. First, the size of the antennas should be as small as possible, otherwise higher order effects will have a greater effect on the output [28]. The antenna grating measures 3.0 µm by 2.8 µm with a 0.22 µm with five gratings and is simulated to achieve 51% upwards emission compared to only 30% downwards emission.

By using the Gerchberg-Saxton algorithm, the required antenna phases are calculated to generate the "MIT" logo in the far field, and a Fourier transform is used to simulate the far field output as seen in Fig. 1.2-4a. It is evident that the pattern emitted by antennas tuned to the calculated phases closely resembles the desired output as seen in the bottom right corner of Fig. 1.2-4a, showing that this method can truly generate arbitrary patterns.

This phased array design with the calculated phase profile and required couplers to evenly distribute power was then fabricated in a CMOS foundry and then tested using a 1550 nm laser as the input. Using an infrared camera, the emitted output is captured as seen in Fig. 1.2-4b, which again closely matches both the simulated output and the desired "MIT" logo. Furthermore, it was also shown that having less antenna elements (32x32) reduced the resolution of the output, further supporting the need for larger scale devices.

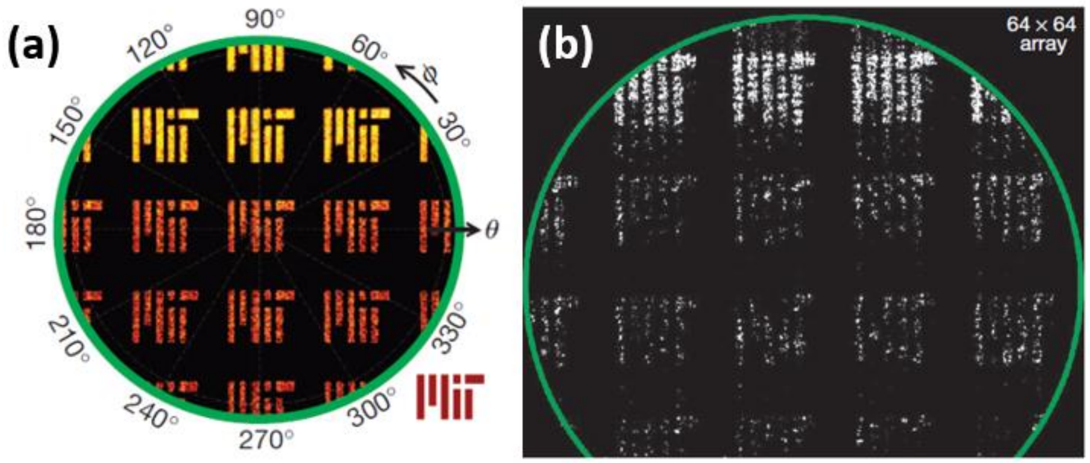

Fig. 1.2-4 (a) Simulated "MIT" logo far field pattern. (**b)** Experimental results of fabricated device

The paper continues by showing the ability for a larger array to effectively steer beams by designing a phased array comprised of antennas with tunable phase. Not only does this freedom of varying phase allow for beam steering with high resolution on small form factor devices, it also opens up the possibility for having dynamic output patterns. To design a tunable phased array, a resistive heater is implemented into the unit cell as seen in Fig. 1.2-5 so that electrical signals can thermo-optically tune the phase of the antennas.

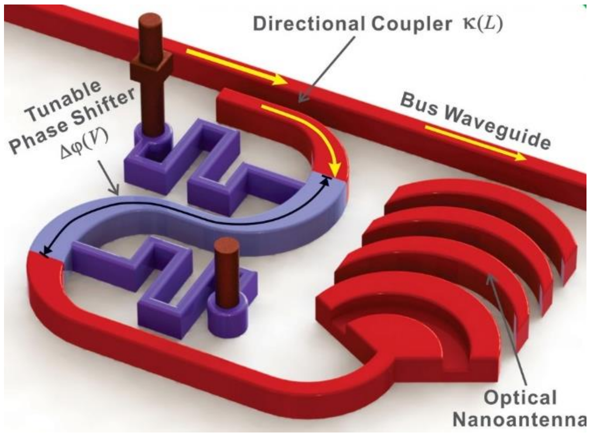
Directional Coupler $\kappa(L)$
Tunable Phase Shifter $\Delta\varphi(V)$
Bus Waveguide
Optical Nanoantenna

Fig. 1.2-5 Tunable phased array antenna design

The device was reduced to an 8x8 antenna phased array to test the phase tuning capabilities and once again fabricated in a CMOS foundry. The phased array was designed to emit a single dot in the far field with no electrical excitation as seen in the left-most pair of outputs in Fig. 1.2-6. Then, as the rest of Fig. 1.2-6 shows, voltages are applied in simulations and experimentally to show the output shifting vertically and horizontally, as well as splitting both vertically and horizontally. By showing that it is possible to design phased arrays that have changeable outputs, this work significantly opens up the potential applications for optical phased arrays as it is now proven that it has beam steering and dynamic pattern generation with high resolution. Again, it is evident that the experimental results closely resemble the simulated results, which further proves that this is a robust method to design phased arrays to emit arbitrary far field outputs.

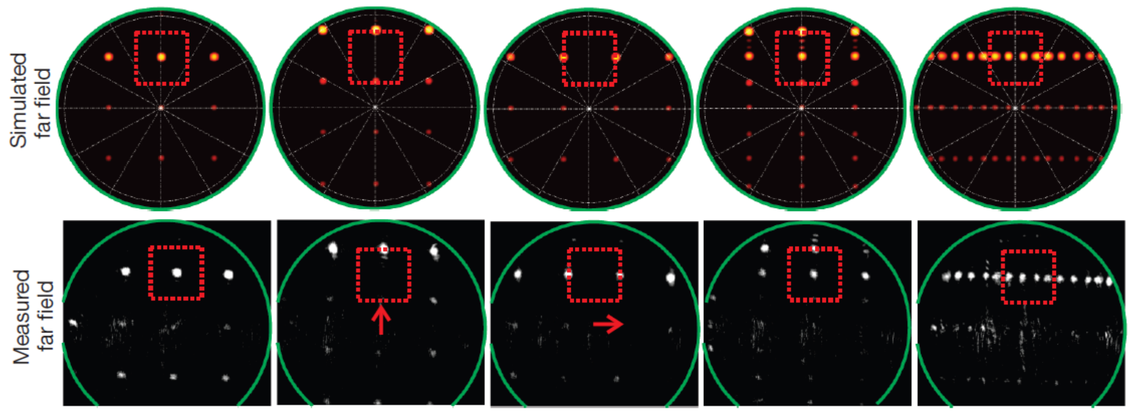

Fig. 1.2-6 Simulated and experimental results of tunable phased array

1.3. 3D Holography

Holography is closely tied to phased arrays since phase and amplitude information is critical to the generation of holograms and the preservation of phase information is the defining characteristic that distinguishes it from photography [29]. Generating a hologram is a two-step process that involves first recording the desired output and reconstructing an image of the original object. To record a hologram, the desired object is illuminated with a coherent light and the diffracted waves are interfered with a phase-related reference wave. This is then shone onto a recording medium, imprinting the phase and amplitude of the diffracted waves as seen in Fig. 1.3-1. The combination of light reflected and scattered by the object combines with the reference beam to form an interference fringe field, which is recorded by the recording media. Then, to reconstruct a 3D image of the object, the recorded hologram just needs to be illuminated by a wave identical to the reference wave. The reconstruction occurs when the reference wave is diffracted by the grating in the recording medium that is formed during the recording of the hologram.

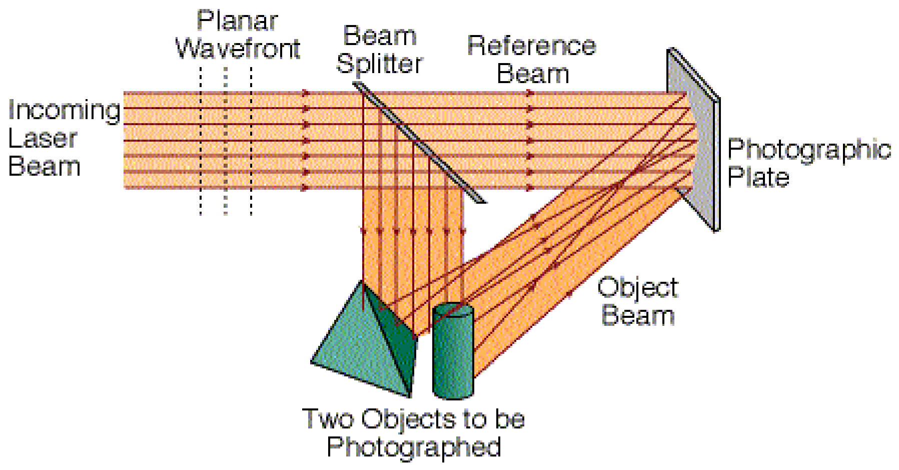

Fig. 1.3-1 Hologram recording

The process can be described mathematically by first beginning with a spherical wave solution to the Helmzholtz equation

$$\vec{E} = Ae^{jkr}e^{j2\pi f t}\hat{e} \tag{1.3.1}$$

where A is the amplitude of the wave, f is the frequency, and $\hat{e}$ is the polarization unit vector. By defining the complex field amplitude to be

$$\psi = Ae^{j\phi} \tag{1.3.2}$$

where $\phi = kr$, the irradiance of the field can be written

$$I = \vec{E} \cdot \vec{E}^* = \psi\psi^*\hat{e} \cdot \hat{e} = \psi\psi^* = |\psi|^2 \tag{1.3.3}$$

The field of the recording ψ_H is then just the interference between the object ψ_O and the reference wave ψ_R and can be written as a superposition of the two waves

$$\psi_H = \psi_O + \psi_R \tag{1.3.4}$$

Using (1.3.3), the irradiance of this field is then

$$I_H = (\psi_O + \psi_R)(\psi_O + \psi_R)^* = |\psi_O|^2 + |\psi_R|^2 + \psi_O{\psi_R}^* + \psi_R{\psi_O}^* \tag{1.3.5}$$

The transmittance T of the recorded hologram field can then be written as directly proportional to the irradiance of the field

$$T = K[|\psi_O|^2 + |\psi_R|^2 + \psi_O{\psi_R}^* + \psi_R{\psi_O}^*] \tag{1.3.6}$$

The equation that describes the recreation of the hologram, or equivalently illumination by the reference wave, can be expressed as a multiplication of the recorded hologram transmittance by the reference wave field

$$\psi_T = K[\psi_R(|\psi_O|^2 + |\psi_R|^2) + \psi_O|\psi_R|^2 + |\psi_R|^2{\psi_O}^*] \tag{1.3.7}$$

From this equation, we see that the first term is the transmitted wave with an attenuation factor. The second term is the original object wave with an amplitude factor and thus is the virtual holographic image of the object. Finally, we can see that the third term is the conjugate object wave also multiplied by an amplitude factor.

Holography has been around since Dennis Gabor's work in its invention in the 1940s, which eventually lead to being awarded the Nobel Prize in 1971 for the discovery [30, 31, 32]. It started getting wide-spread attention with the invention of the laser, which brought about the discovery of optical holograms and the first high-quality 3D images by Leah and Upatnieks [33]. Since then, holography has been successfully utilized for many applications.

One of the major application spaces for holography is in interferometry. Interferometry traditionally uses two optical wavefronts and measures optical path differences by measuring the interference of the wavefronts. A common example is the Mach-Zehnder interferometers, which can be used to map out device surfaces, measure variation in refractive indices, and describe the flow of gasses. However, usually two separate but identical optical beams are required for these applications, which is difficult to accomplish without complex and expensive systems. Holography addresses this issue because it can record desired wavefronts and consistently recreate them for use without the expensive equipment. Holographic interferometry is done by examining fringes due to phase difference in wavefronts stored in holograms to achieve quantitative and qualitative information [34]. This technology has proven effective in applications such as vibration measurement [35] and in providing nanometer resolution characterization of integral features of current and future electronic devices [36]. It has also been critical in detecting defects in manufacturing that helps to maintain a high device yield [37] and 3D digital holography has shown the ability to detect critical details that radiographs and CT scans are not able to pick up on in the biomedical field [38].

Holography has also been found to be useful in lithography techniques. Lithography is the transferring of a mask onto a resist covered wafer in order to outline structures during integrated circuit fabrication. Traditionally this has been accomplished using techniques such as contact printing, proximity printing, and step-and-repeat, however, each has its own issues. Contact printing can cause contamination and damage to the wafer, proximity printing has a limited resolution, and step-and-repeat methods are usually very complex and expensive. A holographic system provides a non-intrusive, full-field, and cost effective solution to lithography while maintaining the required high resolution [39]. Basically, the holographic solution can use real-image projection to overlay the mask image onto the wafer and eliminate the need for lenses. This eliminates the complexities of a lens system and also utilizes the high resolution achievable by holography.

Holography can also be used to make optical elements with the power to direct a light wave, similar to the previously discussed phased arrays. These elements can be made by recording a fringe pattern generated by interfering two light beams. Holography can be used to make gratings for spectrographic instruments and this is advantageous because they can be made without random and periodic groove variation and they have low light scatter [40]. It has also proven to be useful in optical beam scanning as it has the ability to combine both beam deflection and focusing into a single element, which traditional scanning methods cannot. Basically, the beam can be flexible and can easily change and focus based on the surface being scanned. A common example of where these are used can be found in supermarket scanners

[41]. Obviously this flexibility is important as items bought at the supermarket can easily vary in shape and size and the scanners must be able to account for this.

Finally, holography provides a promising tool in the field of 3D displays as it inherently generates 3D images. Advances in 3D display technology allow the generation of images without specialized eyewear, which opens up many possibilities in medical, industrial, and military imaging. Research in the field has shown the capability to generate updatable 3D displays which can record and display images within minutes and provide hours of viewing without refreshing the display system [42]. In a more commercial setting, this technology also opens up the possibility for 3D TVs [43]. Systems have already been designed showcasing the capabilities of real-time acquisition, transmission, and 3D display of dynamic scenes that don't require specialized glasses to view [44].

Chapter 2

Concept of Technique

The discovery of a method to generate arbitrary far field patterns using phased arrays has opened up many windows of opportunities within the field of photonics and communication [27]. That work highlighted the versatility and robustness of optical phased arrays due to the ability to project complex patterns in a small chip-sized form factor while utilizing the CMOS fabrication processes. This work attempts to expand on that previous discovery by proposing a method to design phased arrays with the capability to project 3D images. By projecting an image designed for viewing in the far field, the previous work was limited to only a single pattern with varying viewing distances. However, in order to generate arbitrary 3D projections, there is a need for the patterns to change with distance to give the image both space and depth. The proposed method will utilize near field diffraction in order to achieve output planes that are able to change with viewing distance as illustrated in Fig. 2-1. This property of near field diffraction provides for the ability to generate slices of a 3D object in space, creating the basis for generating 3D holograms. Our algorithm will calculate the required phased array parameters analogous to how the Gershberg-Saxton algorithm was used in previous phased array work to calculate the antenna phases for designing phased arrays that emit patterns in the far field.

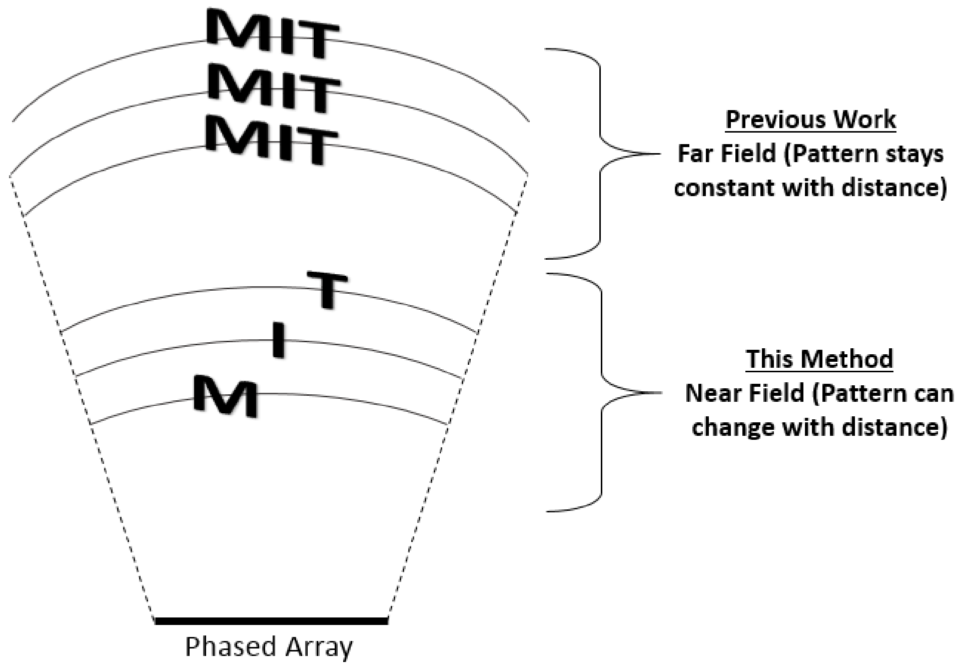
MIT
MIT
MIT
T
I
M
Previous Work
Far Field (Pattern stays constant with distance)
This Method
Near Field (Pattern can change with distance)
Phased Array

Fig. 2-1 Comparison of Far Field and Near Field patterns

Our 3D hologram design technique proposes that we can calculate the needed individual phased array antenna phase and amplitude for an arbitrary 3D pattern by back-propagation. The idea behind this is that diffraction propagation is reversible in the sense that an electric field that is propagated forward a certain distance will be recreated by propagating it backwards the same distance, and vice versa. The method to design these phased arrays entails slicing the desired 3D image into flat planes at varying distances from the desired phased array location. These planes then must be converted to electric field distributions that mimic the slices of the desired 3D image. Each desired electric field plane is then at a different vertical distance from the phased array and is back propagated individually to the common plane at the phased array's theoretical location. Then, the results of the back propagations are superimposed into a single electric field similar to how Brown and Lohmann have achieved the generation of digital binary holograms through superimposing Fourier transforms [45]. An example depicting this back-propagation of output planes and summation at the phased array location is shown in Fig. 2-2 using the MIT logo as an example output. The resultant electric field is discretized into a 128x128 point matrix, with each point corresponding to an aperture of emission from each antenna, and the phase and amplitude is extracted from each point to synthesize the needed parameters for the phased array antennas.

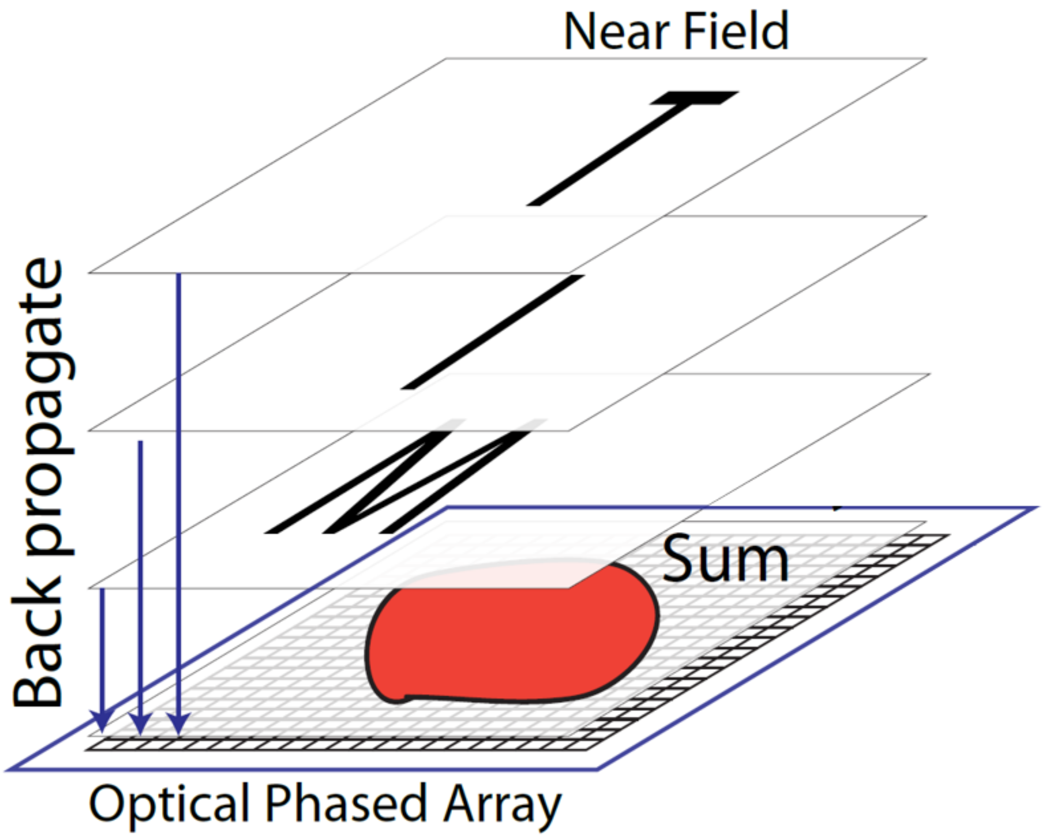

Fig. 2-2 Overview of back propagation and summation of electric fields

Here, we approximate the phased array emission to be an aperture rather than a point source due to the primarily z-oriented emission direction of our previously designed antennas [27, 46]. A point source, on the other hand, would have emission in the x and y directions in addition to the z direction. This x and y directional emission would be a problematic design for an antenna because it would cause each phased array antenna to have an effect on all of the other antennas within the plane of the array. This greatly complicates the calculation of the required phase and amplitude of each antenna because the effects of the surrounding antennas must also be taken into consideration. This approximation allows us to not take into account the antenna emission pattern.

Once the required parameters are ensured for each antenna, the phased array will then be able to emit the total summed electric field. If the emitted electric field is now forward propagated in the z direction, each of the original desired planes in the 3D space will be approximately recreated. Each plane, however, will not be an exact match to the desired output due to the discretization of the field and the contributions from the other output planes. This back propagation technique approximately models the capture of a hologram using light propagation as done in digital holography.

Chapter 3

Diffraction/Fresnel Equation

This design technique relies critically on a proper equation to accurately calculate electric fields being propagated both forwards and backwards. It is important to derive a proper expression for this propagation since it is utilized at various stages of the algorithm. The equation must fit into the stringent requirements of this method, but most importantly, the final expression must ensure that the propagation is accurate within the near field. This chapter will focus on deriving an accurate representation of the Fresnel propagation of electric fields to be used in the rest of this work. It closely follows the derivation of the Fresnel equation as done by Goodman [47].

We begin our derivation of the required diffraction equations by first examining the Helmholtz Equation. Since we are trying to determine the equation of a light pattern at an arbitrary distance P and time t, we can express the function as u(P,t). This scalar field can be expressed as

$$u(P,t) = Re\{U(P)e^{-j2\pi vt}\} \tag{3.1}$$

for a monochromatic wave, where v is the optical frequency. In order to represent an optical wave, it must also satisfy the scalar wave equation

$$\nabla^2 u - \frac{n^2}{c^2}\frac{\partial^2 u}{\partial t^2} = 0 \tag{3.2}$$

Here, ∇^2 is the Laplacian operator, n is the refractive index of the medium of light propagation, and c is the speed of light. By plugging in (3.1) into (3.2) we get

$$\nabla^2 U - \left[\frac{n}{c}(-j2\pi v)\right]^2 U = 0 \tag{3.3}$$

which simplifies to

$$(\nabla^2 + \mathrm{k}^2)U = 0 \tag{3.4}$$

where

$$k = 2\pi n\frac{v}{c}$$

We begin to formulate our diffraction equation for a complex field at a point in space using Green's theorem, which is often regarded as the primary foundation for scalar diffraction theory. It says:

For two complex valued functions of position, U and G, we have a closed surface S surrounding a volume V such that if U, G, and their first and second partial derivatives are single valued and continuous within and over S, then the following is true,

$$\iiint_V (U\nabla^2 G - G\nabla^2 U)dv = \iint_S (U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n})ds \tag{3.5}$$

where the $\frac{\partial}{\partial n}$ is a partial derivative with respect to n, the outward normal direction at each point on S.

In order to apply this to diffraction, we must ensure that a proper G is selected, which we can solve for using the Helmholtz and Kirchhoff integral theorem. This theorem provides a solution for the homogenous wave equation at any arbitrary point in terms of the solution itself and its first derivative on a closed surface around the point. We can follow Kirchhoff and express the optical disturbance at the point of observation P_0 in terms of its values on S, an arbitrary closed surface around P_0. We can select a spherical wave expanding around P_0 with unit amplitude such that Kirchhoff's G at any arbitrary point P_1 can be expressed as

$$G(P_1) = \frac{e^{jkr_{01}}}{r_{01}} \tag{3.6}$$

Where r_{01} is the length of the vector that points from P_0 to P_1.

In order to use this G in Green's theorem, we must ensure it is continuous within the volume V. A small spherical surface with radius ε is inserted to avoid the discontinuity at P_0 as shown in Fig. 3-1, making the surface of integration a composition of the original S and this new sphere

$$S' = S + S_\epsilon \tag{3.7}$$

This G within the new volume of integration V' between S and S_ϵ satisfies the Helmholtz equation as it is simply an expanding spherical wave

$$(\nabla^2 + \mathrm{k}^2)G = 0 \tag{3.8}$$

By plugging in (3.4) and (3.8) into the left side of Green's theorem, we get

$$\iiint\limits_{V'} (U\nabla^2 G - G\nabla^2 U)dv = -\iiint\limits_{V'} (UGk^2 - GUk^2)dv = 0 \tag{3.9}$$

This result allows us to set the right side to zero

$$\iint\limits_{S'} (U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n})ds = 0 \tag{3.10}$$

which we can expand by breaking up S' using (3.7)

$$-\iint\limits_{S_\epsilon} (U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n})ds = \iint\limits_{S} (U\frac{\partial G}{\partial n} - G\frac{\partial U}{\partial n})ds \tag{3.11}$$

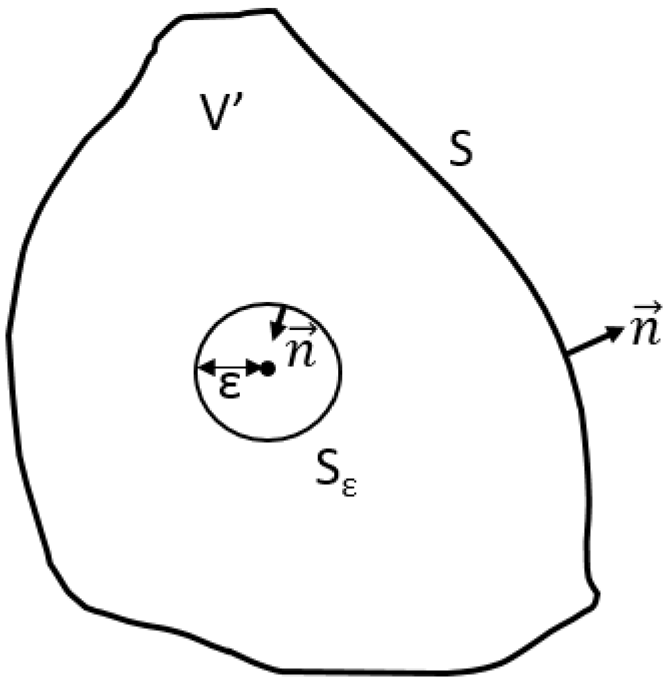

Fig. 3-1 Total surface of integration with small sphere of radius ε around P$_0$ to avoid discontinuity

As previously proven, we have

$$G(P_1) = \frac{e^{jkr_{01}}}{r_{01}}$$

and thus we also have,

$$\frac{\partial G(P_1)}{\partial n} = \cos(\vec{n}, \vec{r}_{01})(jk - \frac{1}{r_{01}})\frac{e^{jkr_{01}}}{r_{01}} \tag{3.12}$$

where $\cos(\vec{n}, \vec{r}_{01})$ is the cosine of the angle between the outward normal $\vec{n}$ and the vector $\vec{r}_{01}$ connecting P$_0$ and P$_1$. When we look at the specific P$_1$ on S_ϵ, we have $\cos(\vec{n}, \vec{r}_{01}) = -1$ and if we let ϵ approach 0 we get

$$\lim_{\epsilon\to 0}\iint_{S_\epsilon}(U\frac{\partial G}{\partial n}-G\frac{\partial U}{\partial n})\,ds \tag{3.13}$$

$$=\lim_{\epsilon\to 0}4\pi\epsilon^2\left[U(P_0)\frac{e^{jk\epsilon}}{\epsilon}\left(\frac{1}{\epsilon}-jk\right)-\frac{\partial U(P_0)}{\partial n}\frac{e^{jk\epsilon}}{\epsilon}\right] \tag{3.14}$$

which simplifies to

$$4\pi U(P_0) \tag{3.15}$$

because the rest of the terms go to zero as ϵ approaches 0. If we then plug this into the previously expanded Green's Theorem (3.11), we get

$$U(P_0)=\frac{-1}{4\pi}\iint_S(U\frac{\partial G}{\partial n}-G\frac{\partial U}{\partial n})ds \tag{3.16}$$

$$U(P_0)=\frac{1}{4\pi}\iint_S[(\frac{e^{jkr_{01}}}{r_{01}})\frac{\partial U}{\partial n}-U\frac{\partial}{\partial n}(\frac{e^{jkr_{01}}}{r_{01}})]ds \tag{3.17}$$

This is known as the Helmholtz and Kirchhoff integral theorem, and it is important to the scalar theory of diffraction because of its ability to express the field at any point P_0 in terms of the boundary values of the wave on a closed surface around it.

In order to move forward, we will examine the diffraction of light by an aperture. This will help us more accurately approximate the electric field due to our phased array. We select a surface of integration consisting of a plane surface, S_1, and a circle of radius R around P_0 cut off by S_1, which we will call S_2 as shown in Fig. 3-2.

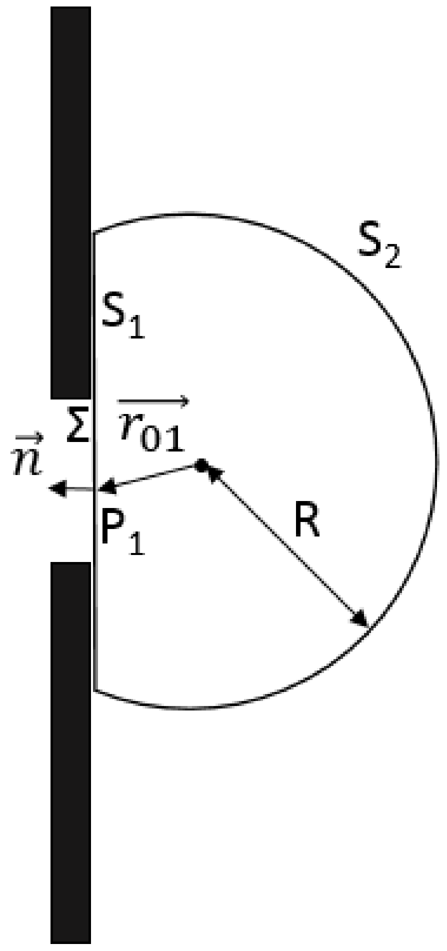

Fig. 3-2 Depiction of diffraction using aperture to solve Kirchhoff formulation

(3.17) then becomes

$$U(P_0) = \frac{1}{4\pi} \iint\limits_{S_1+S_2} (G\frac{\partial U}{\partial n} - U\frac{\partial G}{\partial n})ds \qquad (3.18)$$

Using G from (3.6) that we calculated previously, we have

$$G = \frac{e^{jkr_{01}}}{r_{01}} = \frac{e^{jkR}}{R} \qquad (3.19)$$

on S_2. We similarly get

$$\frac{\partial G}{\partial n} = (jk - \frac{1}{R})\frac{e^{jkR}}{R} \approx jkG \tag{3.20}$$

for large R. This part of the integral then simplifies to

$$\iint_{S_2} [G\frac{\partial U}{\partial n} - U(jkG)]ds = \int_{\Omega} G(\frac{\partial U}{\partial n} - jkU)R^2 d\omega \tag{3.21}$$

where Ω represents the angle that subtends the closed surface S_2 from our point of observation P_0. This result proves that |RG| is bounded on S_2 and the entire integral approaches 0 on S_2 as R becomes large, so long as the Sommerfeld radiation condition is met

$$\lim_{R\to\infty} R(\frac{\partial U}{\partial n} - jkU) = 0 \tag{3.22}$$

However, we know this to be true as the disturbance U vanishes at least as fast as a diverging spherical wave. With the S_2 integration now eliminated, we can express the electric field at P_0 in terms of the integration over S_1,

$$U(P_0) = \frac{1}{4\pi}\iint_{S_1} (G\frac{\partial U}{\partial n} - U\frac{\partial G}{\partial n})ds \tag{3.23}$$

Now, to isolate the portion of S_1 that would correspond to our phased array, we use Kirchhoff boundary conditions to write the integration over Σ, the aperture in Fig. 3.2. The boundary conditions say:

1. Across the surface Σ, the field distribution U and its derivative $\frac{\partial U}{\partial n}$ are exactly the same as they would be in the absence of the screen.
2. Over the portion of S_1 that lies in the geometrical shadow of the screen, the field distribution U and its derivative $\frac{\partial U}{\partial n}$ are identically zero.

The first point guarantees that the integration over the aperture is valid without the screen, while the second point ensures that we can eliminate the surface of integration except the part within the aperture. This shape at P_1 now resembles the desired emission of a phased array to a point P_0. Our electric field at P_0 can therefore be further simplified to

$$U(P_0) = \frac{1}{4\pi} \iint_{\Sigma} (G\frac{\partial U}{\partial n} - U\frac{\partial G}{\partial n}) ds \tag{3.24}$$

We can attempt to simplify the Green's function G such that either G or $\frac{\partial G}{\partial n}$ disappears over the surface S_1. Sommerfeld showed that a Green's function can be generated by having two point sources P_0 and $\tilde{P}_0$ equally spaced from and on opposite sides of S_1 as shown in Fig. 3-3. If both have the same wavelength and oscillate with a 180° phase difference, then our G becomes

$$G\_(P_1) = \frac{e^{jkr_{01}}}{r_{01}} - \frac{e^{jk\tilde{r}_{01}}}{\tilde{r}_{01}} \tag{3.25}$$

which will disappear on our aperture Σ, leaving us with only the $\frac{\partial G}{\partial n}$ term

$$U_I(P_0) = \frac{-1}{4\pi} \iint_{\Sigma} (U\frac{\partial G\_}{\partial n}) ds \tag{3.26}$$

Now to solve for the $\frac{\partial G\_}{\partial n}$ term, we can take the derivative of G_(P_1) to get

$$\frac{\partial G\_}{\partial n}(P_1) = \cos(\vec{n}, \vec{r}_{01}) \left(jk - \frac{1}{r_{01}}\right) \frac{e^{jkr_{01}}}{r_{01}} - \cos(\vec{n}, \vec{\tilde{r}}_{01})(jk - \frac{1}{\tilde{r}_{01}}) \frac{e^{jk\tilde{r}_{01}}}{\tilde{r}_{01}} \tag{3.27}$$

With our surface our P_1 on S_1, we can state

$$r_{01} = \tilde{r}_{01}$$

$$\cos(\vec{n}, \vec{r}_{01}) = -\cos(\vec{n}, \vec{\tilde{r}}_{01})$$

This allows us to write,

$$\frac{\partial G\_}{\partial n}(P_1) = 2\cos(\vec{n}, \vec{r}_{01}) \left(jk - \frac{1}{r_{01}}\right) \frac{e^{jkr_{01}}}{r_{01}} \tag{3.28}$$

and as long as $r_{01} \gg \lambda$, we can reduce this to become

$$\frac{\partial G_-}{\partial n}(P_1) = 2jk\cos(\vec{n}, \vec{r}_{01})\frac{e^{jkr_{01}}}{r_{01}} \tag{3.29}$$

since the $\frac{1}{r_{01}}$ will be small relative to the *jk* term.

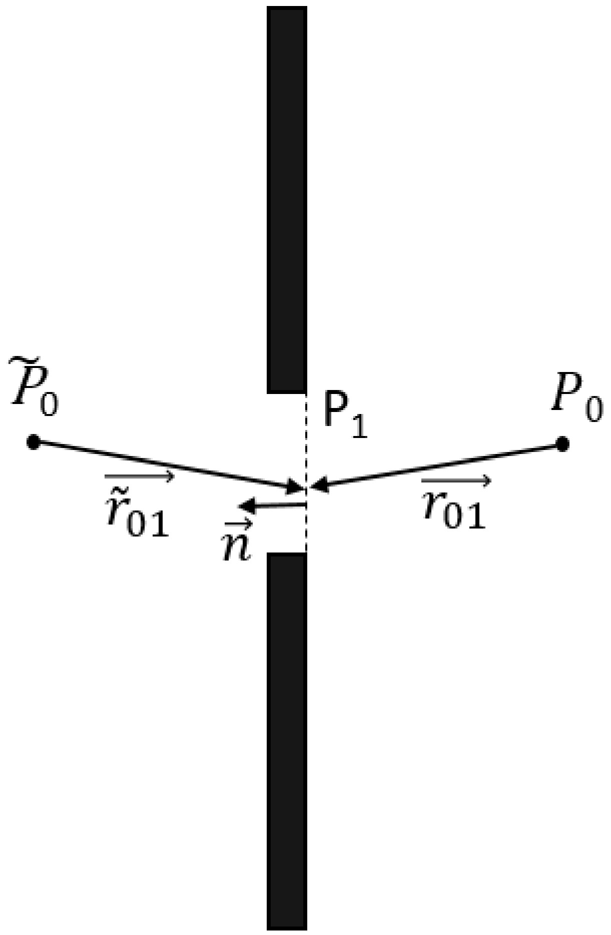

Fig. 3-3 Formulating G_ using opposing point sources to simplify electric field at P_0

If we use this G_ in (3.24) and utilize the fact that G_ approaches 0, our equation becomes,

$$U_I(P_0) = \frac{1}{j\lambda}\iint\limits_{S_1} \left(U(P_1)\cos(\vec{n},\vec{r}_{01})\frac{e^{jkr_{01}}}{r_{01}} \right) ds \tag{3.30}$$

and using the Kirchhoff boundary conditions it becomes

$$U_I(P_0) = \frac{1}{j\lambda}\iint\limits_{\Sigma} \left(U(P_1)\cos(\vec{n},\vec{r}_{01})\frac{e^{jkr_{01}}}{r_{01}} \right) ds \tag{3.31}$$

where the Σ area can emulate the area of emission from our phased array. We can thus transform this equation into its rectangular components by substituting a θ, the angle between the outward normal $\vec{n}$ and the vector $\vec{r}_{01}$ from P_0 to P_1, into the cosine, giving us

$$U_I(P_0) = \frac{1}{j\lambda}\iint\limits_{\Sigma} \left(U(P_1)\cos(\theta)\frac{e^{jkr_{01}}}{r_{01}} \right) ds \tag{3.32}$$

We can readily see from Fig. 3-4 that

$$\cos(\theta) = \frac{z}{r_{01}}$$

which can be substituted to give us

$$U(x,y) = \frac{z}{j\lambda}\iint\limits_{\Sigma} \left(U(\xi,\eta)\frac{e^{jkr_{01}}}{{r_{01}}^2} \right) ds \tag{3.33}$$

where ξ and η correspond to the coordinate system of the input plane and x and y correspond to the coordinate system of the output plane, and we define r_{01} to be

$$r_{01} = \sqrt{z^2 + (x-\xi)^2 + (y-\eta)^2} \tag{3.34}$$

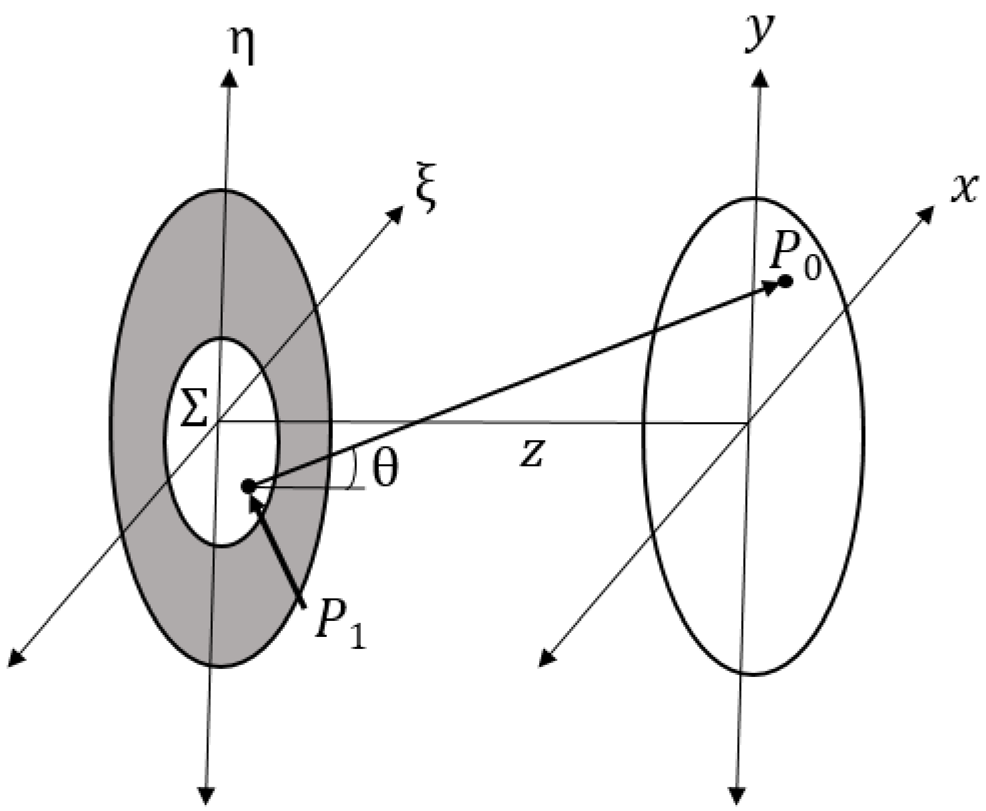

Fig. 3-4 Diffraction from P_1 to P_0 in rectangular coordinates

This r_{01} term can be expanded into its Taylor expansion by using

$$\sqrt{1+a} = 1 + \frac{a}{2} - \frac{a^2}{8} + \cdots \tag{3.35}$$

to give

$$r_{01} = z\sqrt{1 + \frac{(x-\xi)^2 + (y-\eta)^2}{z^2}} \tag{3.36}$$

$$= z\left[1 + \frac{(x-\xi)^2 + (y-\eta)^2}{2z^2} - \frac{((x-\xi)^2 + (y-\eta)^2)^2}{8z^4} + \cdots\right] \tag{3.37}$$

$$= z + \frac{(x-\xi)^2 + (y-\eta)^2}{2z} - \frac{((x-\xi)^2 + (y-\eta)^2)^2}{8z^3} + \cdots \tag{3.38}$$

From this point, if we can prove that the 3rd term is negligible, and thus all subsequent terms are negligible, then we can reduce this to only the first two terms. This is traditionally proven by showing that dropping the third term causes a maximum phase change of less than one radian

$$8z^3 \gg \frac{\pi}{4\lambda}(\max((x-\xi)^2 + (y-\eta)^2))^2 \tag{3.39}$$

However, this requirement is overly stringent and the approximation can still yield accurate results as long as it doesn't have a large effect on the Fresnel diffraction integral. This can be ensured by making sure that the integral is primarily coming from points such that $x \approx \xi$ and $y \approx \eta$.

Alternatively, when dealing with optical wavelengths, the wavelength is many orders smaller than the physical dimensions

$$\lambda \ll z, \lambda \ll \sqrt{(x-\xi)^2 + (y-\eta)^2}$$

Then, practically, we can reduce the r_{01} term as long as

$$\sqrt{(x-\xi)^2 + (y-\eta)^2} \ll z \tag{3.40}$$

It is possible to further reduce our r_{01} term to only the z term, but it is important to be careful where this is done. It is possible to reduce the r_{01} to only a z in the denominator of (3.33) because the error is generally recognized to be small, however, since the r_{01} in the exponent is multiplied by a k term, which is relatively large, we cannot further reduce this term.

Using these reductions, we arrive at the Fresnel Approximation of the Huygens-Fresnel principle.

$$U(x,y) = \frac{e^{jkz}}{j\lambda z} \iint_{-\infty}^{\infty} (U(\xi,\eta) e^{jk\frac{(x-\xi)^2+(y-\eta)^2}{2z}}) d\xi d\eta \tag{3.41}$$

This can quickly be converted to a convolution equation

$$U(x,y) = \iint_{-\infty}^{\infty} (U(\xi,\eta)h(x-\xi, y-\eta)d\xi d\eta \tag{3.42}$$

where

$$h(x,y) = \frac{e^{jkz}}{j\lambda z} e^{jk\frac{(x)^2+(y)^2}{2z}} \tag{3.43}$$

and its Fourier transform can be written

$$H(f_x, f_y) = e^{jkz} e^{-j\pi\lambda z({f_x}^2+{f_y}^2)} \tag{3.44}$$

Here, reaching the Fresnel equations further connects our work with holography as hologram propagation is usually described with these same equations [29]. Furthermore, finding a Fourier transform equation of the integral also lines up with holography as it is often used to calculate hologram projection. These expressions emulating near field diffraction by utilizing the Fresnel approximation are heavily relied upon to calculate our propagation of electric fields.

Chapter 4

Simulations

With the concept of the method set and the proper diffraction equations derived, the next logical step is to put the process to the test and see how it works. The simulations in this section will continue using the example of designing a phased array to emit the MIT logo with the individual letters spaced at different distances from the phased array. First, we must ensure that we are operating within the proper distances so that we are in the near field. Then, we are ready to simulate the results of our design algorithm to see how well they match up with the desired outputs. After evaluating the initial results, we are able to introduce some improvements in order to enhance the output picture as well as show additional simulations to demonstrate the flexibility of the algorithm.

4.1. Finding an Accurate Output Distance

This method utilizes near field diffraction in order to generate output planes that can vary with distance from the phased array to generate a 3D hologram. In order to operate in the near field, a proper range of distances must be determined to ensure that these benefits can be utilized. The space of emission of the phased array is commonly separated into the reactive near-field, radiating near-field, and the far field [48]. First, we differentiate the far field and near field by Fraunhofer distance. This is defined to be the emission distance that separates these two regions when the antennas are larger than half of the wavelength that is being emitted, which is true for our design since we are operating at 1550 nm wavelength and have antennas that are approximately 3 µm by 3 µm. The Fraunhofer distance is defined to be

$$d_f = \frac{2D^2}{\lambda} \tag{4.1.1}$$

where D is the aperture size. This tells us that if we want to operate in the near field, we must select distances less than this Fraunhofer distance. Assuming a phased array aperture of 1.63 mm ($128\ antennas \times 9\mu m\ pitch \times \sqrt{2}$) and wavelength of 1550 nm we calculate this value to be approximately 3.42478m.

We also need to ensure that we avoid operating in the reactive near-field region because it is difficult to predict electric field behavior in this region. While the electromagnetic fields are being radiated, there is an additional reactive component accompanying it. This leads to a sensitivity to electromagnetic absorption in this region (reactive near-field) and back-coupling of the fields. What is commonly seen in this region is generally undeterminable results, which would be detrimental to the algorithm. The distance where this field transitions to the radiating near-field is approximately

$$\frac{\lambda}{2\pi} \tag{4.1.2}$$

This often competes with the previously calculated requirement (3.40) to give us a lower bound on the proper distance from the phased array. We calculate this value to be approximately 0.246 µm for the reactive to radiating near field transition and approximately 1.629 mm for the Fresnel approximation using the same 1550 nm wavelength and 1.63 mm aperture to correspond to greatest difference in x or y.

Simulated results have also shown that near field diffraction has the most stable intensity profiles between approximately 10 times and 100 times the aperture radii [49]. Calculating these values gives us 5.76 mm and 57.6 mm as a range which provides stable intensity profiles and matches with the previously calculated range to ensure operation in the near field.

4.2. Initial Simulation Results

Now we can utilize this knowledge of a valid range of distances from the phased array to operate in the near field and test the validity of our method. This will be done by utilizing MATLAB to simulate the various steps of the algorithm. We will design a 128x128 antenna phased array that will emit the MIT logo where each letter is in focus at a different distance from the phased array as seen in Fig. 4.2-1. Although our algorithm can design phased arrays for arbitrary wavelengths, we will design everything described in this thesis to operate at 1550 nm. This wavelength is commonly employed in silicon photonic devices as it is the wavelength of choice for many fiber optic telecommunication systems. Also, the antennas will be spaced 9 µm from each other in both the x and y directions to match the spacing that will be used in fabrication. This pitch corresponds to a multiple of half of the free-space wavelength, which limits the effects of higher-order interference patterns [48].

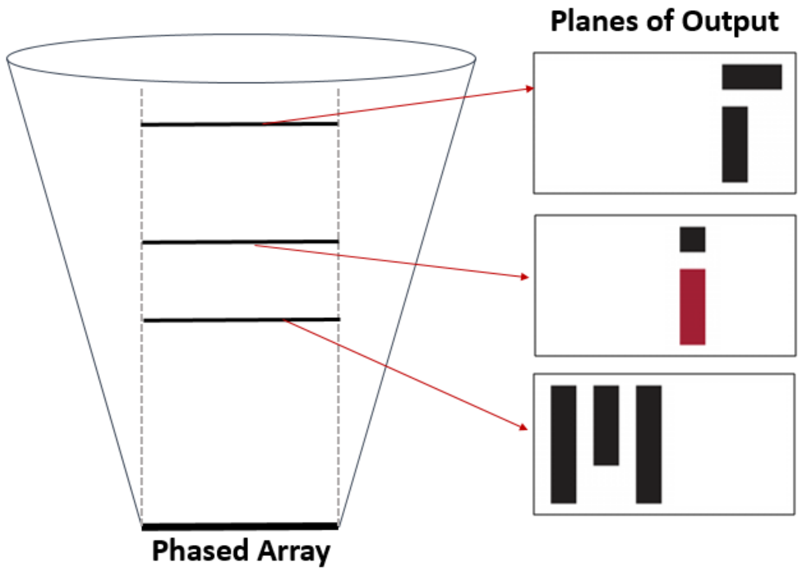

Fig. 4.2-1 Phased array design to emit MIT logo with letters focused at different distances

We take each output plane and convert it into an electric field that closely resembles the letters. For this initial simulation, we accomplish this by generating a binary mask where pixels that make up the letters are fully illuminated, meaning they have equal non-zero amplitude, and the rest of the pixels are given zero amplitude, effectively shutting them off. We design the M to focus at 7 mm from the phased array, the I to focus at 14 mm from the phased array, and the T to the focus at 28 mm from the phased array. Using the previously calculated diffraction equations, we back propagate the output planes to the theoretical location of the phased array (the M is back propagated 7 mm, the I is back propagated 14 mm, and the T is back propagated 28 mm). This back propagation, and future near field propagation, is accomplished by convolving our output with (3.43), or equivalently, we Fourier transform our output, multiply by (3.44), and then inverse Fourier transform the product to generate the propagated field.

$$E_{propagated} = \mathcal{F}^{-1}\left[\mathcal{F}(E_0) \times H\left(f_x, f_y\right)\right] \tag{4.2.1}$$

After summing the back propagated electric fields, we can extract the amplitude and phase profile, the latter which is shown in Fig. 4.2-2. Here, we quickly get an idea of how complex the phase profiles can be and how important it is to have an accurate and reliable method to calculate these fields to tune our phased array antennas.

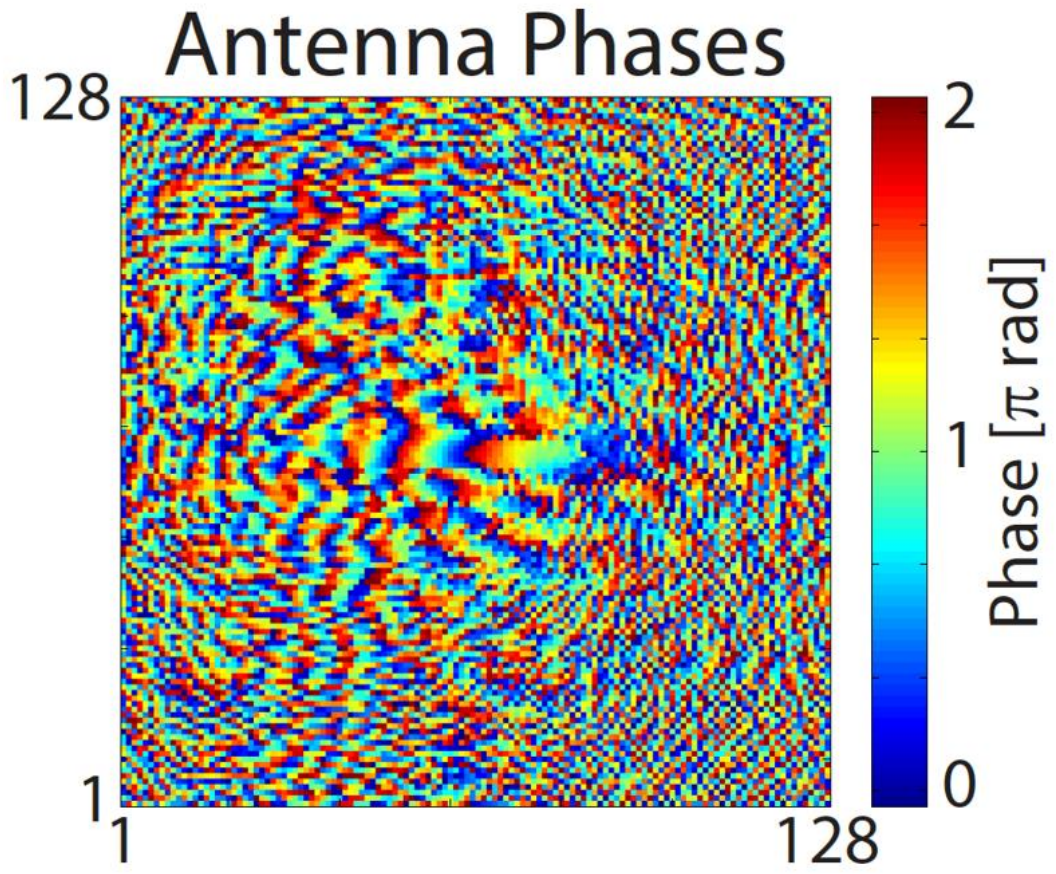

Fig. 4.2-2 Phase profile of summed field that emits the MIT logo

By approximating the phased array antenna emission to an aperture, we can forward propagate the summed field and see how well it recreates our desired output. Fig. 4.2-3 presents the simulation results when focusing at the desired output plane locations. It is evident that the correct letters are in focus at the desired output distances, however, we notice that we can see the effects of the other output letters. This actually makes sense in terms of how human vision interprets viewing things at varying depths. When we see objects at varying distances and focus on one, the objects around the focused object don't disappear. Instead, they will simply become out of focus, similar to what we see in these initial simulation results. However, in order to appeal to a wider range of applications, there is good reason to try to isolate each output. We realize that, since the intensity is the main focus of our output, we have the potential to modify the phase of the outputs in order to improve our results.

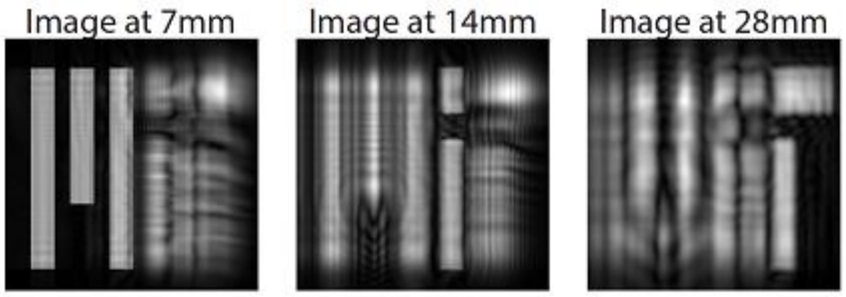

Fig. 4.2-3 Initial simulation results of MIT logo

4.3. Improvements to Method

With the ability to control phase we have the potential to isolate our outputs and look into other ways to utilize this degree of freedom. One issue that comes to mind is the fact that we have collimated light being output from the phased array. This is the main reason we see the effects from the other output planes. Also, there is the potential for loss of information during back propagation when the output plane is the same size as the area of the phased array. The cause of this is primarily due to the conical shape of diffraction. This section will investigate these issues and methods to improve the simulation results.

4.3.1. Randomized Phase

When we initially tested our method using the MIT letters, we did not modify the phase since we used a binary intensity profile as the electric fields, leading to each letter having a flat phase front. This meant that the phased arrays were designed to emit these outputs with all of the pixels having identical phases that are parallel to each other. The issue with this is that the beams are collimated due to the parallel phases and will not disperse greatly with distance, which explains why we can still basically make out the shape of the other letters when they aren't meant to be in focus.

In order to de-collimate the pixels, we add a random phase to all of the output planes. This points the pixels at random directions so that they would come together and be in focus at the desired location and then quickly spread out and disappear when we reached other output distances. A depiction of the two modes of light propagation is shown in Fig. 4.3.1-1 to show how randomized phase can work to improve isolation of outputs.

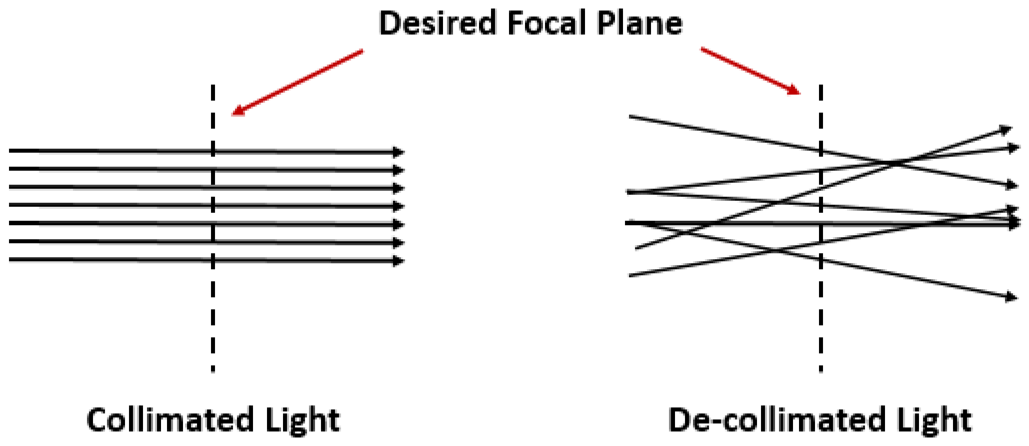

Fig. 4.3.1-1 Comparing collimated light and de-collimated light

We put this hypothesis to the test by simulating two circles each spaced at different distances from the phased array as shown in Fig. 4.3.1-2. Then we add a random phase front with variable randomization to the circle on the right such that

$$E'_{circle}(x,y) = E_{circle}(x,y)\left(e^{jk(Coef)(Rand(x,y))}\right) \tag{4.3.1.1}$$

where $E_{circle}(x,y)$ is the electric field distribution of the original circle, $E'_{circle}(x,y)$ is the new electric field distribution for the circle on the right, Rand(x,y) is a pseudorandom distribution with values in the interval (0,1), and Coef is the coefficient to our random distribution.

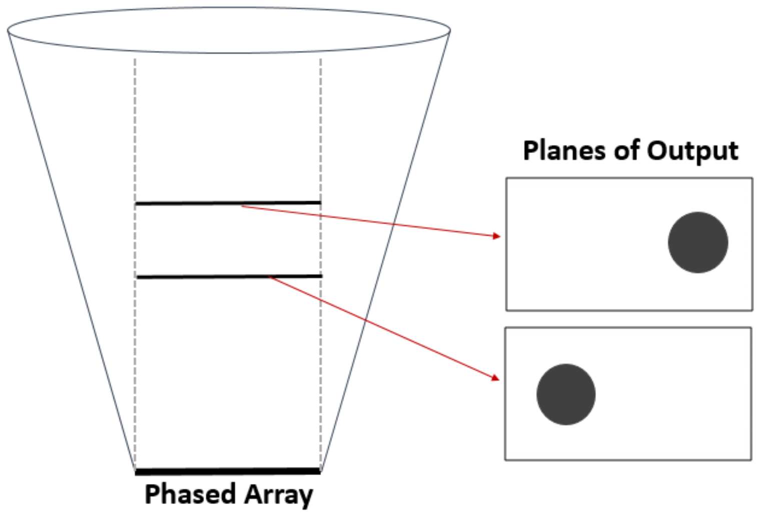

Fig. 4.3.1-2 Simulation setup to test randomized phase

Then, we take our two output electric fields and use our back propagation method to get our summed electric field. We then forward propagate our summed electric field and examine the output at the distance corresponding to the circle without added random phase being in focus. This allows us to see how randomized phase affects the dispersion of the other circle. Fig. 4.3.1-3 shows the simulated results as we increase the coefficient to the pseudorandom distribution from left to right.

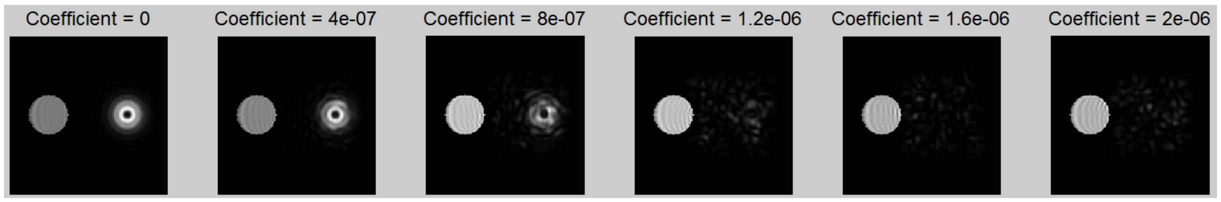

Fig. 4.3.1-3 Effects of randomized phase

At the far left we see the effects with the pseudorandom distribution zeroed out. The circle on the left is in focus and we can clearly see the out of focus circle on the right, similar to what we saw in our initial MIT simulation results. As we look from left to right, we see the effects of increasing the randomization. The circle on the right disperses more and more relative to the circle on the left until we can barely tell there was anything else there.

To get a calculation on the effectiveness of the randomized phase, we decided to calculate a weighted mean standard deviation of the error with the weights corresponding to the location of the pixels. This value gives us a number to describe the spread of the pixels of the circle with randomized phase. The weighted mean standard deviation is defined to be

$$\sigma_{weighted} = \sqrt{\frac{\sum_{i=1}^{N} w_i (x_i - \mu^*)^2}{\sum_{i=1}^{N} w_i}} \tag{4.3.1.2}$$

Where w_i represents the weight and μ^* is the weighted mean. For our case, we have a 2D weight and x_i so we modify this equation so that we sum the values within the matrix and then sum the results for x and y weightings. Here, the weightings are a normalized error between the optimal single circle output and the simulated result. The x_i values simply represent positional values in either the x or y direction. Fig. 4.3.1-4 shows the plot of this calculated value with respect to the randomization coefficient.

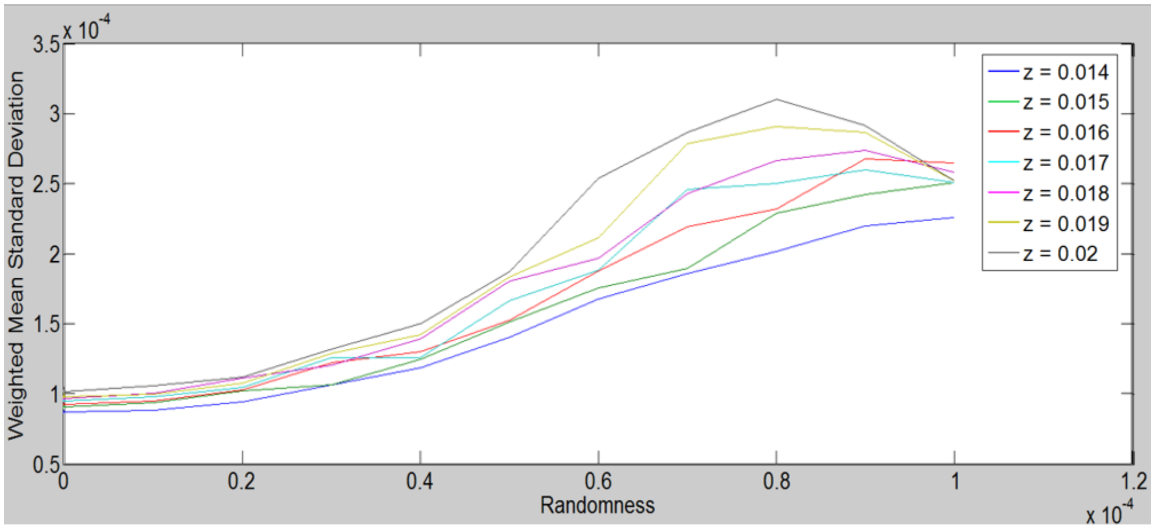

Fig. 4.3.1-4 Spread due to randomized phase at varying distances from phased array

In order to investigate the effects of the distance of the second circle, this plot also shows the calculation with the randomized circle at varying distances from the phased array. The plot shows that as we increase the distance of the circle with random phase from the phased array, the spread of the error also increases. This tells us that we actually need less randomization as we increase distance in order to achieve the same amount of dispersion of the unwanted components of the output. We visually see that when the standard deviation is about 2×10^{-4}, the unwanted circle disperses enough to not be noticeable anymore. Fitting the required randomization coefficient needed to achieve this threshold value can be seen in Fig. 4.3.1-5. This fitting shows that we have an approximately inverse relationship between the distance from the phased array and the required randomization coefficient to allow for a good dispersion of the output,

$$Coef \propto \frac{1}{z} \tag{4.3.1.3}$$

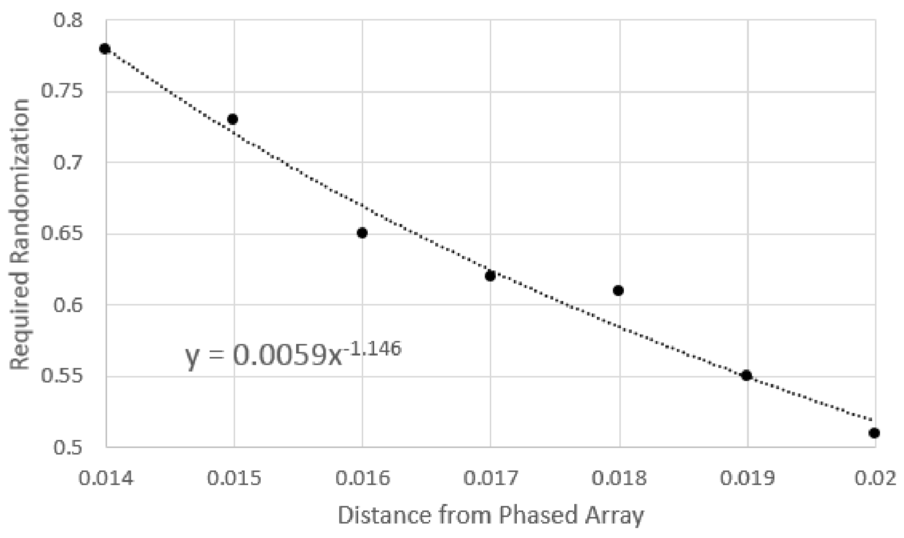

Fig. 4.3.1-5 Relationship between distance and required randomization

This tells us that our final randomized phase profile should have the form

$$\varphi_{random}(x,y) = \frac{A}{z}(Rand(x,y)) \tag{4.3.1.4}$$

where A is some constant and Rand(x,y) is the same random distribution of values in the interval (0,1).

These simulations help to prove that our decision to add random phase to the output planes allow us to better isolate our outputs by de-collimating the light.

4.3.2. Curved Phase

During back propagation, some of the power can potentially be lost outside of the area of the phased array. This is due to the fact that the diffraction can cause the beam to diverge slightly. Losing power outside the area of our phased array equates to losing critical information that would be used to generate our desired outputs. Obviously we could just increase the size of our phased array, but this means putting a limit on the size of the output relative to the phased array. This is a huge limitation because our output would always have to be smaller than the size of our phased array to ensure optimal recreation of the desired output.

We propose that we can add a curved phase front to the output image in order to better focus the back propagated power into the desired area of the phased array. In order to test this hypothesis, we simulated the back propagation of a circle to see the effects with and without curved phase. Fig. 4.3.2-1 shows this concept of having a curved phase front. We see theoretically how some of the power can be lost during back propagation if the phased array is the same size as the output and how adding a curved phase front can fix this.

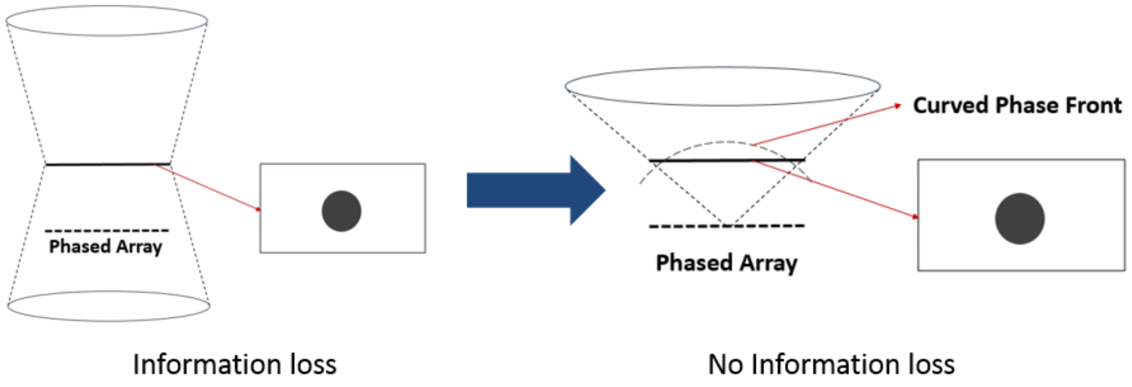

Fig. 4.3.2-1 Reducing information loss by adding curved phase front

First, we will derive a curved phase front to be added to our output planes. Fig. 4.3.2-2 shows what this theoretical curved phase front would look like facing downwards in order to focus the back propagation towards the center of the phased array. We determine that the required phase front at the output plane must be equivalent to a spherical phase front. This can be calculated by starting with the equation for a sphere

$$\sqrt{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}=r^2 \tag{4.3.2.1}$$

Since we want to have the power concentrated towards the center of our phased array, we center the sphere at $x_0 = y_0 = z_0 = 0$ and set the radius equal to the distance between the output plane and the phased array $(r = z_i)$

$$\sqrt{x^2+y^2+z^2}=z_i \tag{4.3.2.2}$$

We can now rearrange this equation to resemble a mapping of a phase front

$$\varphi_{desired}(x,y)=\sqrt{x^2+y^2+{z_i}^2}-z_i \tag{4.3.2.3}$$

Note that we replaced the z term with z_i since the output pixels are all on a plane that is a fixed distance z_i from the phased array. (4.3.2.3) allows us to simplify our desired phase to

$$\varphi_{desired}(x,y) = z_i \times \left(\sqrt{\frac{x^2 + y^2}{{z_i}^2} + 1} - 1 \right) \tag{4.3.2.4}$$

A quick check at the center of our output plane ($x = y = 0$, $z = z_i$) outputs a 0, which fits our desired curved phased front as the phase should be flat over the center.

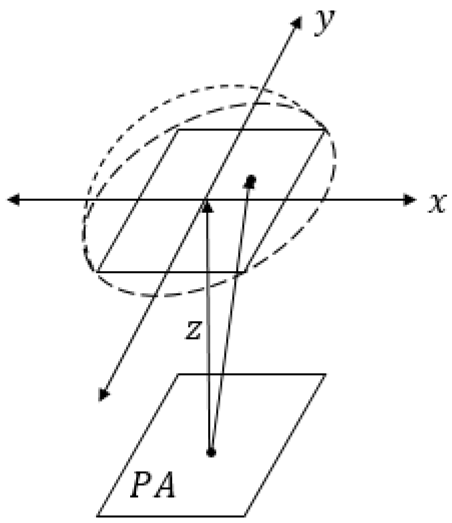

Fig. 4.3.2-2 Curved phase front facing phased array

The simulation to test our theory designs a phased array to emit a circle with diameter equal to the side of the phased array as seen in Fig. 4.3.2-3a where the blue square represents the area of the phased array. The circle is designed to focus in at 28 mm from the phased array so we can use our near field diffraction equations to back propagate it to the theoretical phased array location. The area of operation was expanded to see how power spreads outside of the desired phased array area.

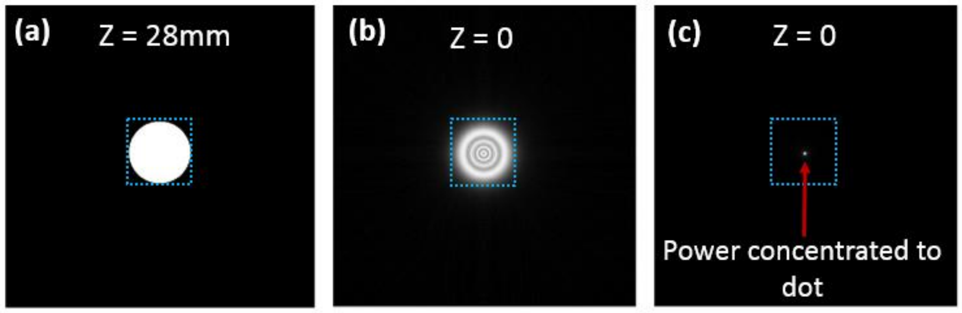

Fig. 4.3.2-3 (a) Desired circular output. **(b)** Result of back propagating (a) to theoretical phased array with flat phase. **(c)** Result of back propagating (a) with added circular phase front.

Fig. 4.3.2-3b shows how the power is spread when back propagated the 28 mm to the theoretical location of the phased array using the original unaltered flat phase. We see power is lost outside the desired phased array area, which could be detrimental to an accurate reproduction of our desired output. The back propagation when adding our derived curved phase is shown in Fig. 4.3.2-3c. We see that when we add the curved phase front, the power is concentrated much more finely into the area of the phased array.

In order to better compare the improvement due to the curved phase front, we calculated the percentage of power lost outside of the phased array when back propagated. Fig. 4.3.2-4a shows the plot of power loss at varying distances of back propagation when we use our original flat phase. It shows that when we reach the 28 mm of back propagation, corresponding to the location of the phased array, that there is approximately 5% of the original power lost outside of the phased array area. This equates to losing 5% of the information that is required to accurate regenerate our output. Similarly, Fig. 4.3.2-4b shows the same plot when we add the curved phase front to the output. Here, there is only approximately 1% power loss when we reach the phased array.

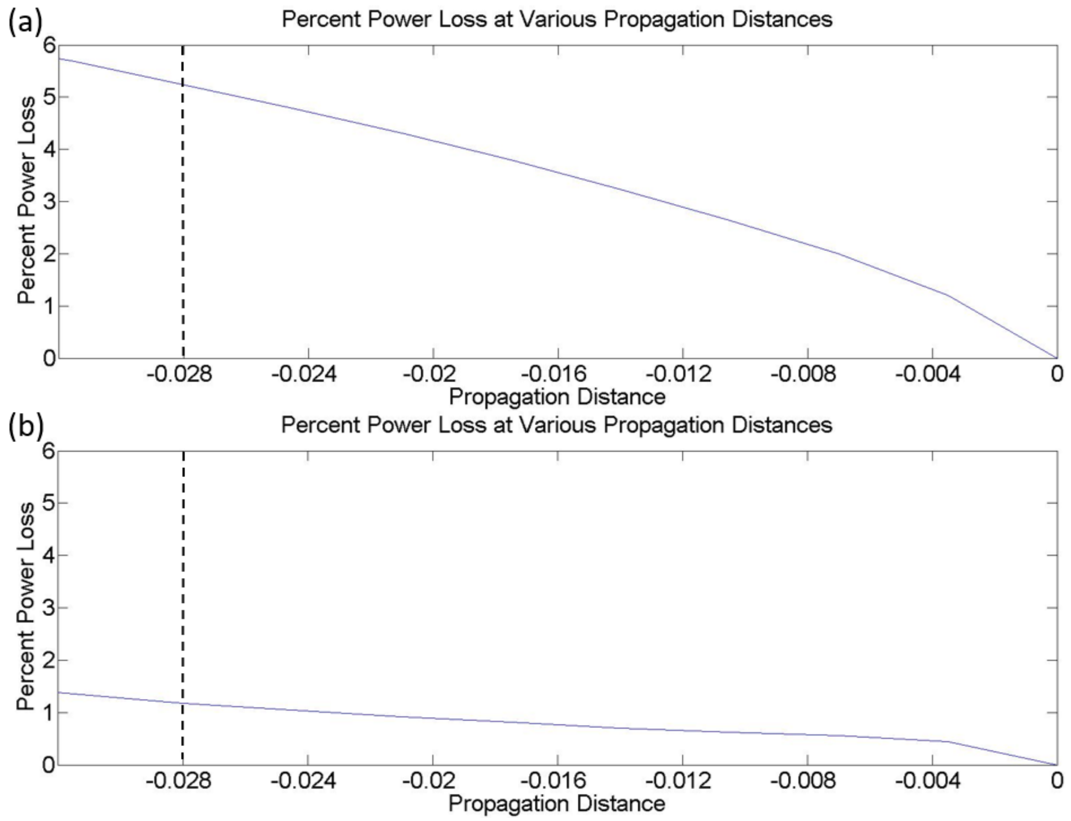

Fig. 4.3.2-4 (a) Calculated percent power lost outside of desired phased array area with varying back propagation distances with unaltered phase (phased array location marked with dotted line). **(b)** Calculated percent power lost with curved phase front.

In order to further investigate the potential of the curved phase front, we also simulated the effects when the output is larger than our phased array area. Fig. 4.3.2-5a shows the desired circular output with diameter equal to double the length of the side of the phased array, double the size of the circle used in the previous simulation. The loss of power is much more pronounced when back propagating with unaltered phase as shown in Fig. 4.3.2-5b. There is a significant amount of power that is clearly outside the area of the phased array. On the other hand, Fig. 4.3.2-5c shows that back propagating using the curved phase front continues to give us concentrated power within the boundaries of the phased array.

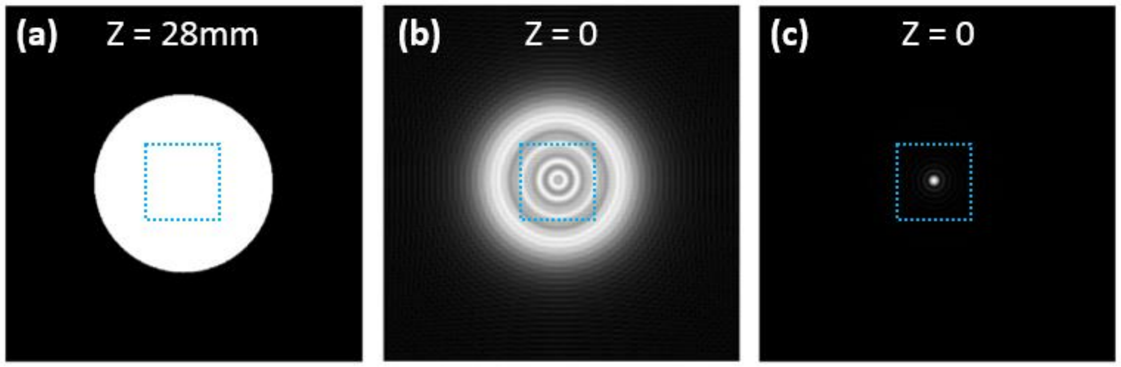

Fig. 4.3.2-5 (a) Desired larger circular output. **(b)** Result of back propagating (a) to theoretical phased array with flat phase. **(c)** Result of back propagating (a) with added circular phase front.

Calculating the same percentage of power lost provides more concrete numbers to these simulation results. Fig. 4.3.2-6a shows that back propagating the larger circular output with unaltered phase to the phased array results in approximately 67.4% of the power being lost outside of the phased array. This amounts to only having 32.6% of the required information to attempt to recreate the desired output. Fig. 4.3.2-6b shows that adding the curved phase front only leads to approximately a 2.5% power loss when back propagated, a much smaller loss compared to that of the flat phase back propagation.

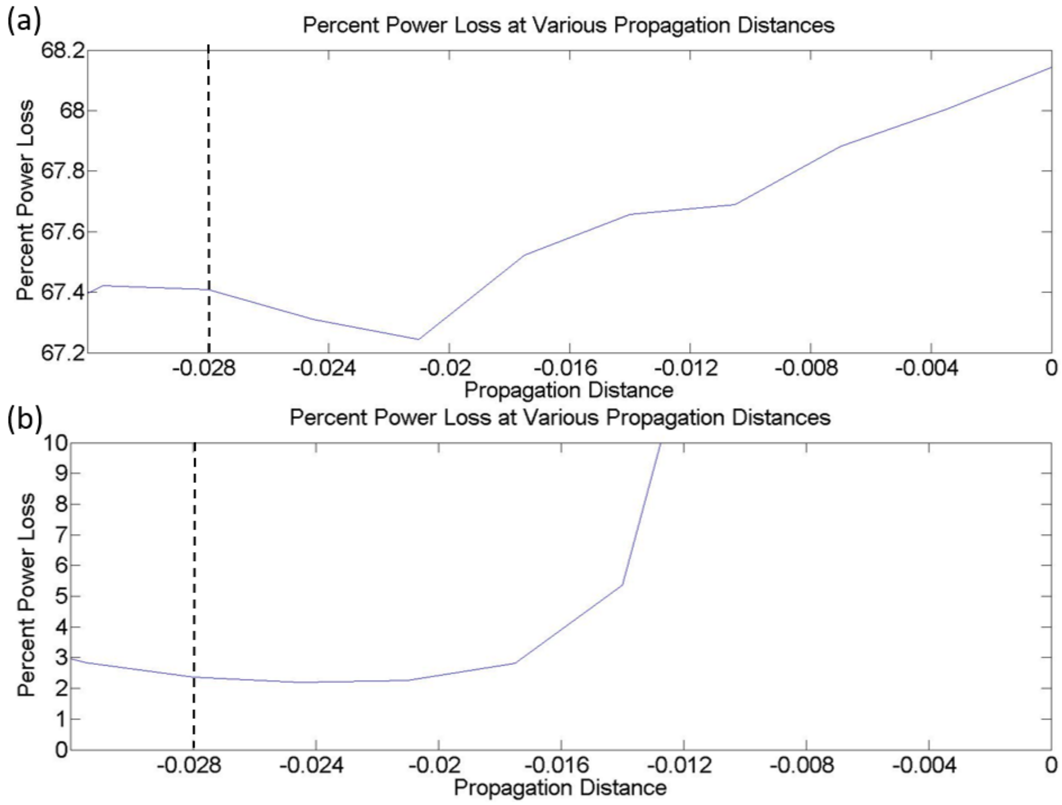

Fig. 4.3.2-6 (a) Calculated percent power loss of larger output with unaltered phase (phased array location marked with dotted line). **(b)** Calculated percent power loss with curved phase front

Finally, we investigate how detrimental this loss of information can be by reusing the simulations of the larger circle output and seeing what the phased arrays would emit using our method. We take the back propagated fields from Fig. 4.3.2-5 and use the 128x128 pixels in the center to tune our phased array parameters. Fig. 4.3.2-7b shows that when we try to forward propagate the field generated using a flat phase front, the output is severely degraded compared to the desired output in Fig. 4.3.2-7a. When we use a phased array designed from the back propagated output with curved phase front, the output much more closely resembles our desired output as shown in Fig. 4.3.2-7c.

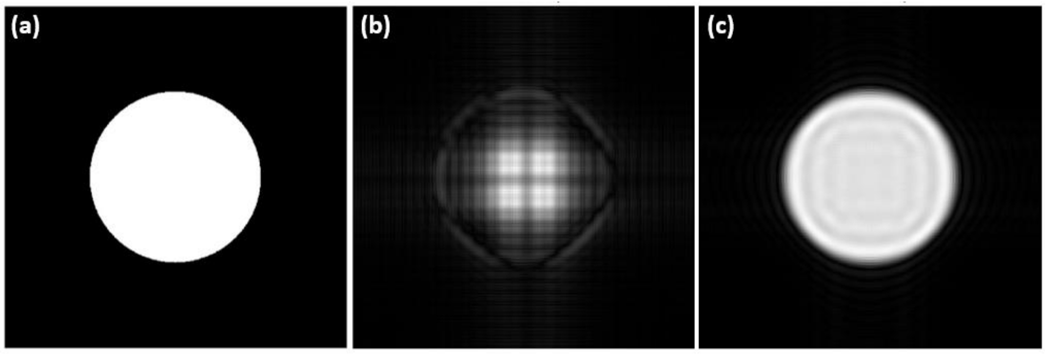

Fig. 4.3.2-7 (a) Desired circular output. **(b)** Output due to phased array designed with back propagated flat phase. **(c)** Output due to phased array designed with back propagated curved phase front

These simulations show us that we can utilize a curved phase front on our outputs prior to back propagation to improve the output images. The curved phase front works to both reduce power lost during back propagation and, more importantly, it allows us to generate outputs larger than the dimensions of our phased array.

4.4. Refined Simulation Results

In this section we combine the proposed improvements to our method in an attempt to better isolate our output images. We add the randomized phase front and the curved phase front to our output planes in order to reduce the effects due to collimated light and back propagation loss. Our output electric fields then look like

$$E_{output}(x,y) = I_{output}(x,y) \times e^{jk(\varphi_{curved}(x,y)+\varphi_{random}(x,y))} \tag{4.4.1}$$

Where I_{output} represents the intensity profile of the output, k is our k-vector of propagation, and our two phase terms are φ_{curved} and φ_{random} from (4.3.2.4) and (4.3.1.4), respectively. If we write out the full equations, we can pull out a z term from the exponential

$$E_{output}(x,y) = I_{output}(x,y) \times e^{jkz[(\sqrt{\frac{x^2+y^2}{z^2}+1}-1)+\frac{A}{z^2}(Rand(x,y))]} \tag{4.4.2}$$

We then implement this phase term into our initial MIT logo simulation and see the comparison of the outputs in Fig. 4.4-1. We see that the effects of the other letters are not as clearly visible in the output planes compared to how they previously were. The letters seem to be better isolated in the sense that it is hard to make out what exists in the other outputs, whereas it is much clearer in the original simulation.

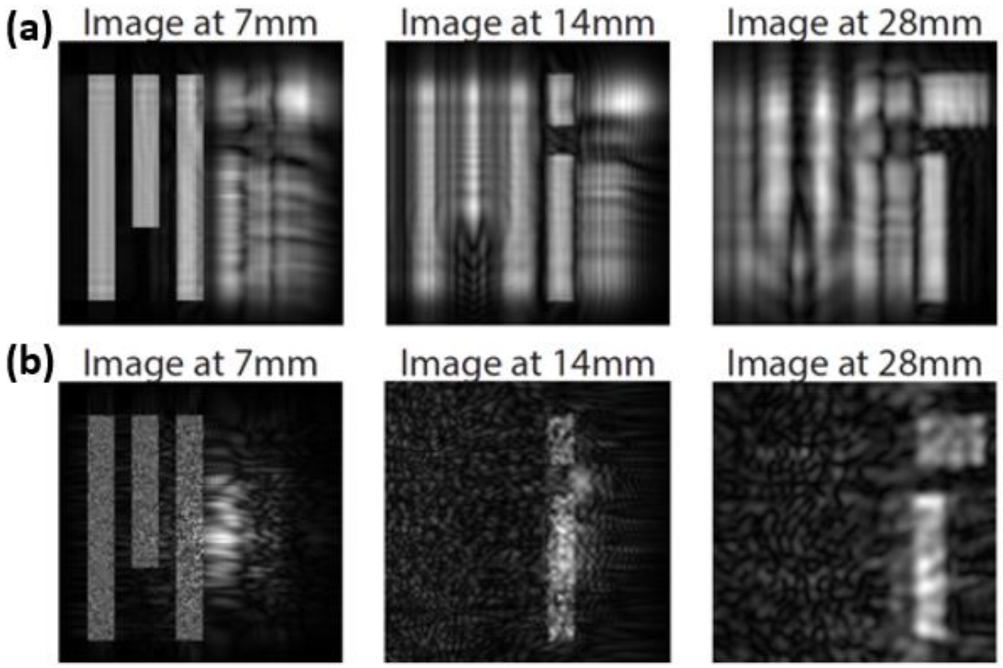

Fig. 4.4-1 (a) Initial simulation results. **(b)** Simulation results using proposed improvements.

4.5. Noise Analysis

So far we have shown a method that has the capability to design phased arrays with the required parameters to emit arbitrary 3D outputs. Our simulation examples have calculated these parameters for a 128x128 array equating to over 16,000 antennas that must each be tuned in both amplitude and phase. Due to complications like fabrication imperfections, the chances that the amplitude and phase profiles exactly match what was designed is small. That is why it is important for us to examine how resilient our method is to noise. As the noise will predominantly come from fabrication issues, we can model our noise as random Gaussian noise.

We first begin by examining the effects of phase noise on our method. In order to simulate the effects, we add a random Gaussian phase noise centered at zero to the MIT simulations and vary the standard deviation σ. Similar to how phase was added to our outputs, this random phase noise was added to the electric field representing the summation of the back propagated outputs

$$E_{PA}(x, y) = E_{summed}(x, y) \times e^{\sigma \times \mathrm{rand(x,y)}} \tag{4.5.1}$$

where $E_{PA}(x, y)$ is the electric field that the phased array will emit, $E_{summed}(x, y)$ is the summation of the back propagated outputs, and $(\sigma \times \mathrm{rand(x, y)})$ generates our Gaussian random noise distribution centered at zero with variable standard deviation σ.

In order to find a more quantitative comparison of the noise effects, we decided to calculate a Hermitian inner product between the outputs with noise and the original output without any noise. This calculates a percentage of projection, or overlap, compared to the original, no noise output. The value is calculated by evaluating

$$\frac{< E_{\sigma=\sigma_i}, E_{\sigma=0} >}{\sqrt{< E_{\sigma=\sigma_i}, E_{\sigma=\sigma_i} > * < E_{\sigma=0}, E_{\sigma=0} >}} \tag{4.5.2}$$

Where $E_{\sigma=0}$ is our output when designed with no noise and $E_{\sigma=\sigma_i}$ is the output when designed with Gaussian noise at varying standard deviations. Fig. 4.5-1 shows the plot of the overlap as the standard deviation of the noise is increased for the M output, the I output, and the T output. The range of standard deviations is chosen to be from 0 to π/2 to give us a range of additional phase offsets predominantly between – π and + π the desired phase difference between antennas. We recognize that the M output seems to degrade the fastest, so we decide to focus our analysis on that output in particular.

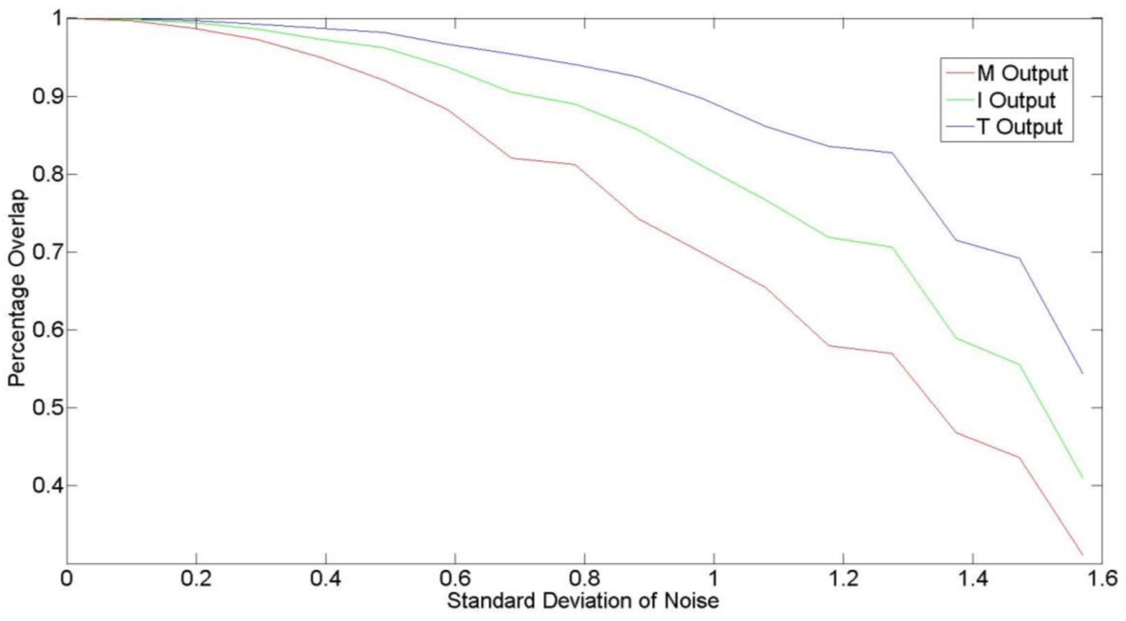

Fig. 4.5-1 Plot comparing overlap calculations of MIT outputs due to phase noise

By examining the percent overlap plot for the M output, we see that even at σ = π/4 the percentage of overlap compared to the original output is still approximately 85%. This is a strong indicator of the resiliency of the method because this σ = π/4 means there can be pairs of antennas that have a phase difference of π/2 more than desired, yet still produce an output that is very close to the output without noise.

To get a more quantitative view, Fig. 4.5-2 shows how the M output changes with noise at a) σ = 0 (no noise) b) σ = π/16 c) σ = π/8 and d) σ = π/4. In all of these outputs it is still clear that the output is supposed to resemble the "M" from the MIT logo, but obviously the output noise begins to rise as the standard deviation of the added noise increases.

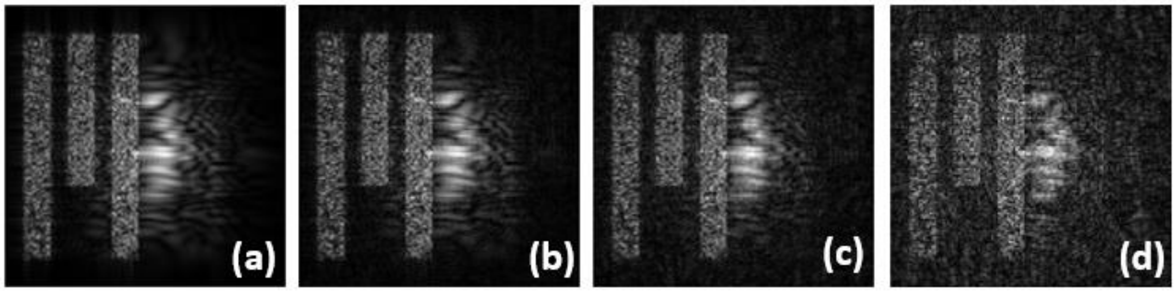

Fig. 4.5-2 (a) M output with no phase noise. **(b)** M output with Gaussian random noise of σ = π/16. **(c)** M output with Gaussian random noise of σ = π/8. **(d)** M output with Gaussian random noise of σ = π/4.

We will perform a similar set of simulations to evaluate the effects of noise on the amplitude. For the amplitude we decided to use a random Gaussian noise distribution centered around 1 so that the standard deviation represents a percentage change of the amplitude

$$E_{PA}(x,y) = E_{summed}(x,y) \times (1 + \sigma^{*}rand(x,y)) \tag{4.5.3}$$

Where $(1 + \sigma^{*}rand(x,y))$ represents our Gaussian random noise distribution centered at 1 with standard deviation σ. We calculate the same Hermitian inner product as 4.5.2 to see quantitatively how our outputs are affected by phase noise.

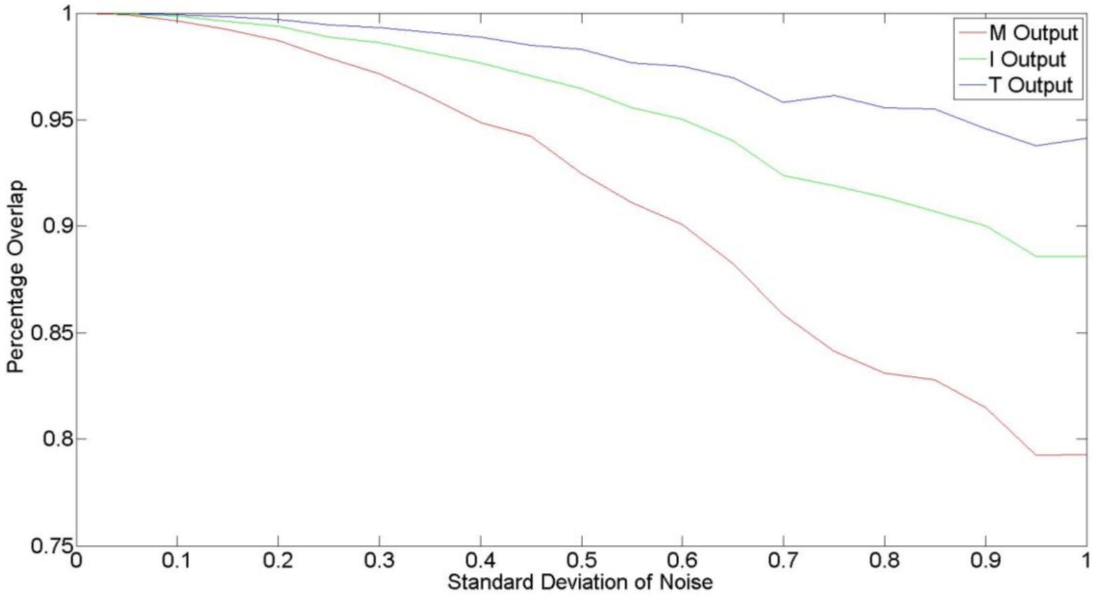

Fig. 4.5-3 Plot comparing overlap calculations of MIT outputs due to amplitude noise

Fig. 4.5-3 shows that the M output again degrades faster than the others due to amplitude noise and we will again turn our attention to that output. The range of standard deviations is chosen to be 0 to 1 to give us amplitude modifications predominantly between -100% and +100%, analogous to the full additional π phase shift used in the phase noise simulations. When we examine the percent overlap calculations of the M output for σ = 0.5, which corresponds to having antenna amplitudes predominantly modified within -50% to +50% of their intended amplitude, we see that there is still approximately 97.5% of an overlap with the original output. It is also important to note that the outputs maintain a higher level of overlap over this range of standard deviations than the phase noise simulations.

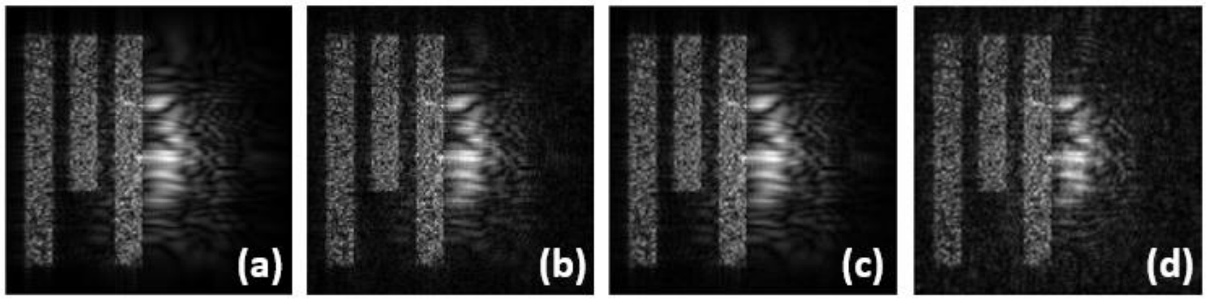

Fig. 4.5-4 (a) M output with no amplitude noise. **(b)** M output with Gaussian random noise of σ = 0.1. **(c)** M output with Gaussian random noise of σ = 0.25. **(d)** M output with Gaussian random noise of σ = 0.5.

The M outputs with varying standard deviation of the added Gaussian random amplitude noise is shown in Fig. 4.5-4 where a) σ = 0 (no noise) b) σ = 0.1 c) σ = 0.25 and d) σ = 0.5. These results strongly resemble the desired M output over these σ and look to be even more resilient to noise compared to the phase noise simulations, which agrees with the overall higher level of overlap we see in Fig. 4.5-3 for the amplitude noise simulations compared to the levels of overlap for the phase noise analysis in Fig. 4.5-1.

4.6. Amplitude and Phase Control via Couplers

Now that we have determined a method to generate phase and amplitude profiles that is resilient to noise and improves the output in comparison to the initial attempt, we will cover a method to allow our phased array to emit this electric field. There are two primary methods for our antennas to emit our desired amplitude and phase profiles. One way to accomplish this is an active method that uses heaters within the path of the unit cell to tune these parameters. The technique that will be described in this section is a passive method that uses a system of directional couplers to achieve the desired parameters.

Directional couplers are devices that redirect a portion of electromagnetic power from a waveguide to another output. Here, we design our couplers by having two waveguides set close enough to allow for the passage of energy from one to the other. The directionality comes from the fact that power is only coupled from the main waveguide to the secondary waveguide and none is coupled the other way around. The amount of power that is coupled into the secondary waveguide is controlled by the gap between the two waveguides and the length L_c of the secondary waveguide as seen in Fig. 4.6-1. This figure shows the coupler attached to the unit cell with our antenna. Along with controlling how much power is transferred, a combination of coupler gap and length also provides a corresponding phase shift. In order to achieve the desired phase on each antenna, then, we have to modify the straight waveguides within our unit cell to make up the difference between the desired antenna phase and the phase shift due to the coupler. We modify the two straight waveguides in the unit cell together so that each component is modified to achieve half of the required offset phase shift or equivalently $\frac{\theta_{offset}}{2}$.

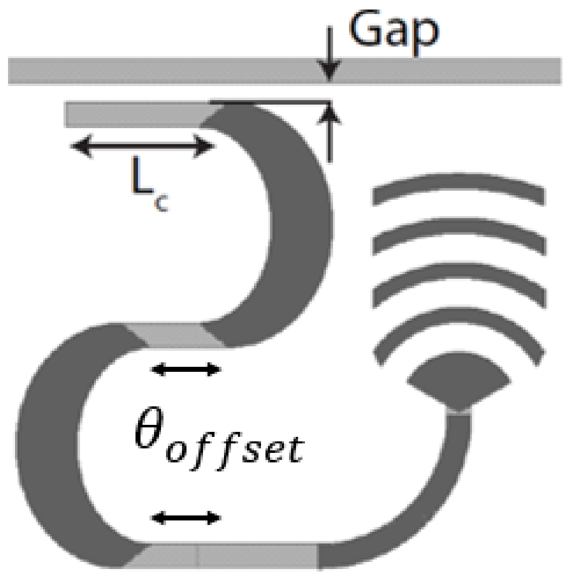

Fig. 4.6-1 Coupler and unit cell parameters

In order to determine a wide dynamic range of coupling coefficients, the proportion of power coupled by the coupler, we used 3D Finite Difference Time Domain (FDTD) simulations of our coupler with varying combinations of coupler gap and length. By placing power and phase monitors at the locations seen in Fig. 4.6-2, we measure power and phase entering the main waveguide and the same values exiting the coupler and entering the unit cell.

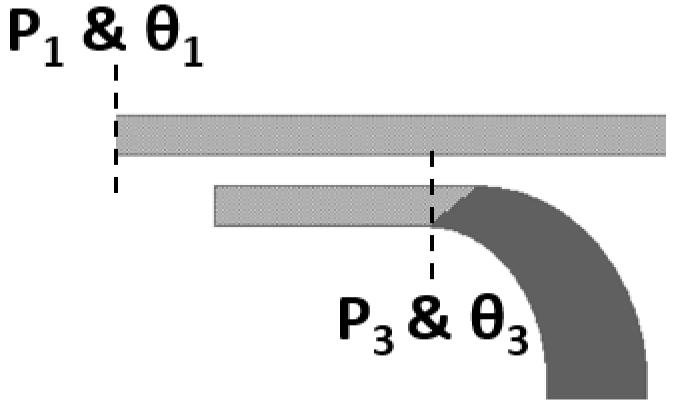

Fig. 4.6-2 Location of power and phase monitors

We calculate power coupling coefficients by taking the proportion of power exiting the coupler over the power entering the main waveguide

$$C = \frac{P_3}{P_1} \tag{4.6.1}$$

Similarly, we also calculate the corresponding phase shift due to the coupler by calculating how much the input wave phase changes from the entrance of the main waveguide to the exit of the coupler. We determined that we can achieve our desired power coupling coefficients by utilizing 3 varying gap distances and a range of coupling lengths between 0.5 μm and 5 μm, ultimately giving us a range of 0.011% to 82.5% of power coupled from one waveguide to the other with no gaps in the range. The increase of elements in our phased array requires this much larger dynamic range of coupling coefficients compared to what was previously used. Fig. 4.6-3 shows the two plots of power coupling coefficients and corresponding phase changes due to these combinations of coupler gaps and lengths. The power coupling coefficients match up with what we expect as a larger gap, or distance between coupler and waveguide, equates to less power being able to be transferred between the two. Similarly, a longer coupler length allows for more power to be coupled as there is more area for the signal to be coupled into the coupler.

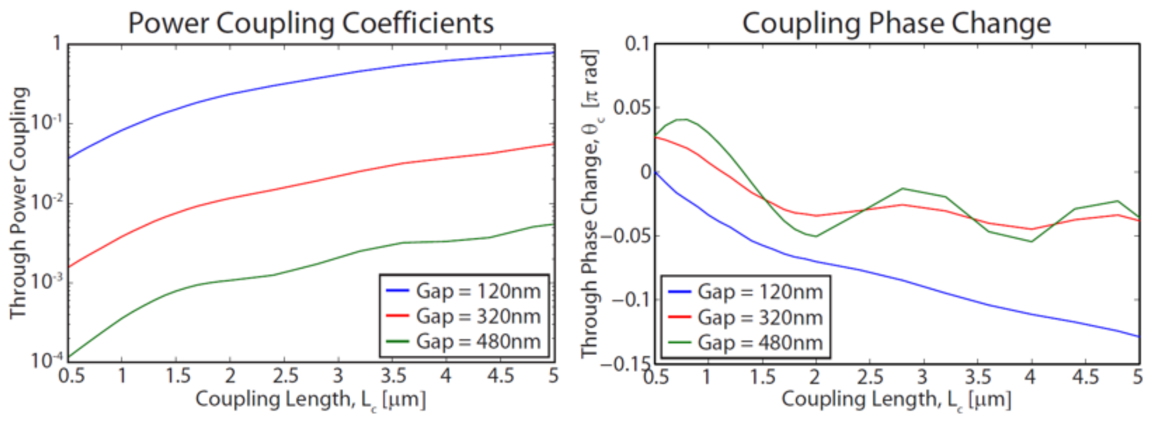

Fig. 4.6-3 Plots of power coupling coefficients and phase shift due to coupler gaps and lengths

The power will be distributed by a system of row couplers and unit couplers as shown in Fig. 4.6-4 in order to achieve the desired antenna amplitudes. The main source of power will come from a laser coupled into our main waveguide on the far left of Fig. 4.6-4 and will be distributed into each row waveguide with the row couplers. Then, the power will finally reach the unit cells through the unit couplers.

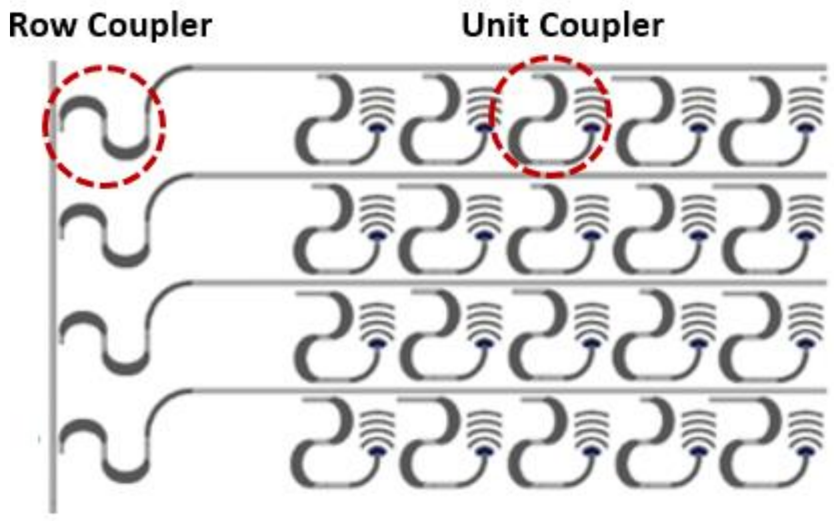

Fig. 4.6-4 System of row and unit couplers

We calculate the required coupling coefficient of each coupler by solving a linear system of equations using our amplitude profile. First, we square the amplitude profile to give us a power distribution. To calculate the required coefficients for the row couplers C_i, we find the total power in each row P_i and total power in the entire system P_{total} and then the required row coupling coefficients are a proportion of the total row power over the remaining power

$$C_i = \frac{P_i}{\prod_{n=1}^{i-1}(1 - C_n)P_{total}} \tag{4.6.2}$$

The required coefficients for the unit couplers $C_{i,k}$ can then be calculated by using the total power for each unit cell $P_{i,k}$ and is the required unit cell power over the remaining power in the row

$$C_{i,k} = \frac{P_{i,k}}{\prod_{n=1}^{k-1}(1 - C_{i,n})P_i} \tag{4.6.3}$$

Once the required coefficient for the row and unit couplers is calculated, we determine the combination of coupler gap and length using the coefficient values for all of the couplers by using our 3D FDTD simulation results.

We then reference these same simulation results and use the selected coupler gaps and lengths to find the corresponding phase shift due to the coupler. We change the length of the straight waveguides in the unit cell to make up the difference between the desired emitted phase and the phase shift due to the coupler

$$\theta_{desired} = \theta_{coupler} + \theta_{offset} \tag{4.6.4}$$

This method utilizing couplers highlights a passive way to cause our antennas to emit the desired amplitude and phase profile that our method calculates.

4.7. Layout

In order to design and fabricate our devices, we must first layout a mask that can be used during lithographic processes to generate the desired structures on a silicon wafer. The layout outlines how the device will look when fabricated on a chip and contains the information of the locations and dimensions of the various components of the system. Layout for electronic components is traditionally done using Cadence, and since we are utilizing a CMOS fabrication process, it can also be used to layout silicon photonic devices.

Smaller devices can be manually laid out by drawing and carefully designing structures with the proper dimensions. However, with a device that contains over 16,000 individual antennas and couplers with their own required dimensions (such as a 128x128 antenna phased array), this process can become very tedious and error-prone to do manually. Fortunately, there is a way to automate the process of drawing and placing the various shapes through scripts written using SKILL code, the Cadence scripting language. Additionally, the drawing of individual components like an antenna can be scripted in SKILL with variable parameters so that they can be called by a larger wrapper script to automate the placement and sizing of a large number of structures.

We wrote a SKILL script that reads in the required coupler and antenna parameters calculated using our algorithm. The code then places waveguides to set up the grid structure of the phased array. It will read in the coupler length and gap requirements for both row and unit couplers and call up our code that outlines a coupler with variable gap and length to place and appropriately size the couplers. Then it will place the antennas and tune the straight waveguide section to account for the coupler phase shift, as previously calculated. The resulting layout when using the calculated parameters to generate a phased array of 128x128 antennas that emits the MIT logo in the near field at 1550 nm wavelength is partially shown in Fig. 4.7-1. The inset shows the parameters that must be properly sized for each antenna and coupler pair.

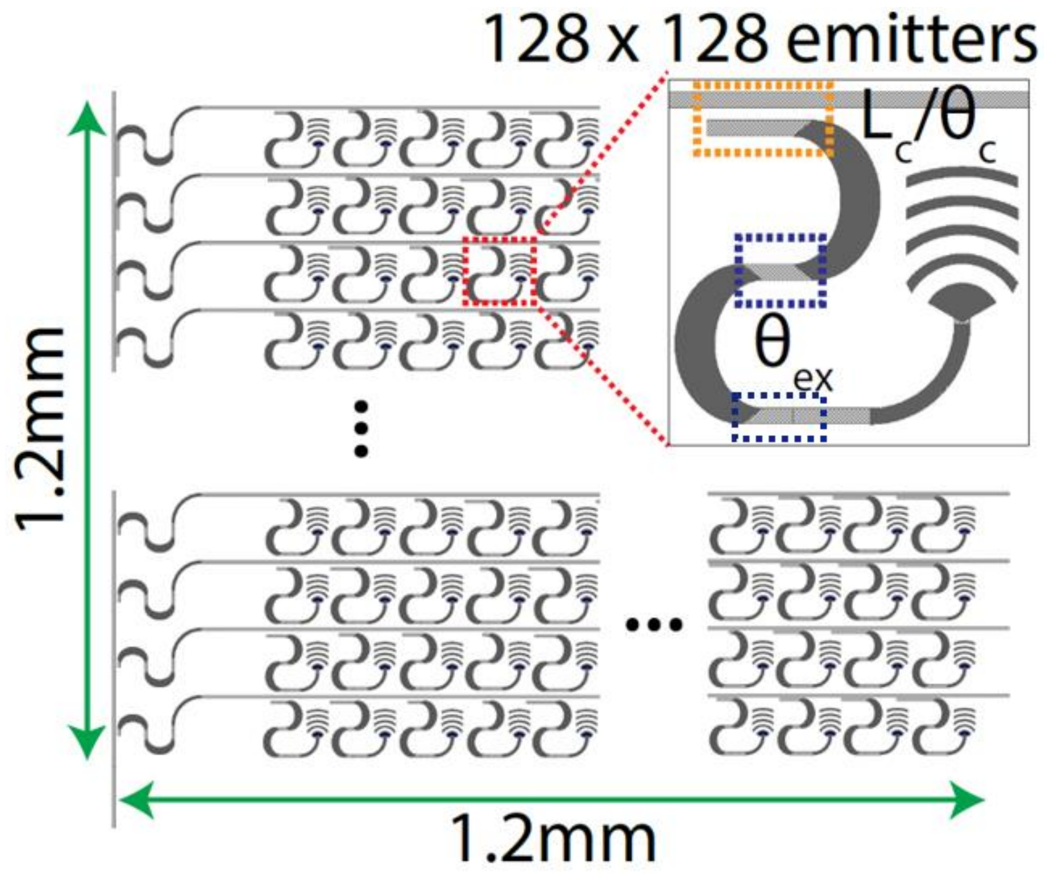

Fig. 4.7-1 Layout of phased array that emits MIT in near field

4.8. Additional Simulations

To test the flexibility of our method to produce arbitrary outputs, we used the algorithm to design a phased array to emit a 3D pyramid as shown in Fig. 4.8-1. The output planes consist of a square of single pixel wide lines that grows smaller and smaller in side length as the planes move further from the phased array until reaching a 2x2 square of pixels at the furthest point, emulating the steps of a pyramid.

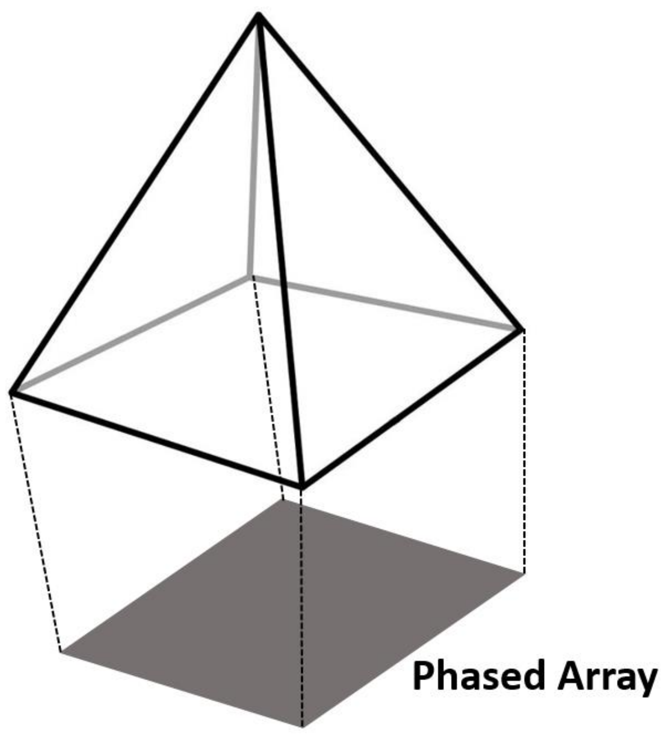

Fig. 4.8-1 Phased array emitted 3D pyramid

This test uses 28 planes equally spaced at 5 times the pitch of the array ($5 \times 9\ \mu m$) to make the pyramid. We follow the algorithm and back propagate all 28 planes and sum them to generate the electric field for the phased array to emit. Then, by forward propagating and viewing the individual output planes, we see how well the algorithm recreates our desired pyramid. We realize that because of the close proximity and large number of the output planes, it is challenging to isolate each output plane and therefore distinguish which part is in focus. To try to better determine the results of this test, we added randomized phase to the pyramid and removed the corners of the squares when building the output to give us the triangular faces of the pyramid with an empty line along the diagonals. Then, to read the outputs, we focus in at the various levels of the pyramid to see which portions are in focus as depicted in Fig. 4.8-2.

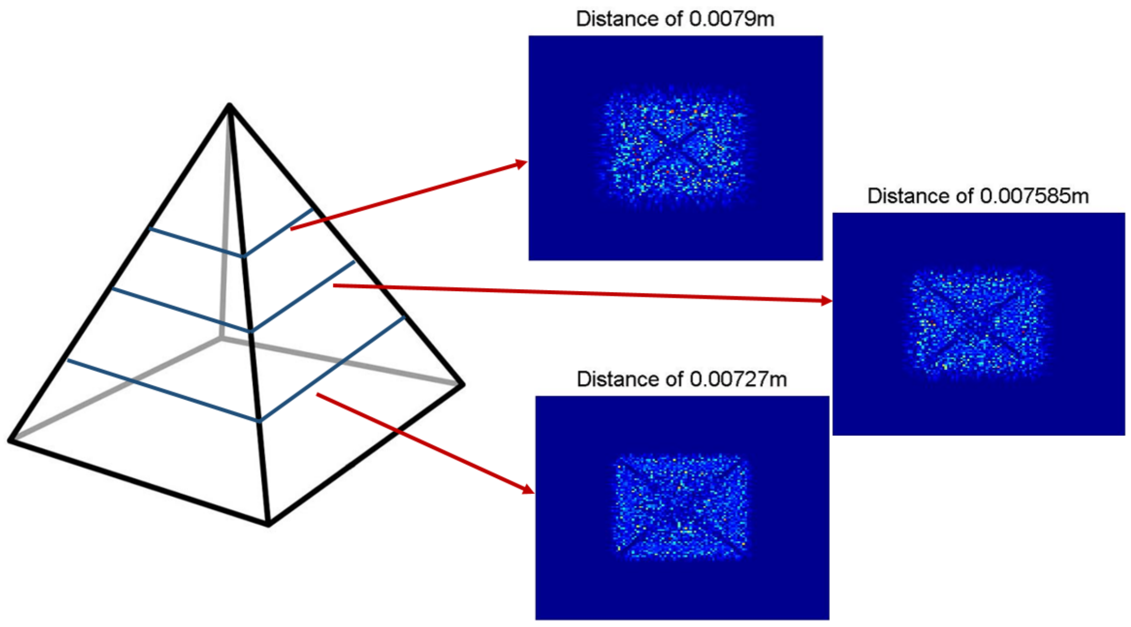

Fig. 4.8-2 Pyramid output slices

The randomized phase here allows the planes to disperse, but because of the close proximity of the outputs, the light will not have much room to fully disperse and their effects will still be visible in the other output planes. The empty pixels between the triangular faces allow us to determine which portion of the pyramid are in focus at any distance as the sharpness of these empty regions will tell us if the section is in focus or not. Fig. 4.8-3 provides a few slices of the output pyramid with the in-focus empty regions circled. We can clearly see in each image that there is a section of the empty spaces in focus, while the other parts are blurred. The top left image corresponds to the base of the pyramid and we can clearly see that the corners are where the empty spaces are in focus. Furthermore, the edges of the base of the pyramid are sharp, which also tells us that the base is in focus. As we move farther from the phased array, we can see that the areas of focus begin shifting towards the center, or equivalently towards the top of the pyramid, as we would expect based on how the pyramid was designed. The edge of the base also becomes less and less sharp corresponding to it becoming more and more out of focus as we move up the pyramid. Again, as this is an output that we are looking at with multiple planes at different depths, what these outputs show makes sense because the other outputs should be blurred and out of focus rather than completely disappearing. These results show that this algorithm is capable of generating phased arrays that can emit varying output projections. This capability not only shows the flexibility of this method, but also proves useful in creating dynamic displays.

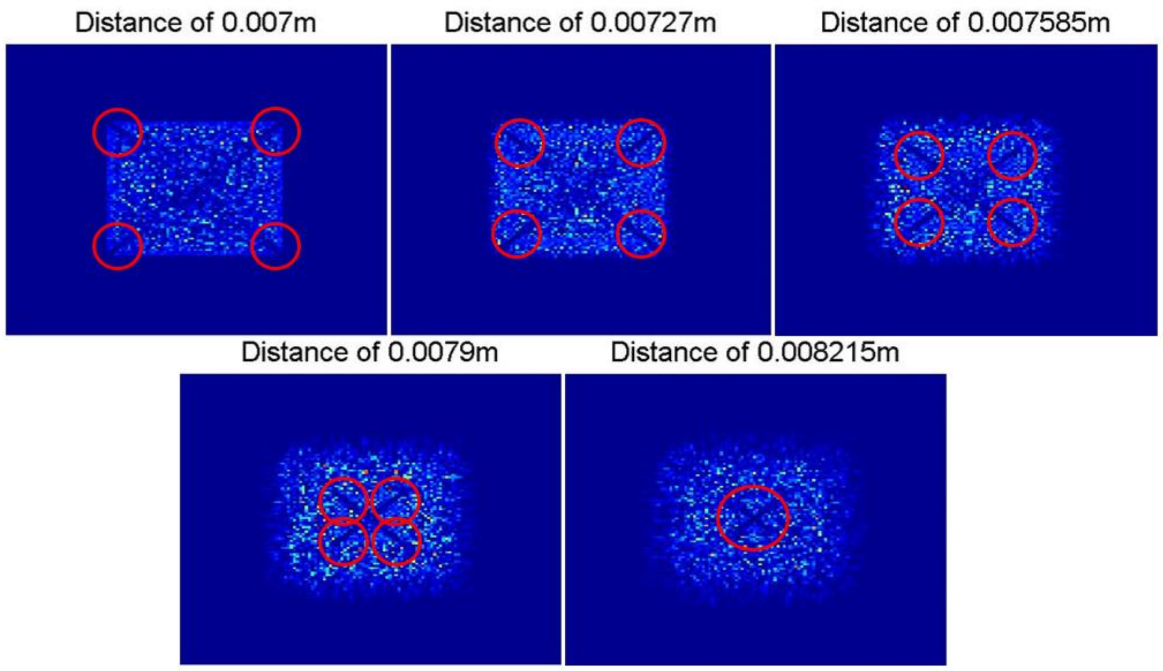

Fig. 4.8-3 Output planes moving away from phased array with focused empty spaces circled

Additionally, we wanted to prove that this algorithm is capable of designing phased arrays for arbitrary wavelengths. To show this, we tested the algorithm to see how the MIT logo example would look when we used 750 nm, a wavelength in the visible spectrum corresponding to the color red. We simply changed our MATLAB code to generate the phase and amplitude profiles using the 750 nm wavelength and the pitch to be a multiple of half of this wavelength. The results we get when we forward propagate these calculated fields is shown in Fig. 4.8-4.

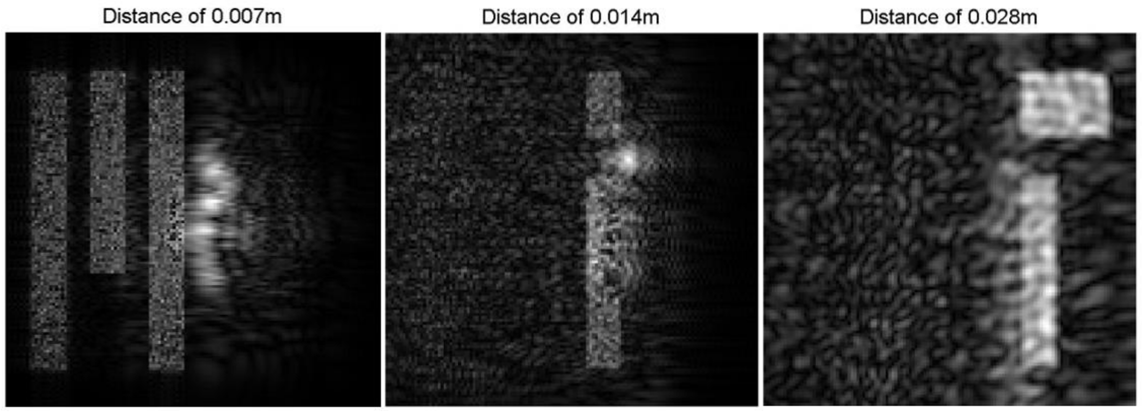

Fig. 4.8-4 MIT logo output using 750 nm wavelength

We see that this output resembles the output generated using a wavelength of 1550 nm. We can clearly see the M, the I, and the T at the individual designed distances from the phased array. This is a simple test without any significant optimization done at this wavelength and we can see that it can emit a desired 3D pattern at varying wavelengths. This flexibility is extremely useful, especially when we consider visible applications that can utilize 3D holograms, such as the 3D TVs previously mentioned.

Chapter 5

Conclusion

In conclusion, this thesis examined the possibility of combining silicon photonics and holographic technologies to open up exciting applications. To do this, we proposed that we could design phased arrays with the capability to emit 3D holograms by working in the near field. To properly design a phased array with this emission, we devised a method that required superimposing back propagated output planes to generate the required phase and amplitude profiles for the antennas to emit. These profiles can then be extracted and discretized to tune the parameters used to design the individual antennas and couplers. A forward emission of the field emitted by the antennas would then recreate the desired output.

We tested our methods and found it to work well in terms of how humans perceive 3D objects. However, to open up our design to more applications, we designed a few improvements to try to isolate our output images and remove output size limitations. This was accomplished by realizing that the outputs were primarily focused on the intensity, meaning that we could modify the phases. By adding randomized phase to the output planes, we were able to decollimate the output beams and allow pixels to disperse faster as we varied distance. Adding circular phase to the output planes allowed us to better focus the information during back propagation to eliminate losses. It also proved to be useful as it allowed us the freedom to generate outputs larger than the size of the phase array itself. These methods were individually tested and proven to accomplish the desired improvements to our algorithm. Comparing it to our original results, it is clear that these improvements work to better isolate our outputs.

The next part of the work details how exactly the phase and amplitude is controlled in our phased array design. By using a system of couplers, we can control how much power each antenna receives and thus control the amplitude. Each coupler, however, also brings about a phase delay to the signal. This can be accounted for by modifying the length in the unit cell waveguide to add or remove the required phase delays to the antennas to achieve the desired phase profile.

Finally, we show the final design that combines all of the work to construct a phased array that emits our MIT output using Cadence. It consists of a 128x128 antenna phased array with 9 µm antenna spacing that is designed to emit the near field pattern when a 1550 nm wavelength laser is used as an input. We also further prove the flexibility of our algorithm by showing that it is capable of not only generating arbitrary outputs, but is also capable of designing phased arrays that operate at arbitrary wavelength, such as within the visible spectrum.

Once the devices have been successfully fabricated, we still have to test the devices to ensure that they function desirably. Assuming this part is successful, the next steps would be to first test the capability to generate varying outputs along with outputs at differing wavelengths. This part shouldn't be too tricky as we have already shown that the algorithm has the capability to design phased arrays with these capabilities. The more interesting step would be to test the possibilities of incorporating phase shifters into a phased array designed to emit 3D holograms. By varying the phase, we can test to see the potential of having a phased array emitting a dynamic output. With electronic signaling, this could be controlled and changed quickly and could possibly lead to moving 3D holograms. This capability is critical to 3D display technology as well as a multitude of other applications.

Bibliography

[1] M. Lipson, "Guiding, Modulating, and Emitting Light on Silicon – Challenges and Opportunities,".Journal of Lightwave Technology **23** (12) 4222–4238 (2005).

[2] T. Barwicz, et al., "Silicon photonics for compact, energy-efficient interconnects," Journal of Optical Networking **6** (1) 63–73 (2006).

[3] E. Timurdogan, et al., "A High-Q Tunable Interior-Ridge Microring Filter," in *CLEO: 2014*, OSA Technical Digest (online) (Optical Society of America, 2014), paper SF2O.3 (2014).

[4] M. R. Watts, and H. A. Haus, "Integrated Mode-Evolution-Based Polarization Rotators," Optics Letters, **30,** 138 (2005).

[5] M. R. Watts, H. A. Haus, and E. P. Ippen, "Integrated Mode-Evolution-Based Polarization Splitter," Optics Letters, **30,** 967 (2005).

[6] T. Barwicz, et al., "Polarization Transparent Microphotonic Devices in the Strong Confinement Limit," Nature Photonics, **1,** 57 (2006).

[7] M. R. Watts, et al., "Low Voltage, Compact, Depletion-Mode, Silicon Mach-Zehnder Modulator," Journal of Special Topics in Quantum Electronics, IEEE J. Sel. Top. Quantum Electron., **16,** 159 (2010).

[8] M. R. Watts, et al., "Vertical Junction Silicon Microdisk Modulators and Switches," Optics Express, **19,** 21989 (2011).

[9] E. Timurdogan, et al., "An ultralow power athermal silicon modulator," Nature Communications, **5,** 1 (2014).

[10] M. R. Watts, "Adiabatic Microring-Resonators," Optics Letters, **25,** 3231 (2010).

[11] Z. Su, et al., "A silicon wavelength selective partial-drop broadcast filter bank," Opt. Lett. **39,** 5459 (2014).

[12] D. Cole, et al., "Integrated Heterodyne Interferometer with On-Chip Modulators and Detectors", Opt. Lett. **40**, 3097 (2015).

[13] R. C. Hansen, *Phased array antennas*, **213**, John Wiley & Sons, 2009.

[14] K. F. Braun, "Electrical oscillations and wireless telegraphy," Nobel Lecture (1909).

[15] C. Alakija, and S. P. Stapleton, "A mobile base station phased array antenna," 1992 IEEE International Conference on Selected Topics in Wireless Communications, 118-121 (1992).

[16] Y. Kawaguchi, et al., "Application of phased-array antenna technology to the 21 GHz broadcasting satellite for rain-attenuation compensation," 2002 IEEE International Conference on Communications, **5**, 2962-2966 (2002).

[17] E. Brookner, "Phased-array radars," Scientific American **252** (2), 94-102 (1985).

[18] G. van Keuk, and S. S. Blackman, "On phased-array radar tracking and parameter control," IEEE Transactions on Aerospace Electronic Systems **29**, 186-194 (1993).

[19] A. J. Fenn, et al., "The development of phased-array radar technology." Lincoln Laboratory Journal **12** (2), 321-340 (2000).

[20] R. Klemm, "Adaptive clutter suppression for airborne phased array radars," IEE Proceedings F (Communications, Radar and Signal Processing), **130** (1), 125-132 (1983).

[21] D. S. Zrnic, et al., "Agile-beam phased array radar for weather observations,"Bulletin of the American Meteorological Society **88** (11), 1753-1766 (2007).

[22] P. F. McManamon, et al., "Optical phased array technology," Proceedings of the IEEE **84** (2), 268-298 (1996).

[23] R. J. Crowley, "Optical antenna array for harmonic generation, mixing and signal amplification," U.S. Patent No. 6,038,060, (14 Mar. 2000).

[24] D. P. Resler, et al., "High-efficiency liquid-crystal optical phased-array beam steering," Optics letters **21** (9), 689-691 (1996).

[25] K. Van Acoleyen, et al., "Off-chip beam steering with a one-dimensional optical phased array on silicon-on-insulator," Optics letters **34** (9), 1477-1479 (2009).

[26] K. Van Acoleyen, H. Rogier, and R. Baets, "Two-dimensional optical phased array antenna on silicon-on-insulator," Optics express **18** (13), 13655-13660 (2010).

[27] J. Sun, et al., "Large-scale nanophotonic phased array," Nature **493** (7431), 195-199 (2013).

[28] A. Yaacobi, E. Timurdogan, and M. R. Watts, "Vertical emitting aperture nanoantennas," Optics letters **37** (9), 1454-1456 (2012).

[29] R. Collier, *Optical holography*, Elsevier, 2013.

[30] D. Gabor, "A new microscopic principle," Nature **161** (4098), 777-778 (1948).

[31] D. Gabor, "Microscopy by reconstructed wave-fronts," Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, **197** (1051), 454-487 (1949).

[32] "The Nobel Prize in Physics 1971". Nobelprize.org. Nobel Media AB 2014. Web. 14 Jul 2015. http://www.nobelprize.org/nobel_prizes/physics/laureates/1971/

[33] E. N. Leith and J. Upatnieks, "Reconstructed wavefronts and communication theory," JOSA **52** (10), 1123-1128 (1962).

[34] T. Kreis, *Handbook of holographic interferometry: optical and digital methods*, John Wiley & Sons, 2006.

[35] P. Giancarlo, W. Osten, and M. E. Gusev, "High-speed digital holographic interferometry for vibration measurement," Appl. Opt. **45**, 3456-3462 (2006).

[36] M. Hÿtch, et al., "Nanoscale holographic interferometry for strain measurements in electronic devices," Nature **453** (7198), 1086-1089 (2008).

[37] C. P. Wood and J. D. Trolinger, "Application of real-time holographic interferometry in the nondestructive inspection of electronic parts and assemblies," Proc. SPIE **1332**, 122-131 (1991).

[38] D. D. Robertson, et al., "Depiction of pelvic fractures using 3D volumetric holography: comparison of plain X-ray and CT," Journal of computer assisted tomography **19** (6), 967-974 (1995).

[39] J. Brook and R. Dandliker, ‘‘Submicrometer Holographic Photolithography,’’ Solid State Technology **33**, 91-94 (November 1989).

[40] J. M. Lerner, et al., "Diffraction gratings ruled and holographic-a review," Proc. SPIE **0240**, 82-88 (1981).

[41] G. T. Sincerbox, "Holographic scanners: applications, performance and design," Laser Beam Scanning **8**, 1-62 (1985).

[42] S. Tay, et al., "An updatable holographic three-dimensional display," Nature **451** (7179), 694-698 (2008).

[43] T. Kreis, "Applications of digital holography: from microscopy to 3D-television," Journal of the European Optical Society-Rapid publications **7**, (2012).

[44] W. Matusik and P. Hanspeter, "3D TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes," ACM Transactions on Graphics (TOG) **23** (3), 814-824 (2004).

[45] B. R. Brown and A. W. Lohmann, "Computer-generated binary holograms,"IBM Journal of research and Development **13** (2), 160-168 (1969).

[46] G. Roelkens, D. Van Thourhout, and R. Baets, "High efficiency Silicon-on-Insulator grating coupler based on a poly-Silicon overlay," Optics Express **14** (24), 11622-11630 (2006).

[47] J. W. Goodman, *Introduction to Fourier optics*, Roberts and Company Publishers, 2005.

[48] C. A. Balanis, *Antenna theory: analysis and design*, **1**, John Wiley & Sons, 2005.

[49] G. D. Gillen and S. Guha, "Modeling and propagation of near-field diffraction patterns: a more complete approach," American journal of physics **72** (9), 1195-1201 (2004).

Appendix A

MATLAB Code

fresnel_advance.m

This function performs the Fresnel diffraction propagation using Fast Fourier Transform

```
U = ifft2(ifftshift(O.*H));
function U = fresnel_advance(U0, dx, dy, z, lambda)
% This function receives a field U0 at wavelength lambda
% and returns the field U after distance z, using the Fresnel
% approximation. dx, dy, are spatial resolution.

% Calculate k-vector and input dimensions
k=2*pi/lambda;
[ny, nx] = size(U0);

% Generate spatial dimensions u,v
Lx = dx * nx;
Ly = dy * ny;

dfx = 1./Lx;
dfy = 1./Ly;

u = ones(nx,1)*((1:nx)-nx/2)*dfx;
v = ((1:ny)-ny/2)'*ones(1,ny)*dfy;

% Calculate FFT of input
O = fftshift(fft2(U0));

```

```
% Generate FFT of Fresnel Diffraction impulse response H
H = exp(1i*k*z).*exp(-1i*pi*lambda*z*(u.^2+v.^2));

% Propagated field is inverse FFT of the multiplication of the
  FFT of the input and the Fresnel Diffraction H
U = ifft2(ifftshift(O.*H));
```

Fresnel128PA_MIT.m

This is the wrapper code used to generate the antenna phase and amplitude profiles to emit the MIT logo, as well as simulating and examining the output

```
% Variable that adds circular and random phase when set to 1
nonflat = 1;

% Wavelength and Antenna spacing variables
lambda = 1.55e-6;
pitch = 9e-6/factor;

% Number of antennas in the x and y direction
N=128;
M=128;

% Output distances
z1 = 0.007;
z2 = 0.014
z3 = 0.028;

% Generate grid matrices of x and y positions
x1 = (-M/2+1)*pitch:pitch:pitch*M/2;
y1 = (-N/2+1)*pitch:pitch:pitch*N/2;
[x1, y1] = meshgrid(x1,y1);

% Generate desired MIT logo for output
I = imread('MIT_logo', 'jpeg');
I = imresize(I,[N M]);
I = double(256*im2bw(I,0.8));
I(36,89)=256;

% Segmenting logo to separate letters
% Isolate M
```

```
I1 = imcrop(I, [0 0 65 128]);
I1 = padarray(I1, [0 128-65], 'post');
I1 = imresize(I1, [N M],'box');

% Add a curved and random phase if nonflat variable is set to 1
if nonflat == 1
    I1 = 1.25*I1.*exp(j*(2*pi*z1/lambda)*(-(-
          sqrt((x1.^2+y1.^2)/z1^2+1)+1)+N^(0.9)*pitch^2/z1^2*rand
          (N,M)));
end

% Isolate I
I2 = imcrop(I, [65 0 20 128]);
I2 = padarray(I2, [0 64],'pre');
I2 = padarray(I2, [0 128-85],'post');
I2 = imresize(I2, [N M],'box');

% Add a curved and random phase if nonflat variable is set to 1
if nonflat == 1
    I2 = 1.85*I2.*exp(j*(2*pi*z2/lambda)*(-(-
          sqrt((x1.^2+y1.^2)/z2^2+1)+1)+1.2*N^(0.9)*pitch^2/z2^2*
          rand(N,M)));
end

% Isolate T
I3 = imcrop(I, [85 0 128-85 128]);
I3 = padarray(I3, [0 84],'pre');
I3 = imresize(I3, [N M],'box');

% Add a curved and random phase if nonflat variable is set to 1
if nonflat == 1
    I3 = I3.*exp(j*(2*pi*z3/lambda)*(-(-
          sqrt((x1.^2+y1.^2)/z3^2+1)+1)+1.6*N^(0.9)*pitch^2/z3^2*
          rand(N,M)));
end
```

```

% Pad outputs
paddingFactor = 5; % Increase this to add more padding
I1 = padarray(I1,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);
I2 = padarray(I2,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);
I3 = padarray(I3,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);

% Back Propagation of M
temp1 = fresnel_advance(I1,pitch,pitch,-z1,lambda);
temp1 = padarray(temp1(((M*paddingFactor-
         M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
         N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-
         M)/2 (N*paddingFactor-N)/2]);
% Sanity check, view forward propagated field
Out1 = fresnel_advance(temp1,pitch,pitch,z1,lambda);
Out1 = Out1/max(abs(Out1(:)));
figure
imshow(abs(Out1(((M*paddingFactor-M)/2+1):((M*paddingFactor-
         M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
         N)/2+N))));

% Back Propagation of I
temp2 = fresnel_advance(I2,pitch,pitch,-z2,lambda);
temp2 = padarray(temp2(((M*paddingFactor-
         M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
         N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-
         M)/2 (N*paddingFactor-N)/2]);
% Sanity check, view forward propagated field
Out2 = fresnel_advance(temp2,pitch,pitch,z2,lambda);
Out2 = Out2/max(abs(Out2(:)));
figure
imshow(abs(Out2(((M*paddingFactor-M)/2+1):((M*paddingFactor-
         M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
         N)/2+N))));

```

```
% Back Propagation of T
temp3 = fresnel_advance(I3,pitch,pitch,-z3,lambda);
temp3 = padarray(temp3(((M*paddingFactor-
         M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
         N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-
         M)/2 (N*paddingFactor-N)/2]);
% Sanity check, view forward propagated field
Out3 = fresnel_advance(temp3,pitch,pitch,z3,lambda);
Out3 = Out3/max(abs(Out3(:)));
figure
imshow(abs(Out3(((M*paddingFactor-M)/2+1):((M*paddingFactor-
         M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
         N)/2+N))));

% Superimpose backpropagated fields
tempFinal = temp1+temp2+temp3;

% Forward propagate summed field to first distance to see
         emitted output
OutF1 = fresnel_advance(tempFinal,pitch,pitch,z1,lambda);
OutF1 = OutF1/max(abs(OutF1(:)));
figure
imshow(abs(OutF1(((M*paddingFactor-M)/2+1):((M*paddingFactor-
         M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
         N)/2+N))));
title(['Distance of ',num2str(z1),'m'],'FontSize',16)

% Forward propagate summed field to second distance to see
         emitted output
OutF2 = fresnel_advance(tempFinal,pitch,pitch,z2,lambda);
OutF2 = OutF2/max(abs(OutF2(:)));
figure
imshow(abs(OutF2(((M*paddingFactor-M)/2+1):((M*paddingFactor-
         M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
         N)/2+N))));
```

```
title(['Distance of ',num2str(z2),'m'],'FontSize',16)

% Forward propagate summed field to third distance to see
          emitted output
OutF3 = fresnel_advance(tempFinal,pitch,pitch,z3,lambda);
OutF3 = OutF3/max(abs(OutF3(:)));
figure
imshow(abs(OutF3(((M*paddingFactor-M)/2+1):((M*paddingFactor-
          M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
          N)/2+N))));
title(['Distance of ',num2str(z3),'m'],'FontSize',16)


% Save summed electric field to extract amplitude and phase
          profiles
save('PA128_MITc_hires.mat','tempFinal');
```

RandPhase.m

This is the wrapper code used to test the effects of randomized phase. Two circles are generated at different distances and one is given randomized phase. They are backpropagated and summed to follow our algorithm. Then we examine the output at the circle with flat phase to see how randomized phase affects the second circle.

```
% This code tests the effects of randomized phase added to
        outputs

% Variable to show subplots
showsub = 1; %set to 1 to show spreading of circles
% Variable to test if we are varying pitch
varypitch = 0; %set to 1 to test pitch

% Wavelength and antenna spacing
lambda = 1.55e-6;
pitch = 9e-6;

% Changing pitch/phased array size
if varypitch == 1
%     pitch = 5e-6:1e-6:15e-6;
%     pitch = 5e-6;
    Nsize=96:16:224;
end

% Output distances
z1 = .007;
z2 = 0.014:0.002:0.02;

% Randomization levels
sigtest=0:0.000005:0.0001;
sigtest=sigtest/50;

% Change randomization level if we are varying pitch
if varypitch == 1
```

```
    sigtest = 0.00005;
end
% Holds values of our spread calculation
spreadCalc=zeros(length(sigtest),length(z2));
if varypitch == 1
    spreadCalc = zeros(length(z2),length(pitch));
end


% Vary over pitch values if we are varying it
for p = 1:length(pitch)
    index = 100;

    % array dimensions
    N=128;
    M=128;

    % Generate grid matrices of x and y positions
    x1 = (-M/2+1)*pitch(p):pitch(p):pitch(p)*M/2;
    y1 = (-N/2+1)*pitch(p):pitch(p):pitch(p)*N/2;
    [x1 y1] = meshgrid(x1,y1);

    % Testing with circles
    [columnsInImage rowsInImage] = meshgrid(1:N, 1:M);
    centerX1 = M/4;
    centerX2 = 3*M/4;
    centerY = N/2;

    radius = M/8;
    % First circle
    circlePixels1 = (rowsInImage - centerY).^2 ...
        + (columnsInImage - centerX1).^2 <= radius.^2;

    I1 = double(circlePixels1);

    % Second circle
    circlePixels2 = (rowsInImage - centerY).^2 ...
        + (columnsInImage - centerX2).^2 <= radius.^2;

    cPixels2 = double(circlePixels2);

    % Padding
    paddingFactor = 5; % Increase this to add more padding
    I1 = padarray(I1,[(M*paddingFactor-M)/2 (N*paddingFactor-
         N)/2]);
    % Generate grid matrices of x and y positions
    x2 = [-
         (M*paddingFactor)/2+1:1:(M*paddingFactor)/2]*pitch(p);
    y2 = [-
         (N*paddingFactor)/2+1:1:(N*paddingFactor)/2]*pitch(p);
    [x2, y2] = meshgrid(x2,y2);

    % Vary over our range of output distances
    for l=1:length(z2)
        disp(['Propagating ' num2str(l) ' of '
         num2str(length(z2)) ' values'])

        % if we want to show subplots
        if showsub == 1
            figure;
        end
        % Vary over phase randomizations
        for k=1:length(sigtest)
            % Add randomized phase to second circle
            I2 =
         cPixels2.*exp(j*(2*pi/lambda)*(sigtest(k)*rand(N,M)));
            I2 = padarray(I2,[(M*paddingFactor-M)/2
         (N*paddingFactor-N)/2]);

            % Back propagate circles
            temp1 = fresnel_advance(I1,pitch(p),pitch(p),-z1,lambda);
            temp2 = fresnel_advance(I2,pitch(p),pitch(p),-z2(l),lambda);

            % Superimpose back propagated fields
            tempFinal = temp1+temp2;
            tempFinal = padarray(tempFinal(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);

            % Forward propagate to first circle
            OutF1 = fresnel_advance(tempFinal,pitch(p),pitch(p),z1,lambda);

            %%%%% Get power error %%%%%
            error = abs(OutF1).^2-abs(I1).^2;
            if varypitch == 1
                error = abs(error);
            end

            % Normalize Output
            OutF1 = OutF1/max(abs(OutF1(:)));
            OutF1=OutF1(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
            % Show subplot if desired
            if showsub == 1
                subplot(2,length(sigtest),k)
                imshow(abs(OutF1).^2);
                title(['Coefficient = ',num2str(sigtest(k))],'FontSize',16)
            end

```

```
    % Output at second circle
    OutF2 = fresnel_advance(tempFinal,pitch(p),pitch(p),z2(l),lambda);
    OutF2 = OutF2/max(abs(OutF2(:)));

    if showsub == 1
        subplot(2,length(sigtest),k+length(sigtest))
        imshow(error); % power error is already in power
        title(['Error when Coefficient=' num2str(sigtest(k))],'FontSize',16)
    end

    % Calculate weighted mean std. dev. for measure of spread
    errorNormalized = error/sum(sum(error)); % Need the error to be like a probabilitiy (sumsum=1) to be a weight
    x_center(k) = sum(sum(errorNormalized.*x2));
    y_center(k) = sum(sum(errorNormalized.*y2));

    if varypitch == 1
        spreadCalc(l,g) = sqrt(sum(sum(((x2-x_center(k)).^2+(y2-y_center(k)).^2).*errorNormalized))); %Weighted Mean Std. Deviation, http://en.wikipedia.org/wiki/Weighted_arithmetic_mean
    else
        spreadCalc(k,l) = sqrt(sum(sum(((x2-x_center(k)).^2+(y2-y_center(k)).^2).*errorNormalized))); %Weighted Mean Std. Deviation, http://en.wikipedia.org/wiki/Weighted_arithmetic_mean
    end
end
```

```
    end
        % Plot
        if varypitch ~= 1
            figure;
            plot(sigtest,spreadCalc)
            xlabel('Randomness')
            ylabel('Weighted Mean Standard Deviation')
        end
end
% Plot
if varypitch == 1
    figure;
    plot(Nsize,spreadCalc);
    xlabel('Pitch')
    ylabel('Weighted Mean Standard Deviation')
end
```

CircularPhase.m

This is the wrapper code used to test the effects of circular phase. A output shape is chosen and circular phase can be added to the shape if desired. Then this output is propagated a range of distances both forward and backwards. The percentage of the power that is outside the desired phase array area (128x128) is calculated to compare the flat phase and the circular phase.

```
% This code tests the effects of circular phase added to outputs

% Variable to control output shape
shape = 2; %1 = M from MIT logo, 2 = circle, 3 = Gaussian, default = 128x128 box
% Variable to switch between flat and circular phase
circular = 1; %1 to turn on circular phase front

% Wavelength and antenna spacing
lambda = 1.55e-6;
pitch = 9e-6;

% array dimensions
N=128;
M=128;

% Output distance
z1 = 0.028;
% Generate grid matrices of x and y positions
x1 = (-M/2+1)*pitch:pitch:pitch*M/2;
y1 = (-N/2+1)*pitch:pitch:pitch*N/2;
[x1, y1] = meshgrid(x1,y1);

% Choose output shape
switch shape
    case 1
        % Original MIT Logo (testing with just the M)
        I = imread('MIT_logo', 'jpeg');
        I = imresize(I,[128 128]);
        I = double(256*im2bw(I,0.8));
        I(36,89)=256;
        I1 = imcrop(I, [0 0 65 128]);
        I1 = padarray(I1, [0 128-65], 'post');
        I1 = imresize(I1, [N M],'box');
    case 2
        % Testing with a circle
        [columnsInImage rowsInImage] = meshgrid(1:N, 1:M);
        centerX = 64;
        centerY = 64;

        radius = 64;
        circlePixels = (rowsInImage - centerY).^2 ...
            + (columnsInImage - centerX).^2 <= radius.^2;
        I1 = double(circlePixels);
    case 3
        % Testing with Gaussian
        wid = 0.2E-3; % This is the 1/e width
        I1 = exp(-(x1.^2+y1.^2)/(wid^2));
    otherwise
        % Testing with box
        I1 = zeros(N,M)+256;
end

% Circular phase front with radius of z1 if user wants it
if circular == 1
    I1 = I1.*exp(j*(2*pi*z1/lambda)*(-(-
         sqrt((x1.^2+y1.^2)/z1^2+1)+1)));
end

% Padding
paddingFactor = 5; % Increase this to add more padding
I1 = padarray(I1,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);

% Plot our distrubution with no propagation
```

```
figure
subplot(2,1,1)
imagesc(abs(I1))
title('No prop - Abs()')
colorbar
subplot(2,1,2)
imagesc(angle(I1))
title('No prop - Phase()')
colorbar

% Back and forward propagate distances
z = [-(2*z1):0.0035:(2*z1)]*1;
disp(['Propagating ' num2str(length(z)) ' values'])
% Values of percentage power lost outside desired area
PowerLoss = zeros(length(z),1);

% Vary over our distances
for k=1:length(z)
    % Propagate by z(k)
    temp = fresnel_advance(I1,pitch,pitch,z(k),lambda);
    % Normalize
    temp = temp/max(abs(temp(:)));
    % Show propagated field
    figure;
    imshow(abs(temp));
    % Calculate percentage of power lost outside desired PA area
    PowerLoss(k)=(sum(sum(abs(temp).^2)))-
         sum(sum(abs(temp(((M*paddingFactor-
         M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
         N)/2+1):((N*paddingFactor-N)/2+N))).^2));
    PowerLoss(k)=PowerLoss(k)/(sum(sum(abs(temp).^2)));
    PowerLoss(k)=100*PowerLoss(k);
    title(['Propagation of ' num2str(z(k)) ' m '],'FontSize',26)
end

```

```
% Plot
figure;
plot(z,PowerLoss);
title('Percent Power Loss at Various Propagation
         Distances','FontSize',24);
xlabel('Propagation Distance','FontSize',24);
ylabel('Percent Power Loss','FontSize',24);
```

NoiseAnalysis.m

This code adds a random Gaussian noise to the phase or amplitude profiles of the summed backpropagated outputs. Then it forward propagates this field and stores the outputs. An inner product is calculated between these stored fields and the resultant field when no noise is added to calculate an overlap between the results.

```
% This code tests the effects of amplitude or phase noise on
        forward
% propagation

% Wavelength & antenna spacing
lambda = 1.55e-6;
pitch = 9e-6;

% Array dimensions
N=128;
M=128;

% Output distances
z1 = 0.007;
z2 = 0.014;
z3 = 0.028;

% Generate grid matrices of x and y positions
x1 = (-M/2+1)*pitch:pitch:pitch*M/2;
y1 = (-N/2+1)*pitch:pitch:pitch*N/2;
[x1 y1] = meshgrid(x1,y1);

% MIT logo
I = imread('MIT_logo', 'jpeg');
I = imresize(I,[128 128]);
I = double(256*im2bw(I,0.8));
I(36,89)=256;

```

```
% Segmenting Logo
% M output
I1 = imcrop(I, [0 0 65 128]);
I1 = padarray(I1, [0 128-65], 'post');
I1 = imresize(I1, [N M],'box');
I1 = 1.25*I1.*exp(j*(2*pi*z1/lambda)*(-(-
        sqrt((x1.^2+y1.^2)/z1^2+1)+1)+N*pitch^2/z1^2*rand(N,M)))
        ;

% I output
I2 = imcrop(I, [65 0 20 128]);
I2 = padarray(I2, [0 64],'pre');
I2 = padarray(I2, [0 128-85],'post');
I2 = imresize(I2, [N M],'box');
I2 = 1.85*I2.*exp(j*(2*pi*z2/lambda)*(-(-
        sqrt((x1.^2+y1.^2)/z2^2+1)+1)+1.2*N*pitch^2/z2^2*rand(N,
        M)));

% T output
I3 = imcrop(I, [85 0 128-85 128]);
I3 = padarray(I3, [0 84],'pre');
I3 = imresize(I3, [N M],'box');
I3 = I3.*exp(j*(2*pi*z3/lambda)*(-(-
        sqrt((x1.^2+y1.^2)/z3^2+1)+1)+1.6*N*pitch^2/z3^2*rand(N,
        M)));

% Padding
paddingFactor = 5; % Increase this to add more padding
I1 = padarray(I1,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);
I2 = padarray(I2,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);
I3 = padarray(I3,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);

% Back propagation of M
temp1 = fresnel_advance_FoundOnline(I1,pitch,pitch,-z1,lambda);
```

```
temp1 = padarray(temp1(((M*paddingFactor-
        M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
        N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-
        M)/2 (N*paddingFactor-N)/2]);
Out1 = fresnel_advance_FoundOnline(temp1,pitch,pitch,z1,lambda);
Out1 = Out1/max(abs(Out1(:)));
figure
imshow(abs(Out1(((M*paddingFactor-M)/2+1):((M*paddingFactor-
        M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
        N)/2+N))));

% Back propgation of I
temp2 = fresnel_advance_FoundOnline(I2,pitch,pitch,-z2,lambda);
temp2 = padarray(temp2(((M*paddingFactor-
        M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
        N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-
        M)/2 (N*paddingFactor-N)/2]);
Out2 = fresnel_advance_FoundOnline(temp2,pitch,pitch,z2,lambda);
Out2 = Out2/max(abs(Out2(:)));
figure
imshow(abs(Out2(((M*paddingFactor-M)/2+1):((M*paddingFactor-
        M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-
        N)/2+N))));

% Back propagation of T
temp3 = fresnel_advance_FoundOnline(I3,pitch,pitch,-z3,lambda);
temp3 = padarray(temp3(((M*paddingFactor-
        M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-
        N)/2+1):((N*paddingFactor-N)/2+N)), [(M*paddingFactor-
        M)/2 (N*paddingFactor-N)/2]);
Out3 = fresnel_advance_FoundOnline(temp3,pitch,pitch,z3,lambda);
Out3 = Out3/max(abs(Out3(:)));
figure
```

```
imshow(abs(Out3(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N))));

% Variation of phase and amplitude noise
phasenoise = 0:pi/32:pi/2;%[0,pi/16,pi/8,pi/4];
ampnoise = 0:0.05:1;%[0,0.05,0.1,0.5];
% Holds calculations
OutM_phase = zeros(M,N,length(phasenoise));
OutI_phase = zeros(M,N,length(phasenoise));
OutT_phase = zeros(M,N,length(phasenoise));
OutM_amp = zeros(M,N,length(ampnoise));
OutI_amp = zeros(M,N,length(ampnoise));
OutT_amp = zeros(M,N,length(ampnoise));

% Superimposing back propagations
tempF = temp1+temp2+temp3;
tempF = tempF(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
% Test over Phase noise variations
for k=1:length(phasenoise)
% Add noise to phases of original summed field
tempFinal = tempF.*exp(phasenoise(k)*randn(M,N)); %Gaussian Phase Noise Analysis
tempFinal = padarray(tempFinal,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);

% Forward propagate to M
OutF1 = fresnel_advance_FoundOnline(tempFinal,pitch,pitch,z1,lambda);
OutF1 = OutF1/max(abs(OutF1(:)));
% Forward propagate to I
```

```
OutF2 = fresnel_advance_FoundOnline(tempFinal,pitch,pitch,z2,lambda);
OutF2 = OutF2/max(abs(OutF2(:)));
% Forward propagate to T
OutF3 = fresnel_advance_FoundOnline(tempFinal,pitch,pitch,z3,lambda);
OutF3 = OutF3/max(abs(OutF3(:)));

% Hold forward propagated fields
OutM_phase(:,:,k)=OutF1(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
OutI_phase(:,:,k)=OutF2(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
OutT_phase(:,:,k)=OutF3(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
end

for k=1:length(ampnoise)
% Add noise to amplitude of original summed field
tempFinal = tempF.*(1+ampnoise(k)*randn(M,N)); %Gaussian Amp Noise Analysis
tempFinal = padarray(tempFinal,[(M*paddingFactor-M)/2 (N*paddingFactor-N)/2]);

% Forward propagate to M
OutF1 = fresnel_advance_FoundOnline(tempFinal,pitch,pitch,z1,lambda);
OutF1 = OutF1/max(abs(OutF1(:)));

```

```
% Forward propagate to I
OutF2 = fresnel_advance_FoundOnline(tempFinal,pitch,pitch,z2,lambda);
OutF2 = OutF2/max(abs(OutF2(:)));

% Forward propagate to T
OutF3 = fresnel_advance_FoundOnline(tempFinal,pitch,pitch,z3,lambda);
OutF3 = OutF3/max(abs(OutF3(:)));

% Hold forward propagated fields
OutM_amp(:,:,k)=OutF1(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
OutI_amp(:,:,k)=OutF2(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
OutT_amp(:,:,k)=OutF3(((M*paddingFactor-M)/2+1):((M*paddingFactor-M)/2+M),((N*paddingFactor-N)/2+1):((N*paddingFactor-N)/2+N));
end


% Inner Product
innerproductM_phase=zeros(length(phasenoise),1);
innerproductI_phase=zeros(length(phasenoise),1);
innerproductT_phase=zeros(length(phasenoise),1);
innerproductM_amp=zeros(length(ampnoise),1);
innerproductI_amp=zeros(length(ampnoise),1);
innerproductT_amp=zeros(length(ampnoise),1);
for k=1:length(phasenoise)
    % Calculate inner product for fields altered by phase noise
```

```
        innerproductM_phase(k)=sum(sum(OutM_phase(:,:,k).*conj(OutM_phase(:,:,1))))/sqrt(sum(sum(OutM_phase(:,:,k).*conj(OutM_phase(:,:,k))))*sum(sum(OutM_phase(:,:,1).*conj(OutM_phase(:,:,1)))));
        innerproductI_phase(k)=sum(sum(OutI_phase(:,:,k).*conj(OutI_phase(:,:,1))))/sqrt(sum(sum(OutI_phase(:,:,k).*conj(OutI_phase(:,:,k))))*sum(sum(OutI_phase(:,:,1).*conj(OutI_phase(:,:,1)))));
        innerproductT_phase(k)=sum(sum(OutT_phase(:,:,k).*conj(OutT_phase(:,:,1))))/sqrt(sum(sum(OutT_phase(:,:,k).*conj(OutT_phase(:,:,k))))*sum(sum(OutT_phase(:,:,1).*conj(OutT_phase(:,:,1)))));
end
for k=1:length(ampnoise)
    % Calculate inner product for fields altered by amplitude noise
        innerproductM_amp(k)=sum(sum(OutM_amp(:,:,k).*conj(OutM_amp(:,:,1))))/sqrt(sum(sum(OutM_amp(:,:,k).*conj(OutM_amp(:,:,k))))*sum(sum(OutM_amp(:,:,1).*conj(OutM_amp(:,:,1)))));
        innerproductI_amp(k)=sum(sum(OutI_amp(:,:,k).*conj(OutI_amp(:,:,1))))/sqrt(sum(sum(OutI_amp(:,:,k).*conj(OutI_amp(:,:,k))))*sum(sum(OutI_amp(:,:,1).*conj(OutI_amp(:,:,1)))));
        innerproductT_amp(k)=sum(sum(OutT_amp(:,:,k).*conj(OutT_amp(:,:,1))))/sqrt(sum(sum(OutT_amp(:,:,k).*conj(OutT_amp(:,:,k))))*sum(sum(OutT_amp(:,:,1).*conj(OutT_amp(:,:,1)))));
end
```

```

% Plot inner products due to amplitude/phase noise
figure;
plot(phasenoise,abs(innerproductM_phase));
title(['Inner Product of M due to Phase Noise'],'FontSize',16);
figure;
plot(phasenoise,abs(innerproductI_phase));
title(['Inner Product of I due to Phase Noise'],'FontSize',16);
figure;
plot(phasenoise,abs(innerproductT_phase));
title(['Inner Product of T due to Phase Noise'],'FontSize',16);
figure;
plot(ampnoise,abs(innerproductM_amp));
title(['Inner Product of M due to Amplitude
        Noise'],'FontSize',16);
figure;
plot(ampnoise,abs(innerproductI_amp));
title(['Inner Product of I due to Amplitude
        Noise'],'FontSize',16);
figure;
plot(ampnoise,abs(innerproductT_amp));
title(['Inner Product of T due to Amplitude
        Noise'],'FontSize',16);

% Plots
figure;
set(gca,'FontSize', 16);
plot(phasenoise,abs(innerproductM_phase),'r');
hold on
plot(phasenoise,abs(innerproductI_phase),'g');
hold on
plot(phasenoise,abs(innerproductT_phase),'b');
xlabel('Standard Deviation of Noise');
ylabel('Percentage Overlap');
leg1 = legend('M Output', 'I Output', 'T Output');
```

```
hold off
% Plots
figure;
set(gca,'FontSize', 16);
plot(ampnoise,abs(innerproductM_amp),'r');
hold on
plot(ampnoise,abs(innerproductI_amp),'g');
hold on
plot(ampnoise,abs(innerproductT_amp),'b');
xlabel('Standard Deviation of Noise');
ylabel('Percentage Overlap');
leg2 = legend('M Output', 'I Output', 'T Output');
hold off
```

CouplerRatioCalc.m

This code calculates the required coupling coefficients for the row and unit couplers. It follows the algorithm explained earlier that uses a system of equations to calculate the coefficients that are dependent on previous coefficients within the row or column.

```
% This code calculates the required coupling coefficients needed to achieve
% the desired amplitudes

% Get phased array electric field
load('FinalPhasedArray128.mat');
[r c]=size(tempF);
M = 128;
N = 128;
tempF=tempFinal(((r-M)/2+1):((r-M)/2+M),((c-N)/2+1):((c-N)/2+N));
% Calculate distribution of required power for each antenna
PowerDist=abs(tempF).^2;

[rows,cols]=size(tempF);
% Calculate column vector of required power per row
TotalProws=sum(PowerDist,2);
% Calculate total power required for entire system
TotalP=sum(TotalProws);

% First row ratio will just be the required power for that row divided by the
% total power of the system
Rowratios=zeros(rows,1);
Rowratios(1)=TotalProws(1)/TotalP;

% For loop to calculate rest of row ratios
for iter=2:rows
    % TotalP(1-r(1))(1-r(2))...(1-r(i-1))r(i) = P(i), calculate r(i)
    Rowratios(iter)=TotalProws(iter)/TotalP;
    for k=1:iter-1
        Rowratios(iter)=Rowratios(iter)/(1-Rowratios(k));
    end
end

Colratios=zeros(rows,cols);
% For loop to calculate individual antenna coupling ratios
for rowiter=1:rows
    % TotalP(1-r(1))(1-r(2))...(1-r(i-1))r(i) = P(i), calculate r(i)
    Colratios(rowiter,1)=PowerDist(rowiter,1)/TotalProws(rowiter);
    for coliter=2:rows
        Colratios(rowiter,coliter)=PowerDist(rowiter,coliter)/TotalProws(rowiter);
        for k=1:coliter-1
            Colratios(rowiter,coliter)=Colratios(rowiter,coliter)/(1-Colratios(rowiter,k));
        end
    end
end
```