UNIVERSITY OF Waterloo

Department of Mechanical and Mechatronics Engineering

Work Term Report 200: Development of a simulation enabled phase retrieval algorithm

A Report Prepared For:
The University of Waterloo

Prepared By:
Adam Lastovka

January 22, 2024

Table of Contents

List of Figures

List of Tables

1.0 Summary

In summary, this report has outlined the development of a simulation-based phase retrieval algorithm for implementation in a laser micro machining station. An overview of the phase retrieval problem was given, and the unsuitability of conventional optimization techniques was discussed. The Gerchberg-Saxton algorithm was then explained, and initial phase retrieval results were shown to be inadequate motivating the development of the wave propagation simulation.

The physical system was described, and the modelling considerations and assumptions were presented. Different wave propagation models were evaluated and the Rayleigh-Sommerfield diffraction method was selected. The laser beam profile as well as the lens representations were shown, and limitations were discussed. The forward and backward propagation paths were then modelled.

The Diffractio library was used to implement the mathematical model, and a custom object-oriented simulation framework was developed for ease of use. A method to virtually increase the sampling of the computation window while maintaining simulation fidelity was created to improve simulation accuracy. The Gerchberg-Saxton algorithm was then implemented with additional considerations to account for the computation window extending beyond the SLM and the effects of supersampling.

Due to time constraints, experimental data could not be collected to fully verify the simulation. Instead, the simulation was tested qualitatively by simulating analytically defined phase masks with known diffraction patterns. The simulation results are promising as the diffraction patterns match the expected results. The location of the focus from the lens was analytically verified using the lens equation. The effect of supersampling was investigated, and the expected increase in resolution and accuracy was observed. From this it was determined that the base sampling of 1024x1072 was sufficient to accurately represent the system as the intensity distributions did not change with increasing sampling.

The phase retrieval algorithm was then tested and the results of three tests were discussed. Generated intensity distributions closely matched the target distributions (including complex targets such as an image of the moon) in all cases showing the efficacy of the phase retrieval algorithm. Only a couple iterations were required for good retrieval results which was surprising. The effect of Supersampling on phase retrieval was briefly investigated, however an improvement in performance was not observed. Recommendations for further development and investigation were then presented. The need for experimental verification of the simulation and generated phase masks was emphasized, and tuning parameters were identified. Suggestions for further improvement and research were then discussed.

2.0 Introduction

The problem of phase retrieval is widespread in many applications dealing with electromagnetic waves such as crystallography, optical systems, or communication networks. Phase retrieval is the process of finding the phase of a signal or wave solely from its measured intensity at a point in space.

I encountered the problem of phase retrieval as part of the Laser Micro Machining team at HiLase. Part of the 7-axis laser machining station that I was working on was a Pulsar Photonics flexible beam shaper module. Using the beam shaper, the intensity profile of a laser beam can be altered from an input distribution (usually a gaussian) to some desired intensity distribution. This has many applications and is useful for machining specialized surface structures, or for multi-beam machining which can increase machining speed significantly.

The beam shaper uses a liquid-crystal-on-silicon (LCoS) phase-only spatial-light-modulator (SLM) which is an optical element that can be programmed with a phase mask to change the phase of incident light at different spatial locations. The altered laser beam then diffracts through the rest of the laser system until it reaches the machining surface and produces the desired intensity distribution. This diffraction process is very complex.

Several mathematical models exist to describe the propagation of waves through different media. Given a known SLM configuration, these can be used to find the electrical field in the image plane (and thus the intensity distribution), however, given a desired intensity distribution these equations cannot yield the necessary phase at the SLM without knowing the phase at the image plane. This motivates the problem of phase retrieval.

In the context of my work term, the beam shaper module came with proprietary software that allowed programming of the SLM, and even included a well-tuned phase retrieval algorithm. Unfortunately, the software package was very cumbersome to use and did not have a well-defined way to configure the phase masks being uploaded to the SLM. On top of this, there was no technical documentation describing the inner workings of the software and the supplier provided limited customer support which posed concerns for future operations in case the system had to be changed and recalibrated. Thus arose a need to replace the existing software with another algorithm to generate desired target intensity profiles.

Furthermore, to complicate the problem at hand, the laser machining station was not set to be operational until the end of my coop term and so an experimental methodology could not be used. To combat this, a simulation-based approach was pursued. The use of a simulation also offered several advantages over practical experiments including the ability to extract electrical field information at any point in the system, allowing simulation of thousands of test cases, and avoiding the expense of running the laser station.

2.1 Problem Definition

In summary, the tasks I tackled during my work term were the following:

1. Develop a simulation of the laser system to enable testing of generated phase masks.
2. Develop an algorithm that generates a phase mask given a target intensity distribution.

2.2 Background Information

To understand the topics discussed in this report, an intuitive understanding of the electromagnetic nature of light is necessary. Light is a wave of electric and magnetic fields that propagates through free space at the speed of light. It is characterized by a wavelength λ, and an amplitude as shown in the diagram below. Wavelength is related to frequency by the equation shown in eq. (1) where *c* is the speed of light and *f* is the frequency.

$$\lambda = \frac{c}{f} \quad (1)$$

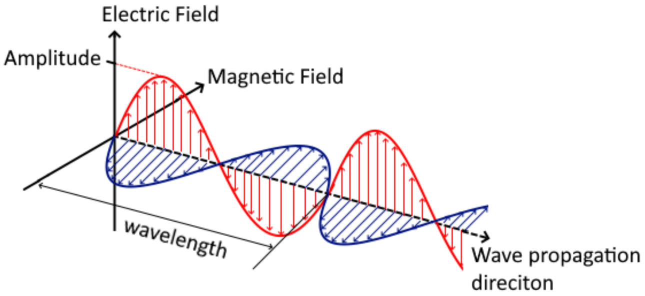


Figure 1 Electromagnetic wave diagram

The square of the amplitude is the intensity of the light. While the electric and magnetic fields are vector fields by definition, for the analysis in this paper a scalar field approximation will be sufficient as the methods used don't require a vector representation of light. It follows that the oscillation of the electric field at a point in space as a function of time can be described by eq. (2) assuming it is propagating through a lossless medium.

$$E(t) = A_0 cos(\omega t + \phi) \quad (2) \qquad E(t) = A_0 e^{j\omega t + \phi} \quad (3)$$

A_0 is the amplitude of the wave, ω is the angular velocity, which is related to frequency, and ϕ is the phase shift. For many purposes it is useful to use a phasor representation instead which makes use of the complex plane. This representation simplifies many mathematical calculations involving fields and is shown in eq. (3).

A laser is a device that emits highly coherent and collimated light. This means that the produced electromagnetic field is monochromatic (oscillates with only one frequency) such that at two arbitrary points in the field the phase difference is constant. This property is essential for the applications discussed in this paper.

While phase is seemingly secondary in importance to the amplitude of a wave, for imaging purposes it is very significant. This is because at point in space, the phase defines the component of the amplitude which is expressed in the electric field at that point. This is demonstrated in Figure 2 below.

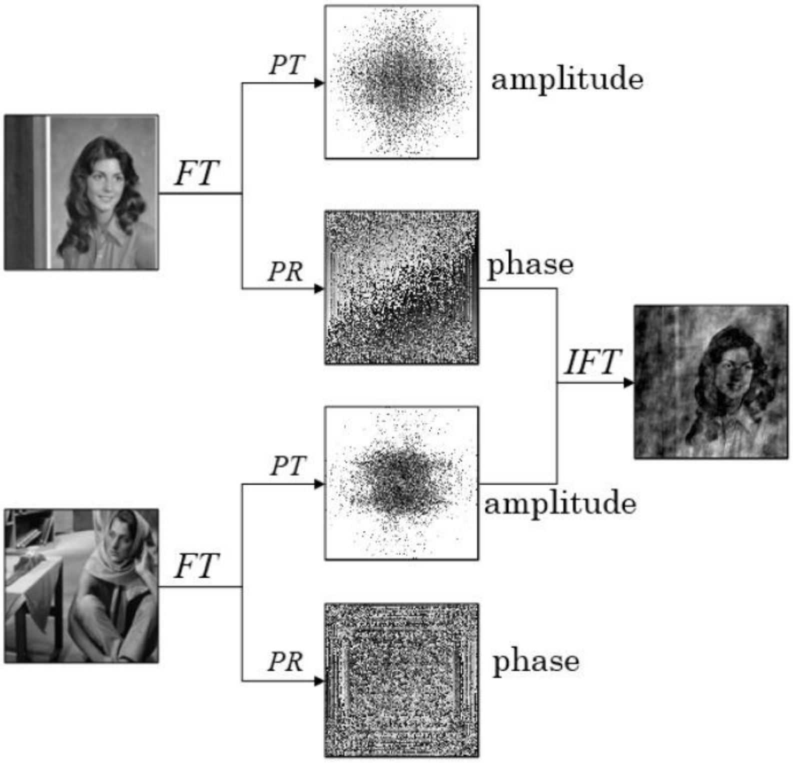


Figure 2 Diagram showing importance of phase in imaging [1]

As can be seen, when the phase of one image and the amplitude of another image are combined in the frequency domain and then converted back to the spatial domain, the image from which the phase was taken is visible. This imaging situation mimics the situation that occurs in the beam shaper system and illustrates why the SLM can change the shape of the intensity distribution only through phase changes.

3.0 Phase retrieval algorithm

At first glance, the problem of finding a phase mask to produce a desired intensity distribution is an optimization problem that could be solved using traditional optimization methods such as stochastic gradient descent, genetic algorithm, or particle swarm. However, these methods are ill-suited for this problem for several reasons:

- High dimensional parameter space –The SLM has 1024x1072 programmable pixels which equals 1097728 parameters, and the overall search space has $256^{1097728}$ potential states. To effectively map this search space and find an optimal solution would be extremely computationally intensive.
- Complex objective function – the function describing the relationship between the phase mask and the output intensity distribution is very complex and involves non-linear operations which pose a challenge for optimization algorithms.
- Non-convex optimization problem – Due to the nature of wave diffraction many local optima and plateaus will exist in the search space. This significantly hinders optimization convergence.

Instead, physically-informed algorithms exist that can effectively solve the phase retrieval problem. The most popular algorithm for phase retrieval is the Gerchberg-Saxton (GS) algorithm as described in [2]. Starting with a random initial phase distribution and the measured input amplitude at the object plane, the algorithm applies a Fourier transformation to convert to a frequency representation. There the target intensity distribution is enforced while maintaining the propagated phase information. An inverse Fourier transformation is then applied to convert to the spatial domain yielding an estimate of the phase information at the object plane. This process is repeated until convergence is achieved. The algorithm is presented visually in the image below.

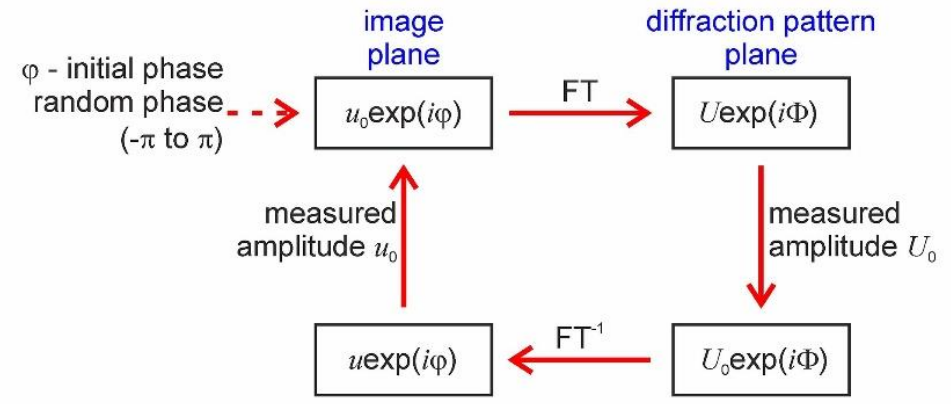


Figure 3 Visualization of GS algorithm [3]

Through this process, the phase corresponding to a target intensity distribution at the image plane is found. This is then converted back to the spatial domain to yield the phase mask required to produce the desired intensity distribution.

3.1 Initial GS algorithm implementation

This algorithm was implemented in python using the NumPy library [4]. The results after 100 iterations of the GS algorithm are shown below.

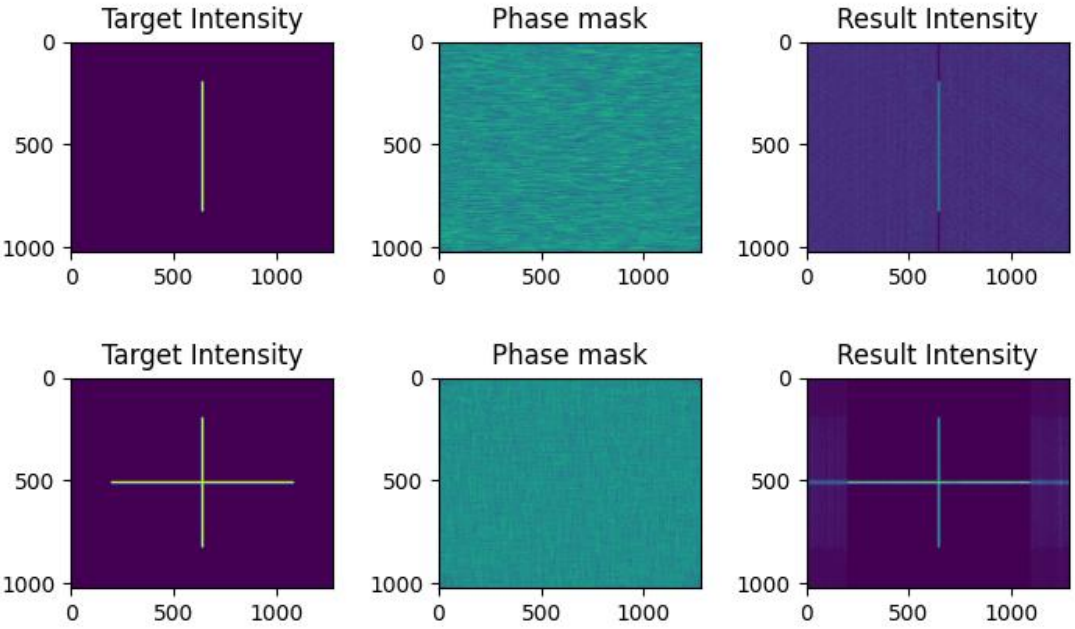


Figure 4 Initial GS algorithm results

While promising, these results are misleading as they do not correspond to what would be observed in the laser system. This is because the original GS algorithm retrieves the phase of the image using only the Fourier transform which is only valid in representing wave propagation in the far field. Furthermore, the algorithm does not account for additional optical elements in the path of the wave as is the case in the laser machining system and thus the results are not representative. For this reason, the wave propagation model had to be developed.

A diagram showing the implemented phase retrieval algorithm is shown in Figure 5. Details of the code implementation are discussed in section 4.6.

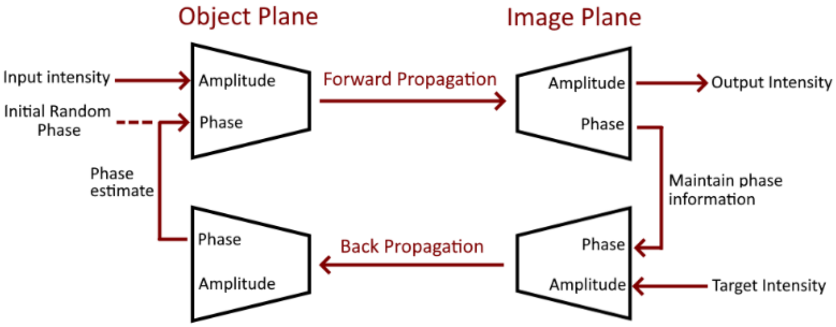


Figure 5 Implemented GS algorithm.

4.0 Simulation development

The first stage of developing the wave propagation simulation involved characterizing the laser system. An annotated diagram of the relevant part of the laser system is shown below:

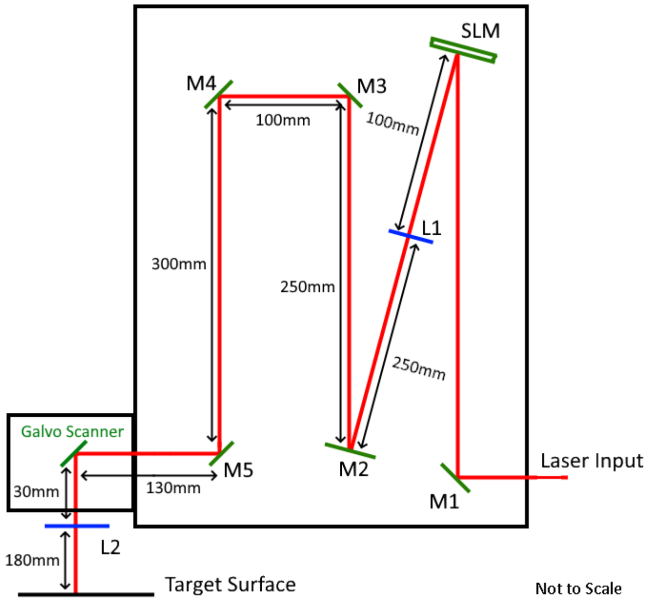


Figure 6 Diagram of Pulsar beam shaper

The dimensions shown were measured using a digital caliper and the values were rounded to the nearest whole number to account for measurement errors. The distances from M5 to the galvo scanner mirrors and from the galvo scanner to L2 represent rough estimates as these distances could not be measured without dismantling the galvo scanner. Furthermore, it is important to note that lens L2 was roughly 10cm long thus the measurement to the target surface was taken from the bottom most face of the lens. In the developed model it was approximated as a thin lens.

Elements M2,M4, M5 and the galvo scanner are mirrors with a very high reflectance and were modelled as perfect mirrors. Mirrors M1 and M3 were semi-transparent mirrors that were intended to enable beam profile monitoring; however, this introduced an undesirable intensity and phase change. In this report this effect was not modelled. Lenses L1 and L2 have a focal length of f=700mm and f=160mm respectively. L2 was an F-Theta lens which serves to create a uniform image height regardless of the incident angle of the laser beam. As the galvo scanner wasn't used, the laser beam was always orthogonal to the F-Theta lens and thus the lens acted as a normal converging lens.

Only optical elements downstream of the SLM had to be modelled as the laser incident on the SLM does not change. The laser beam had the following characteristics:

- 6x6mm isotropic gaussian intensity profile
- 1032 nm wavelength
- 0-100W Power (25W max allowable for SLM)

The SLM had a resolution of 1024x1272 pixels and had a size of 12.8mm x 15.9mm. It follows that each pixel had a dimension of 12.5um x 12.5um. Furthermore, each pixel could be programmed to produce a phase shift from -π to π with 256 discrete levels. In the beam shaper control software, the SLM is programmed using a 1024x1072 grayscale PNG image with each pixel being in the range [0,210].

4.1 Mathematical model

The main models of scalar wave propagation are the Rayleigh-Sommerfeld (RS) and Helmholtz-Kirchoff diffraction integrals. While closely related, the RS method is more general, and can be applied to a wider range of scenarios [5]. For this reason, the Rayleigh-Sommerfeld model was selected for this simulation. Additionally, several approximations of the Rayleigh-Sommerfeld diffraction integral exist including the Fraunhofer and Fresnel approximation which are valid in the Far-field, and Near-field respectively. These approximations simplify the calculation of the integral and make it less computationally intensive at the cost of accuracy [6]. As such, the approximations were avoided to establish a high accuracy simulation to serve as a foundation for trials with less intensive algorithms.

Having selected a diffraction model, the propagation of the laser beam through the system could be modelled. Figure 7 shows the steps required to propagate from the input to the target (forward propagation), as well as propagation in the reverse direction (back propagation). Back propagation had to be modelled as it is a necessary step in the phase retrieval algorithm described in Section 3.0.

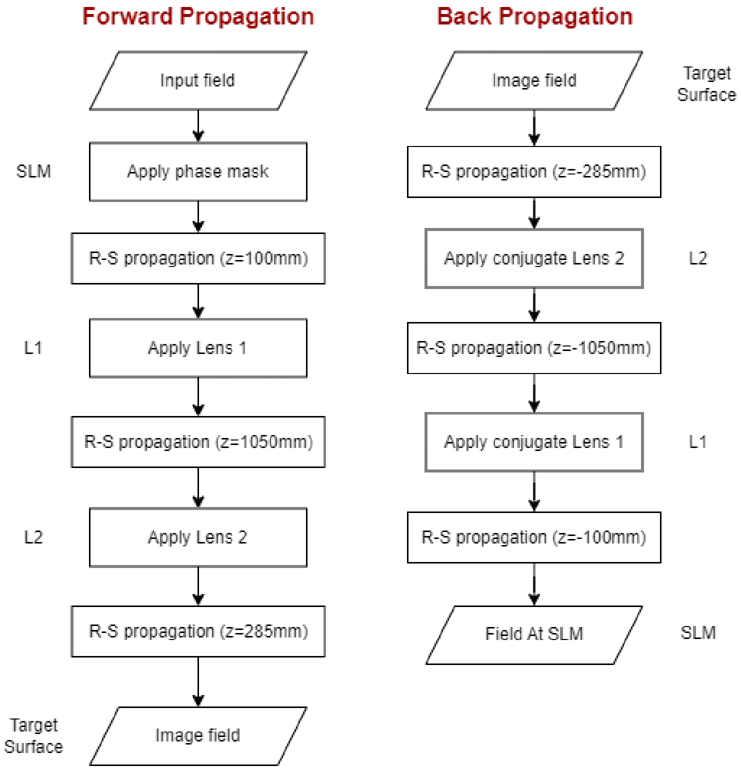


Figure 7 Wave propagation model flowchart

A well-developed implementation of RS diffraction was found in the python Diffractio library [7]. The library uses an FFT based direct integration method to solve the diffraction integral which yields very accurate results [8]. Additionally, the algorithm implemented in the library calculates a convenient quality factor which expresses whether the sampling of the computational field is sufficient to yield accurate results (by satisfying the Nyquist criterion).

This library provides an object-oriented framework for creating arrays representing the scalar fields and applying the RS method to simulate wave propagation. Details of the python implementation are shown in the following sections.

4.2 Computational Window definition

The computation window is the spatial area that is being simulated. It has two key dimensions associated with it – it's spatial size and pixel sampling. The spatial size defines the physical size of space where the scalar fields are being calculated, and the sampling defines the grid of points at which the area is sampled. The chosen spatial size and sampling are displayed in the figure below.

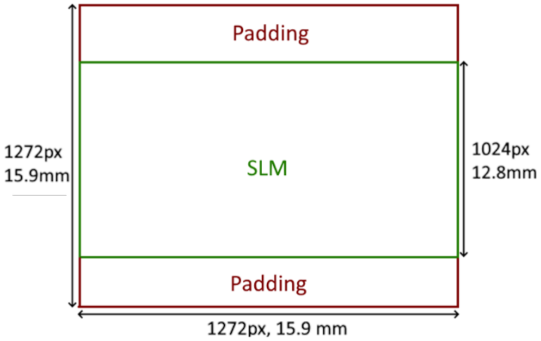


Figure 8 computation window diagram

This set-up represents the simplest representation of the system as each pixel in the SLM is represented by one sampling point. A lower resolution would result in aliasing and loss of phase information. On the other hand, a higher sampling rate would yield a more accurate simulation at the cost of being more computationally intensive. For initial trials, the 1072x1272 sampling was used to test the simulation and in later trials the use of super sampling was investigated as described in section 4.5. The padding was added to create a square computation window which prevents distortions that could arise from the Fourier transformations used in the RS method.

4.3 Gaussian beam, and lens definition

The input gaussian beam was created using Diffractio's built in gauss_beam method which creates a 2D gaussian distribution with a specified beam waist. Per the beam specifications, the beam waist was set to 6mm resulting in the distribution shown in Figure 9 as a heat map. The maximum amplitude was arbitrarily set to 1 as the absolute intensity does not affect the beam propagation. Furthermore, the gaussian beam has a uniform zero phase field.

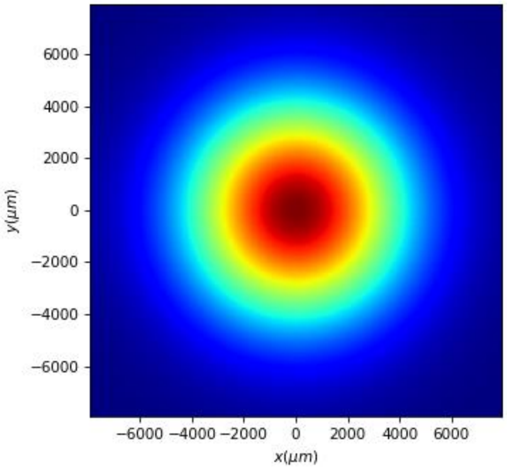


Figure 9 Heat map of Input gaussian beam intensity

Similarly, the lenses L1 and L2 were generated using Diffractio's built in function lens_spherical() which generates the lens field representation given a specific focal length. The phase fields are shown in Figure 10.

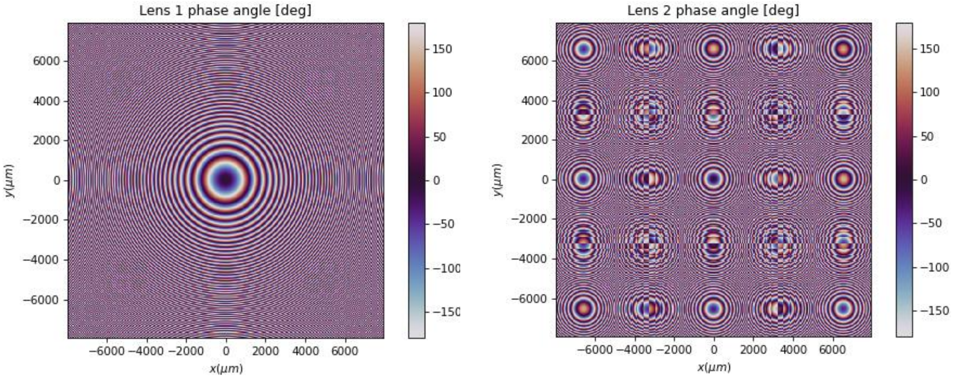


Figure 10 Heat map of Lens phase representation

The lens fields had a uniform amplitude of 1 which corresponds to perfect transmittance (no energy loss). While this is an idealized assumption, the actual energy losses are likely negligible.

4.4 Object oriented framework

For ease of use, an object-oriented framework was used to implement the simulation. In object-oriented programming, data-structures called objects are defined as instances of a class. A class lays out the properties and methods that each object can have. By defining a simulation object, key information about the system such as the current phase mask or the sampling can be stored and then easily accessed by methods such as, for example, forwards propagation. The properties and methods of the developed simulation class are shown in Table 1 and Table 2.

Table 1 Simulation object properties

Property	Description
sampling_size	Integer defining the sampling of the computation window
supersampling_factor	Integer defining the supersampling factor
x_size, y_size	Float value defining the spatial size of the computation window
x_vect, y_vect	Vectors of spatial coordinates of the computation field
Lens_1, lens_2	Scalar field of lenses
lens_1b, lens_2b	Scalar field of conjugate lenses
Mask	Scalar field of phase mask
Input_field	Scalar field of incident laser beam
Image_field	Scalar field produced in the image plane

Table 2 Simulation object methods

Method	Description
Propagate_forward	Forward propagates the input_field through the system and updates the image_field
Propagate_ backward	Back propagates the image_field through the system and returns the field at the SLM
Extract_mask	Helper function to extract the phase field of the SLM and scale it to the pixel range required by the control software.
Clip_SLM	Sets field outside of SLM to zero and returns resulting field
Retrieve_mask	Runs the phase retrieval algorithm for a set number of iterations and outputs the resulting phase mask

The simulation object is initialized by specifying the wavelength of the laser beam to be propagated, the computation window size, the sampling, and the SLM mask. The initialization function then computes the spatial vectors for the spatial dimension vectors which are then used to generate the source intensity, and the lens masks. The properties in Table 1 are then initialized with the calculated values.

4.5 Supersampling

Supersampling is a process usually used for anti-aliasing of computer images. It involves increasing the resolution of an image by taking multiple samples of a single pixel and then taking an average resulting in smoothing of jagged edges in raster images. In the context of this simulation, supersampling would involve increasing the sampling of the computation window and taking the precautions necessary to maintain accuracy.

Supersampling was implemented for several reasons. Firstly, it allows the definition of target intensity distributions with greater precision than the standard 12.5µm resolution. Secondly, a higher sampling increases the accuracy of the propagation model. Lastly, it was speculated that supersampling would improve phase retrieval performance, especially for very small target features, due to the greater amount of phase information exchanged in every iteration.

A uniform sampling distribution was used as this corresponds to creating ‘virtual’ pixels with a smaller size. Different sampling densities were defined using the supersampling factor as shown in Figure 11.

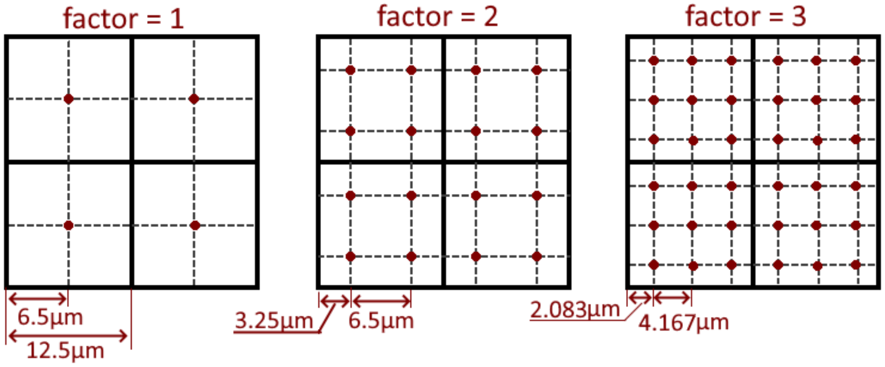


Figure 11 Sampling grids

4.6 Phase retrieval implementation

While the implemented algorithm follows the diagram described in section 3.1. Several practical modifications had to be made.

Firstly, due to the computational window being larger than the SLM, after back propagation, there is phase information in the area outside of the SLM. If the back propagated phase field was used directly, the retrieval accuracy would be artificially high due to the extra phase information[1]. To correct for this, the areas outside of the SLM were set to 0 before passing the field to the mask property.

Secondly, to ensure that supersampling didn't artificially increase the resolution of the SLM, the back propagated phase field had to be "pixelated" such that an accurate representation of each pixel was achieved. This was done by first computing the average phase value for each n x n array of values representing one pixel, and then setting the phase of each point in the pixel to the average phase. Practically this was done using cv2.resize() function with the inter_linear method, and then supersampling the resultant array.

Finally, to quantify the accuracy of the phase retrieval and enable convergence tracking, the mean square error (MSE) and the structural similarity index measure (SSIM) of the target and output intensity fields are calculated in each iteration.

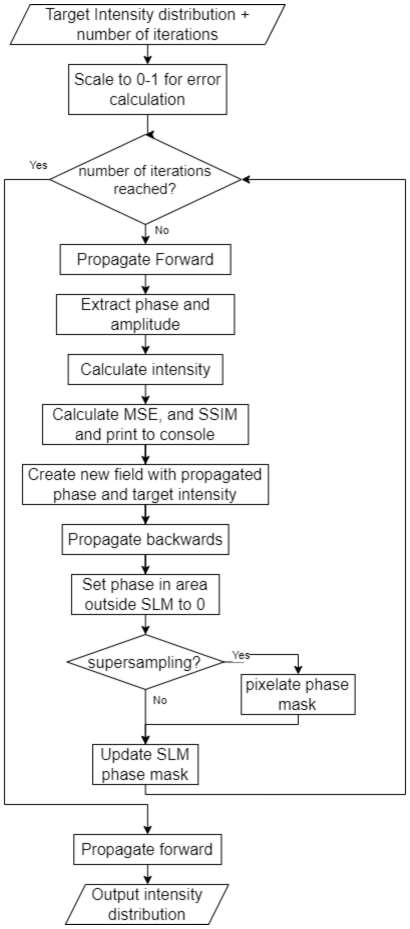

Target Intensity distribution + number of iterations
Scale to 0-1 for error calculation
number of iterations reached?
Yes
No
Propagate Forward
Extract phase and amplitude
Calculate intensity
Calculate MSE, and SSIM and print to console
Create new field with propagated phase and target intensity
Propagate backwards
Set phase in area outside SLM to 0
supersampling?
Yes
No
pixelate phase mask
Update SLM phase mask
Propagate forward
Output intensity distribution

Figure 12 phase retrieval flowchart

5.0 Results

Having developed the simulation and phase retrieval algorithm, both were tested to investigate their performance.

5.1 Simulation testing

Due to the lack of experimental data the accuracy of the simulation could not be established directly[2], however, analytical phase masks with well-defined expected outputs were simulated to gain a qualitative baring of the simulation accuracy. These masks were generated using the beam shaper support software. The results of the simulations are shown below in Figure 13.

[1] It is necessary to note, that this would only boost the retrieval accuracy if the input intensity distribution spanned outside the SLM.

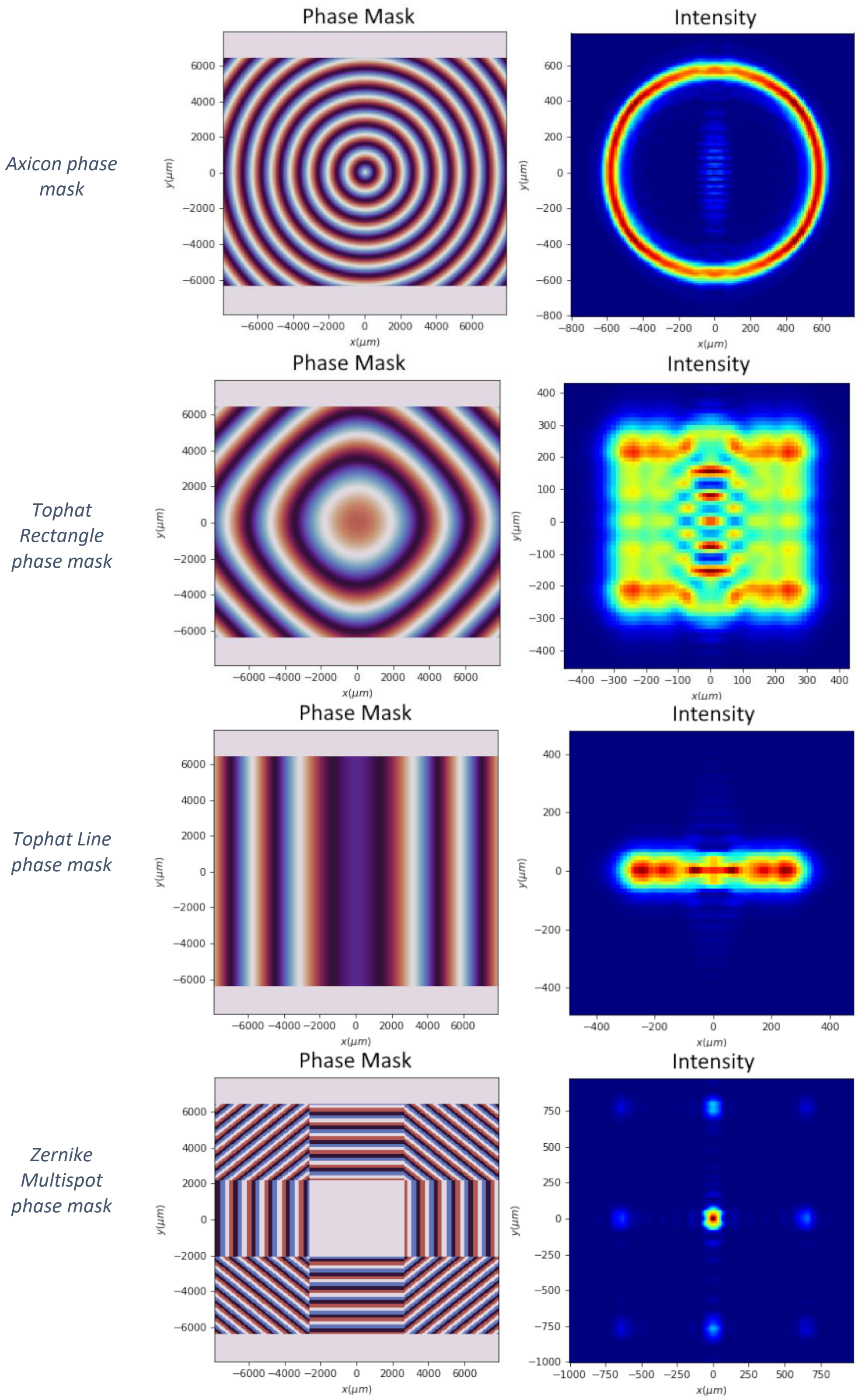


Figure 13 Analytical phase mask simulation results

These results were very promising as each phase mask produces the expected diffraction pattern with a high quality. It is necessary to note however, that the focus (just before which the diffraction patterns were captured) is located 29.5cm from lens 2 as opposed to the measured 18cm. The most likely reason for this is that the measurement is inaccurate as the bottom most part of the roughly 10cm long lens was used as the reference point for the measurement. If instead the top part was used, the expected focal length would be much closer to the simulated focal length. The alternative possibility is that the thin lens approximation is not valid, and a higher fidelity representation should be used.

The focal length of 29.5cm was verified using the lens equation as shown in eqs. (1-2).

$$\frac{1}{f} = \frac{1}{s_1} + \frac{1}{s_2} \qquad (1) \qquad\qquad s_2 = \frac{1}{\frac{1}{f}-\frac{1}{s_1}} = \frac{1}{\frac{1}{160}-\frac{1}{350}} = 294.737mm \qquad (2)$$

Supersampling

The effect of supersampling on the produced diffraction patterns was investigated. Comparisons of the intensity distribution produced by tophat line and tophat rect masks with different supersampling are shown below.

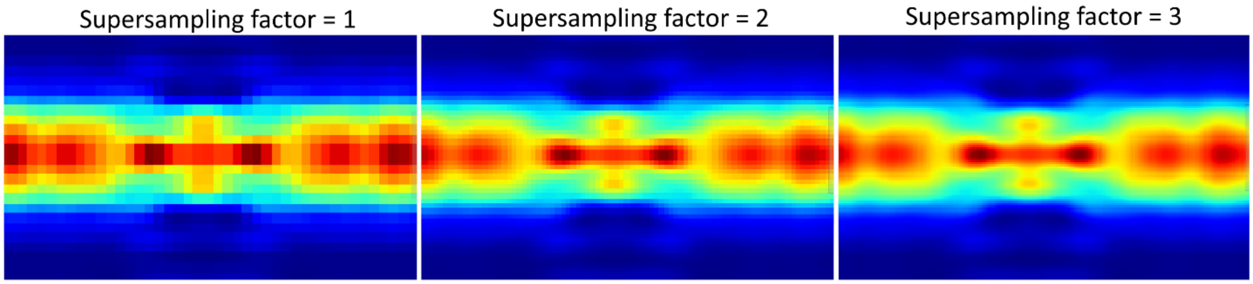


Figure 14 Close up of tophat line intensity distribution with varying sampling

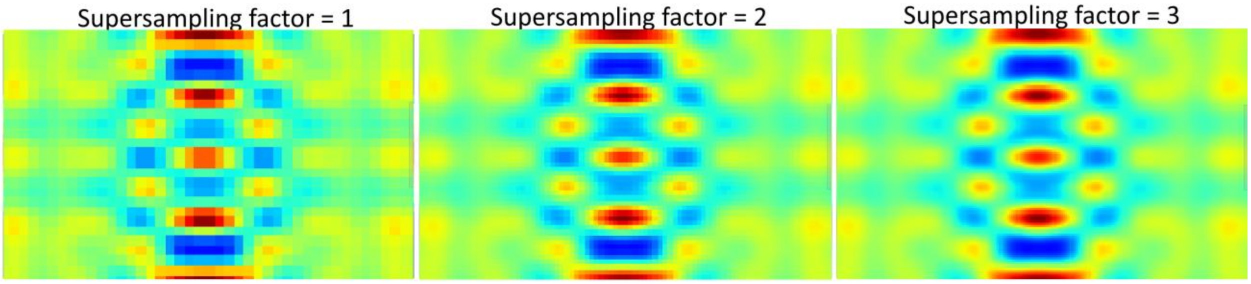
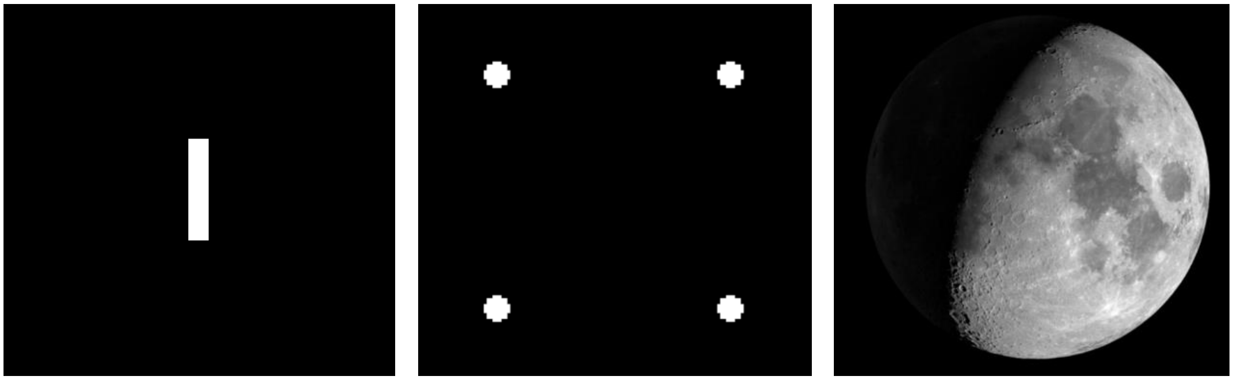


Figure 15 Close up of tophat rect intensity distribution with varying sampling

As can be seen, supersampling has the expected effect of increasing the resolution of the diffraction pattern. Furthermore, the general intensity distribution does not change which indicates that the minimal sampling size of 1024x1072 is sufficient to accurately capture the diffraction. A downside of the increase sample size is the corresponding increase in computation time. Specifically, an O(n^2) time complexity was observed which is expected.

5.2 Phase retrieval testing

The phase retrieval algorithm was tested with various test images and its performance was measured using the error metrics. Three representative trials are presented here, and the target images used are shown below.



a) 20px x 4px line *b) 2x2 dot matrix 80px separation* *c) Moon image [9]*

Figure 16 Phase retrieval test images

These images were chosen as they best demonstrate the performance of the algorithm. The 20px line corresponds to a length of 250µm which is a typical length that could be used in micromachining applications. Likewise, the 2x2 dot matrix was selected to test whether the algorithm can generate multispot masks that are used in multibeam machining. Finally, the moon image was used to test the algorithm's ability to capture many small features in a large image. The resulting phase masks and produced intensity distributions are shown in Figures 18, 20, and 22.

Due to the small magnitude of the MSE and SSIM metrics and the small relative change throughout the phase retrieval process, they were modified for better visualization. This was done by taking the logarithm of both. Furthermore, for consistency, the SSIM was subtracted from 1 so that an SSIM of 0 represented a perfect reconstruction like with MSE.

As shown in Figure 17, the line target exhibits a very good phase retrieval result with the intensity distribution closely matching the target. The length of the intensity distribution is also 250µm showing that scale is accurately captured. Of note is the fact that a relatively high number of iterations was required to achieve a good visual result, this was the case with all target distributions that were smaller than 500µm.

The MSE and SSIM error shown in Figure 18 have very similar convergence characteristics with a several order of magnitude improvement in the first iteration. A slow improvement is then seen over the rest of the iterations. The sudden decrease in error around the 22nd iteration is unusual and was not observed in other trials.

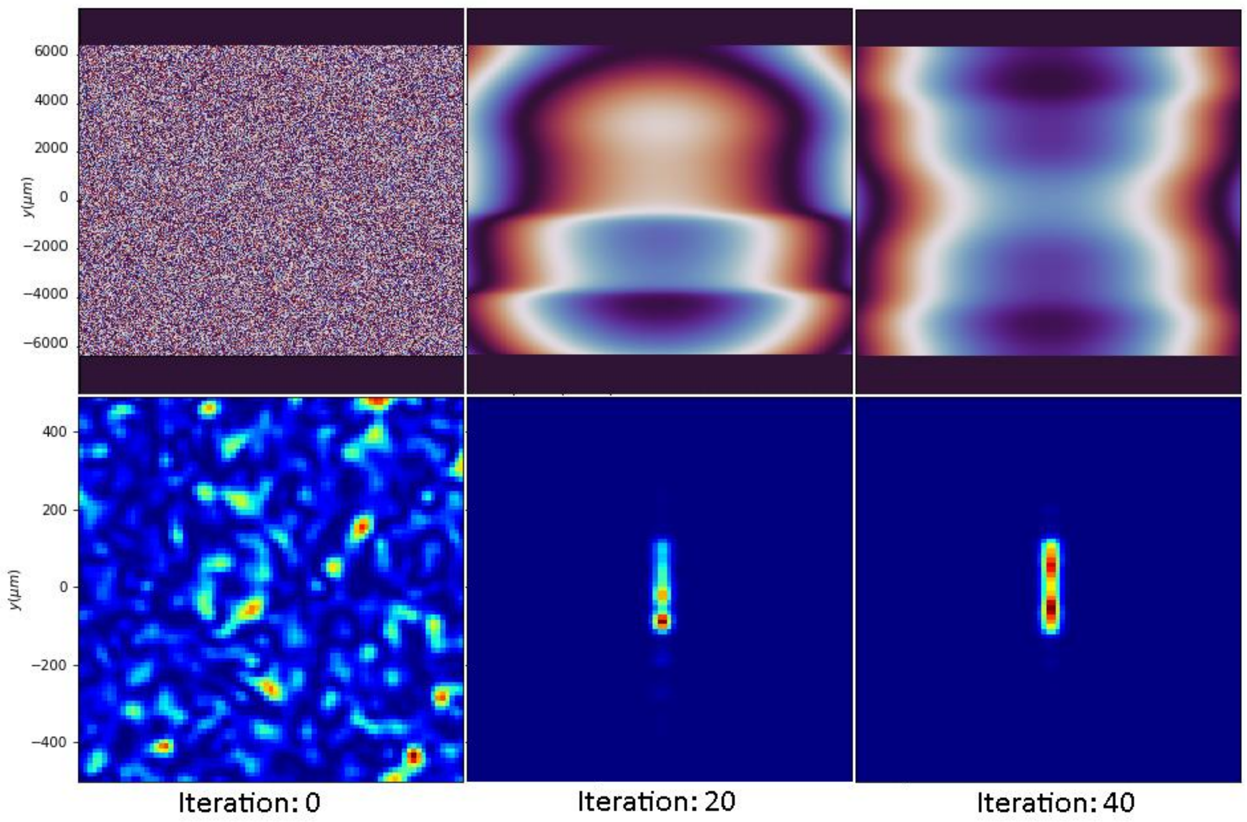


Figure 17 20px Line phase retrieval

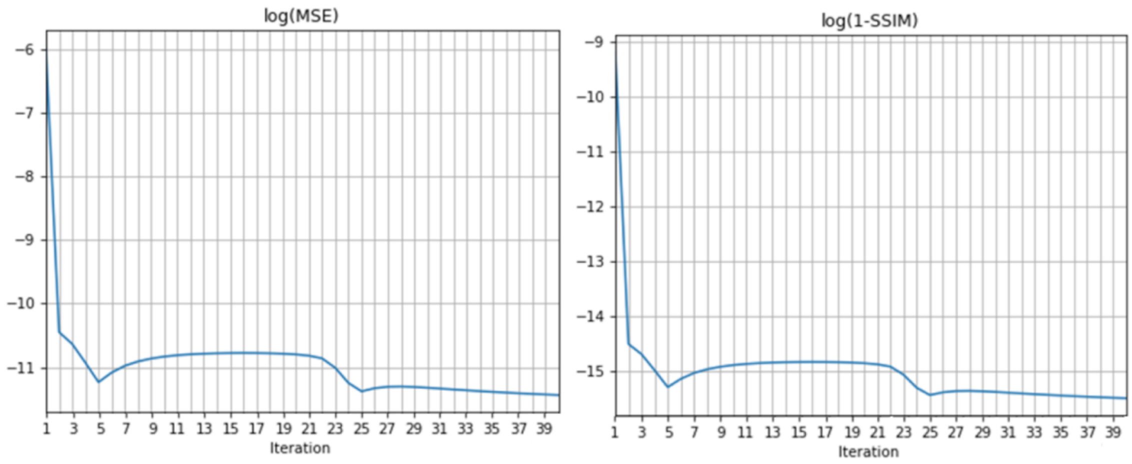


Figure 18 20px Line phase retrieval error metrics

The 2x2 dot matrix also shows very good phase retrieval with an almost perfect result at just 20 iterations. There is little noticeable difference between the intensity distribution at 20 and 40 iterations showing that the algorithm has plateaued. This is further demonstrated in the error plots where a plateau is reached at around 20 iterations. As with the line target, the error decreases very sharply in the first iteration.

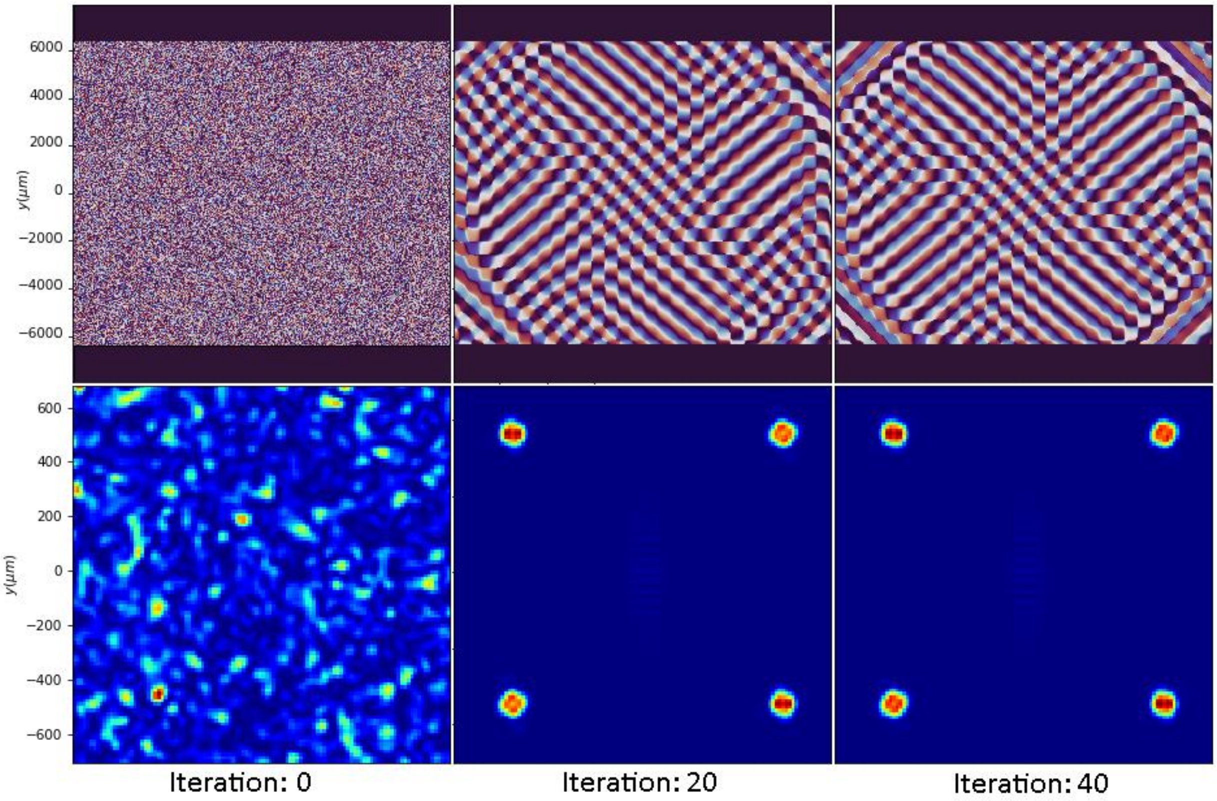


Figure 19 2x2 dot matrix phase retrieval

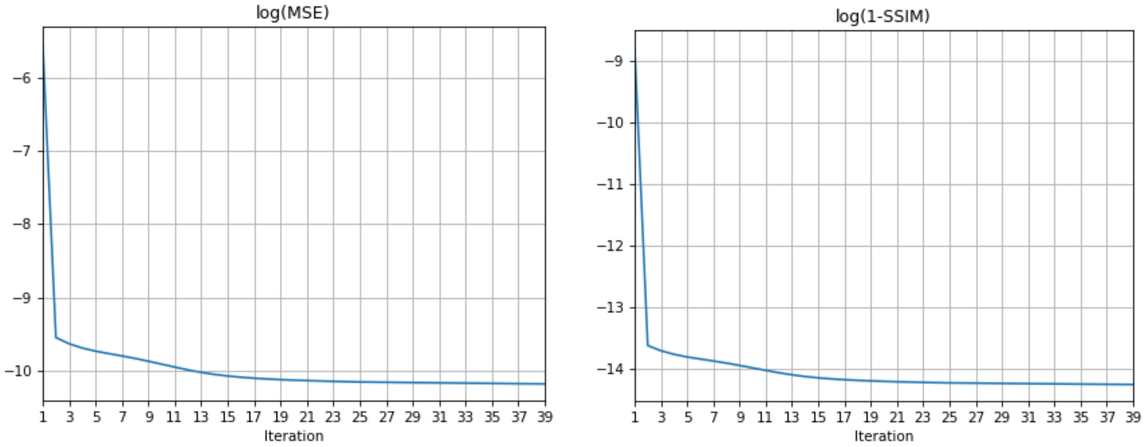


Figure 20 2x2 dot matrix phase retrieval error metrics

The Moon image is also successfully generated with the retrieved phase mask; however, a moderate amount of noise is present. Furthermore, there is a high intensity spot in the center of the image corresponding to the zero-order diffraction mode of the laser beam. This is commonly encountered with phase masks in practice and is usually corrected by physically filtering the zeroth order. Furthermore, as with the 2x2 matrix, little visual improvement in the intensity distribution is seen after the 20th iteration which is supported by the error plots. An unexpected result is that the log(1-SSIM) is lowest at the first iteration, and a similar local minima is seen in the log(MSE) plot. The error seemingly increases with the second iteration after which an expected gradual improvement is seen. The result after one iteration is shown in Figure 23 and discussed further.

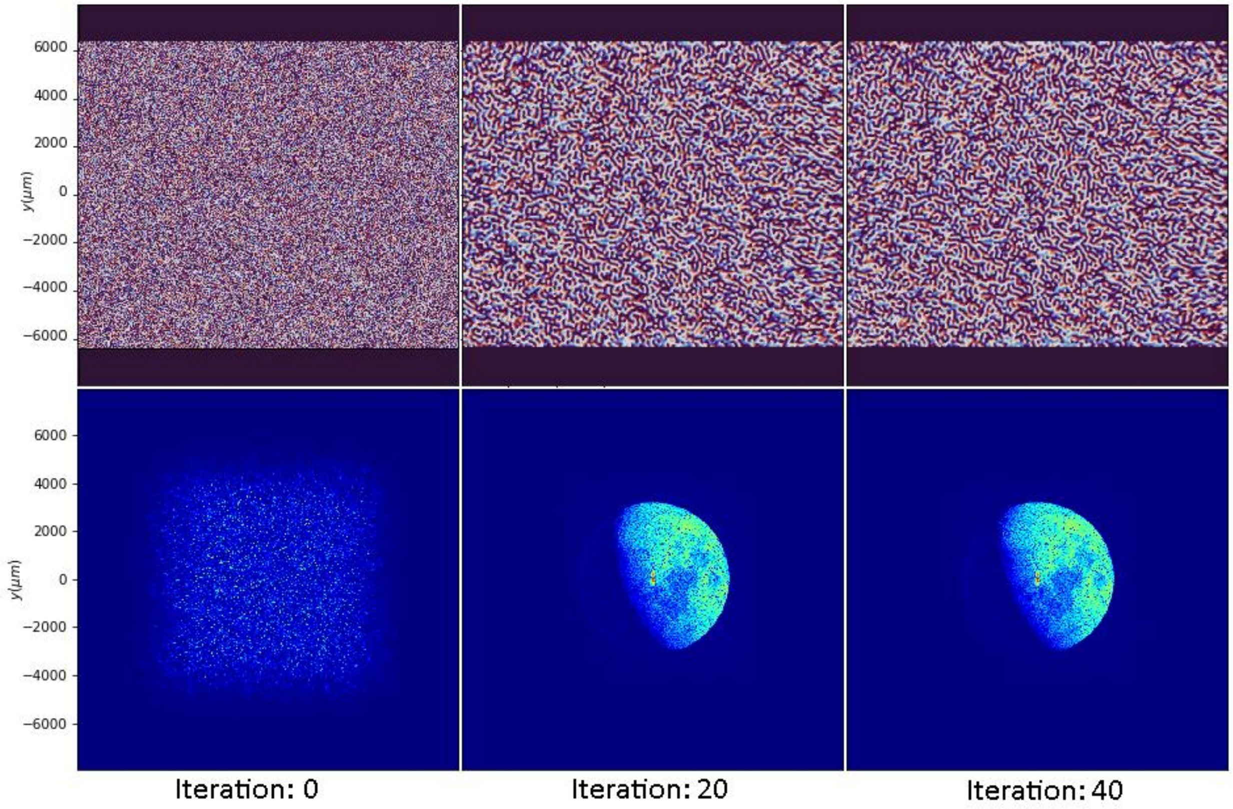


Figure 21 Moon image phase retrieval

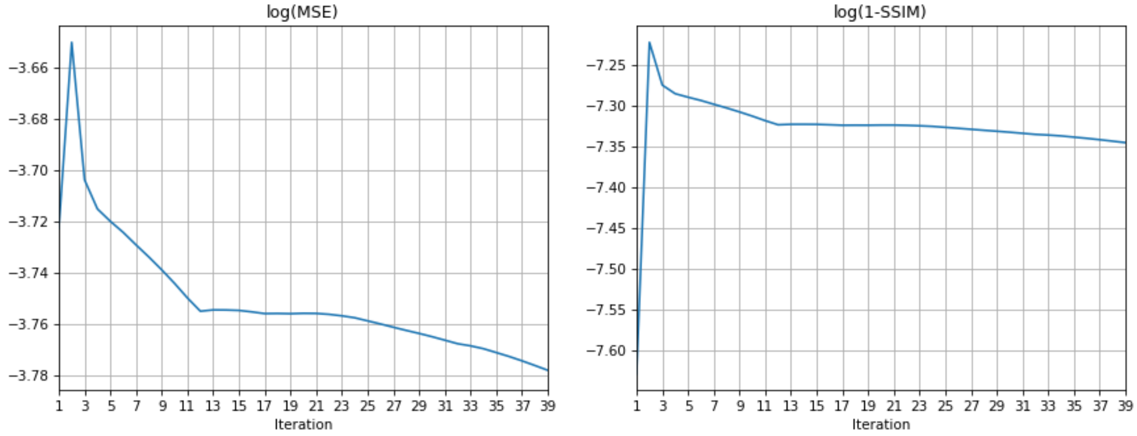


Figure 22 Moon image phase retrieval error metrics

As can be seen in Figure 23, the moon image is captured moderately well after only 1 iteration, however a lot of noise is present. Visually, the image reconstruction is worse than the results after 20 and 40 iterations which conflicts with the lower numerical error. This seems to suggest that the MSE and SSIM measures do not accurately capture the image reconstruction error.

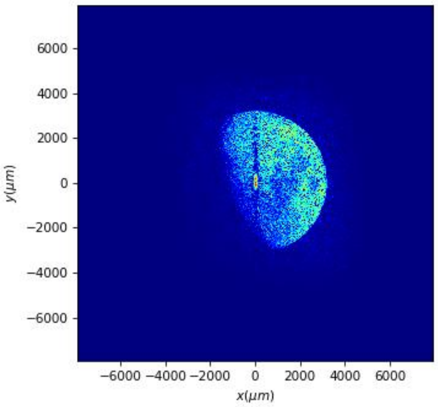


Figure 23 Moon intensity distribution after 1 iteration

Overall, the phase retrieval algorithm performs very well in all test cases as retrieved phase masks produced intensity distributions that closely matched the specified target distributions.

Supersampling

The effect of Supersampling on phase retrieval performance was briefly investigated. The 20px line target was selected for these trials as it was the most demanding for the retrieval algorithm. The PR algorithm was run for 10 iterations and the resulting intensity are shown below in Figure 24.

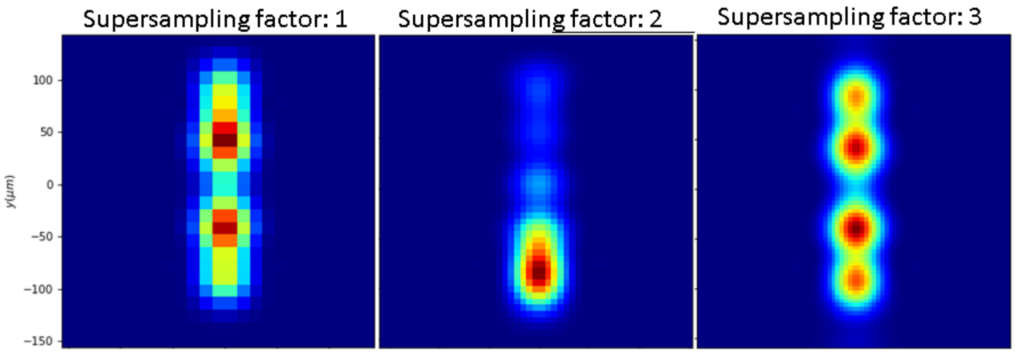


Figure 24 supersampling effect on phase retrieval

As can be seen, the increased sampling decreases retrieval performance. This is unexpected, however further trials should be conducted to verify that this is the case at higher iterations. It is important to note that the increased sampling significantly slowed down the retrieval process.

5.3 Physical testing

Unfortunately, due to time constraints, only a single set of experiments was conducted. These experiments showed that retrieved phase masks produced the desired target shapes, however the size of the shapes did not match that of the simulated intensity distributions. This is an overall positive result as it indicates that the simulation and phase retrieval algorithm are mostly correct, however the simulation will have to be tuned.

6.0 Conclusions

In conclusion, a simulation of the beam shaping system was successfully developed in python using the Rayleigh-Sommerfield diffraction method to model wave propagation. A supersampling method was implemented which successfully increased the resolution of the simulation. The simulation was then evaluated qualitatively and seemingly showed high accuracy, however, the real accuracy could not be established due to a lack of experimental data.

The simulation was then implemented in the Gerchberg-Saxton phase retrieval algorithm and its performance was tested. The modified algorithm performed well in all test cases as generated phase masks for all target intensity distributions produced matching simulated intensity distributions. The effect of supersampling on phase retrieval performance was investigated, however contrary to expectations supersampling decreased the effectiveness of the algorithm.

Limited experiments were then conducted on the physical system. These showed that retrieved phase masks produced the desired target shapes, however their size did not match that of the simulated intensity distributions.

7.0 Recommendations

Before further development of the phase retrieval algorithm, it is essential that the simulation is experimentally verified. This could be done by simulating a set of test phase masks and testing the same phase masks in the laser system and then comparing the resulting intensity distributions. A set of Tophat Line masks with different width settings would be recommended for tuning the simulation as it provides an easily measurable length that can be compared with the simulation. More complex masks could then be used to check the broader accuracy of the simulation. This process should be repeated to verify that the observed phase retrieval results match reality.

The main tuning parameter of the simulation will likely be the distance between L1 and L2 as this has a direct impact on the scale of the diffraction patterns. With the current limited test results it is assumed that a simple scale correction is all that is required to tune the simulation, however it is possible that more inaccuracies exist. If that is the case a more thorough analysis of the model and system will be required, and perhaps more aspects of the system will have to be modelled (such as the imperfect mirrors M1 and M3).

There are several ways that the phase retrieval algorithm could be developed further to achieve greater performance. Firstly, the effect of different initial phase masks on convergence could be investigated as a large degree of variation between trial runs was observed. Secondly, better error metrics should be developed to ensure an accurate measure of retrieval performance as the MSE and SSIM metrics did not correspond perfectly to visual accuracy of the retrieved image. Finally, the software implementation should be optimized to decrease the time required per computation so that more iterations of phase retrieval can be done.

An interesting avenue of future research would be investigating the real time application of the retrieval algorithm within the laser system to correct for errors and aberrations with a feedback control approach. This would be a tremendously powerful feature if implemented successfully. Additionally, the use of conventional optimization techniques to improve the retrieved phase masks could result in meaningful improvements.

8.0 References

[1] Y. C. Tieyu Zhao, "Modified Gerchberg-Saxton (G-S) Algorithm and Its Application," *Entropy,* vol. 22(12), p. 1354, 2020.

[2] R. W. G. a. W. O. Saxton, "Gerchberg, R. W.; Saxton, W. O. (1972). "A practical algorithm for the determination of the phase from image and diffraction plane pictures," *Optik,* vol. 35, p. 237–246., 1972.

[3] Koogid, *Gerchberg-Saxton algorithm for iterative phase retrieval, FT is Fourier transform,* License: CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=99253324, Accessed: 01.10.2024.

[4] C. M. K. v. d. W. S. e. a. Harris, "Array programming with NumPy," *Nature,* vol. 585, p. 357–362, 2020.

[5] J. W. Goodman, Introduction to Fourier Optics, Roberts and Company Publishers, 2005.

[6] A. A. J. P. U. Sander Konijnenberg, "BSC Optics," *TU Delft Open,* 2021.

[7] L. M. S. Brea, "Python diffraction and interference, (Version 0.2.4), [Documentation]," [Online]. Available: https://diffractio.readthedocs.io/en/latest/.

[8] F. &. W. A. Shen, "Fast-Fourier-transform based numerical integration method for the Rayleigh-Sommerfeld diffraction formula," *Applied Optics,* no. 45, p. 1102–1110, 2006.

[9] N. P. D. L. Ernie Wright, "NASA's Scientific Visualization Studio," [Online]. Available: https://svs.gsfc.nasa.gov/5048/. [Accessed 16 01 2024].