Imperial College
London

Final Report for MEng individual project

MATLAB ALGORITHMS FOR HOLOGRAPHIC STIMULATION IN OPTOGENETICS

Author name:
Yann Wei-Yan Zhong
CID: 01098279
Imperial College London, Department of Bioengineering
Supervisor:
Dr Christopher Rowlands

Submitted in partial fulfilment of the requirements for the award of BEng/MEng in Biomedical Engineering from Imperial College London

18th of June 2019

Word Count: 5363

Abstract

Optogenetics, the field of using light in controlling neuronal response, is an exciting field which could offer alternative, more effective solutions in many neurological and medical applications. It is currently limited in its efficiency, however, by a combination of low temporal and spatial resolution, as solving one often compromises the other. To this end, CGH (Computer Generated Holography) offers an alternative in high speed projections of light sources to allow for targeted volumetric stimulation of neural circuits. First a discussion into the backgrounds of the relevant fields will be presented, followed by examination of past implementations. Finally, methods and results of the Gerchberg-Saxton algorithm and its variants, a popular phase retrieval algorithm used in hologram generation, are presented here. Further discussion into future work and alternatives are also touched upon.

Acknowledgements

I would like to thank my supervisor, Christopher Rowlands, for his continued assistance in the development of this project and for the Bicycle cards.

Contents

1 Introduction **4**
1.1 Background on optogenetics 4
1.2 Background on holography 5
1.3 Past implementations and the problem at hand 8
1.4 Aim of the project 10

2 Theory **12**
2.1 Fourier theory 12

3 Methods **14**
3.1 2D GS, methods 14
3.2 DMD and SLM simulation, methods 17
3.3 3D GS with Fresnel diffraction, methods 18

4 Results **21**
4.1 First iteration of normal 2D GS, results 21
4.2 DMD and SLM simulation, results 22
4.3 3D GS with Fresnel diffraction, results 23

5 Discussion **25**
5.1 First iteration of normal 2D GS, discussion 25
5.2 DMD and SLM simulation, discussion 25
5.3 3D GS with Fresnel diffraction, discussion 25
5.4 Further work and alternatives considered 26

6 Conclusion **27**

References **28**

Appendix A 2D GS code **31**

Appendix B 3D GS code **33**

Appendix C DMD and SLM modifications **36**

Appendix D Code for Fresnel propagation **37**

1 Introduction

The introduction will be split into a background on both topics relevant to the project: optogenetics, then holography, followed by a section of prior methods used, the problem at hand, and finally, the objectives of this project.

1.1 Background on optogenetics

Optogenetics: opto from Greek, *optikós*, meaning visible, is a recent field born from the hybridization of optical technologies and molecular genetics. It revolves around controlling pre-encoded cells, usually neurons, through the triggering of neuronal excitation or inhibition, with light as a source, making use of light sensitive proteins known as opsins. This 2015 "Method of the Year" is attractive in both its temporal and spatial accuracy [1] and has already proved to successfully restore vision in non-human species such as mice [2] and macaques [3]. It also has future potential in effectively treating aliments and conditions such as depression and spinal cord injuries [4], but currently still faces many challenges and obstacles to be overcome.

Historically, neurons have been stimulated through the direct use of stimulation electrodes - which come with a number of limitations: lack of accuracy in terms of area stimulated, invasive procedure, not to mention keeping the electrode itself in place [5]. To this end, light as a stimulation method would seem to solve many of our problems thanks to both its spatial and temporal resolution, while also being less of a burden on the physical setup side of things. The idea of using light as a switch for human and animal neurons was discussed as early as 1999 by a co-discoverer of the DNA molecule, Francis Crick, which at the time seemed an idea "far-fetched but conceivable [6]".

Since then, numerous advances have been made in the direction of Crick's vision, with early work by Miesenbock et al. in 2002 for their creation of chARGe, allowing the stimulation of generalist vertebrate neurons with light [7]. More recently, Deisseroth et al. pioneered the use of optogenetics as a technique and gave it its name in 2009 for their work in the stimulation of Parvalbumin neurons in mice [8], but not before their adaptation of channelrhodopsin2 (ChR2) as a light gated cation channel in 2005. Now a vastly used opsin in optogenetics, ChR2 is usually inserted via viral vectors and allows for the release of charged ions when exposed to a certain wavelength

of light, in this case green or blue light at a wavelength of roughly 470 nm [1]. Since then, many new types of opsins have been both discovered and cloned from existing ones to react to different wavelengths of light or to allow for different functions such as excitation and/or inhibition as shown in figure 1 and figure 2. As such, optogenetics can be considered an extremely promising field in relative infancy which this project hopes to improve upon with work in holography.

Table 1. Single-Component Optogenetic Tools with Both Spectral and Kinetic Data Published

Opsin	Mechanism	Peak Activation λ	Off Kinetics (τ, ms)*	Kinetics References
Blue/Green Fast Excitatory				
ChR2	Cation channel	470 nm	~10 ms	Boyden et al., 2005; Nagel et al., 2003
ChR2(H134R)	Cation channel	470 nm	18 ms	Nagel et al., 2005; Gradinaru et al., 2007
ChR2 (T159C)	Cation channel	470 nm	26 ms	Berndt et al., 2011
ChR2 (L132C)	Cation channel	474 nm	16 ms*	Kleinlogel et al., 2011
ChETAs: ChR2(E123A) ChR2(E123T) ChR2(E123T/T159C)	Cation channel	470 nm (E123A) 490 nm (E123T)	4 ms (E123A) 4.4 ms (E123T) 8 ms (E123T/T159C)	Gunaydin et al., 2010; Berndt et al., 2011
ChIEF	Cation channel	450 nm	~10 ms	Lin et al., 2009
ChRGR	Cation channel	505 nm	4-5 ms* (8-10ms)	Wang et al., 2009; Wen et al., 2010

Figure 1: Blue/Green fast excitatory opsins and activation wavelengths[9]

Yellow/Red Inhibitory				
eNpHR3.0	Chloride pump	590 nm	4.2 ms	Gradinaru et al., 2010
Green/Yellow Inhibitory*				
Arch/ArchT	Proton pump	566 nm	9 ms	Chow et al., 2010
eBR	Proton pump	540 nm	19 ms	Gradinaru et al., 2010

Figure 2: Yellow/Red/Green inhibitory opsins and activation wavelengths[9]

1.2 Background on holography

Holography goes back as far as 1947 - invented then by Dennis Gabor as an effort to improve electron microscopy resolution [10], it was in 1962, with the advent of the laser light (LASER: Light Amplification by Stimulated Emission of Radiation), that the first optical holograms were implemented in

the United States and the Soviet Union roughly simultaneously. More likely than not due to pop culture references such as the princess Leia "hologram" from the first Star Wars movie, holograms are often confused for volumetric images, the latter being three-dimensional images that can be viewed from any angle. For the purpose of this report, all subsequent discussions on holograms refer to its traditional definition, which should be dissociated from that of the three-dimensional volumetric image projection, which is another field of research and development in of itself [11].

At its core, a basic optical hologram is generated as a result of splitting a coherent light source (such as a laser beam) through a beamsplitter in order to get an illumination beam and a reference beam. The former goes through the target object and joins the latter through mirror arrangements, their combined illumination hitting a photographic plate in order to create an interference pattern. An interference pattern is a result of two waves (usually, light) travelling in the same medium and meeting at one point to either constructively (additively) or destructively (subtractively) sum up, i.e. interfere. This seemingly random looking pattern (figure 3) allows for the reconstruction of the original object that was hit by the illumination beam, when illuminated by the original coherent light source (figure 4). In fact, it is not uncommon to refer to both the interference pattern and the reconstructed object as holograms.

Recent advances in computational power have made the use of CGH (Computer Generated Holography) more and more popular, bypassing the conventional need for a dual beam and plate setup in order to create interference patterns [12], instead relying on established algorithms in order to create custom 3D intensity patterns that can then be printed onto masks and/or films, or diffracted through optical elements such as SLM (Spatial Light Modulators) or DMD (Digital Micromirror Devices) that impose phase or intensity modulation onto an input beam's wavefront. Such algorithms include PLS (Point Light Source) algorithms, Multi-view algorithms, or phase retrieval algorithms such as the GS (Gerchberg-Saxton) algorithm [13], which is the focus of this report.

The use of CGH proves to be an attractive method in controlling neuronal excitation and inhibition for optogenetics for its ability to manipulate light in a volume. We would like for large numbers of neurons in specific planes to

be targeted individually with a high temporal resolution, without triggering excitations and responses from planes lying directly above or below them. This can partly be achieved through CGH, as interference patterns that do not have to be based on real objects can be calculated and diffracted through different optical tools, which is elaborated on in latter sections.

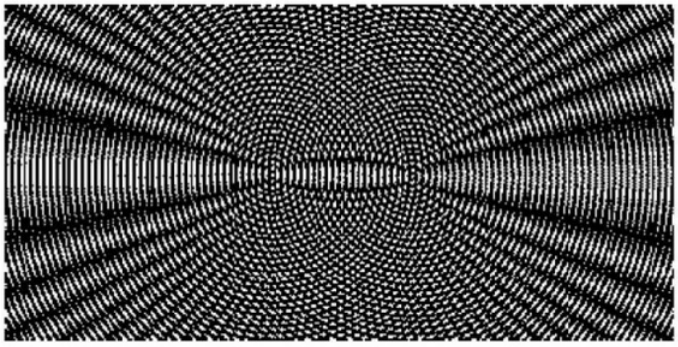

Figure 3: Interference pattern from illumination and reference beam [14]

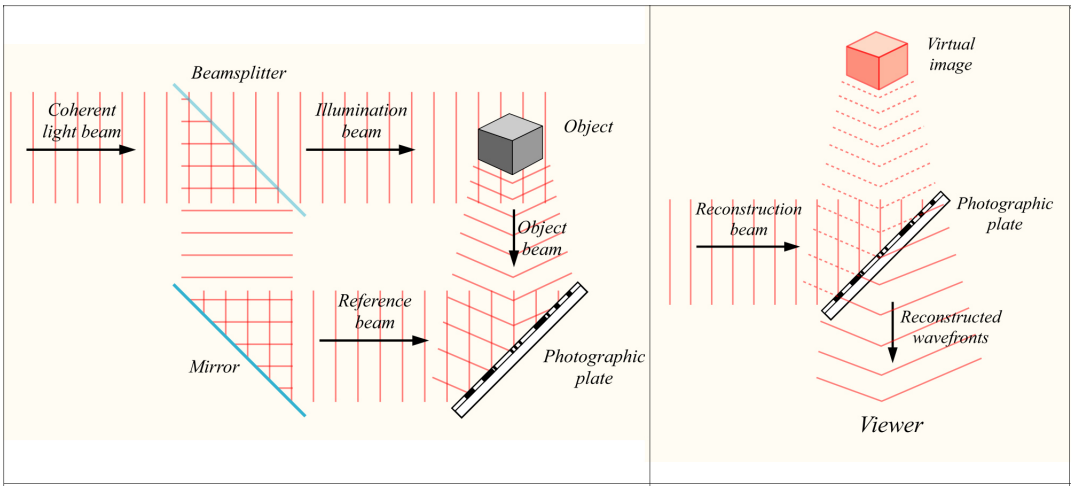

Figure 4: Creation and reconstruction of a simple object hologram [15]

1.3 Past implementations and the problem at hand

Optogenetics through single-photon illumination has historically been achieved with great temporal resolution. Boyden et al., in their original paper, illuminate ChR2-positive neurons with blue light to successfully induce rapid depolarizing currents within 50 microseconds [1], achieving high temporal resolution. However, little to no spatial resolution and depth is achieved. This is largely due to the fact that brain tissue will inevitably scatter light, greatly degrading spatial resolution. The underlying problem for single-photon illumination traditionally lies in achieving the desired spatial accuracy. Excitation is typically not able to be confined to a single plane at a z-plane perpendicular to the axis of illumination [16]. Even at single cell resolution, light must cross planes of neurons in order to arrive at the desired plane, inevitably influencing responses from the planes crossed.

A popular alternative to single-photon illumination is two-photon excitation, which makes use of two excitation wavelengths to allow for much more targeted illumination (see figure by Yang et al. at figure 5) but is currently limited to the parallel stimulation of less than a hundred neurons at once due to power constraints. First shown in 2009 by Rickgauer and Tank, two-photon excitation of ChR2 is possible through two-photon laser-scanning microscopy (TPLSM), by continously scanning the cell soma along a spiral trajectory for about 30 ms, giving spatially localized fluorescence excitation in deep scattering tissue, as well as increased depth penetration [16].

The non-linearity of two-photon approaches with regards to excitation from light allows for excitation confined to much more accurate areas, but this is at the expense of lowered temporal resolution due to rapid deactivation of ChR2, as outlined by Pegard et al., and the much higher time taken for two-photon scanning approaches, as opposed to faster parallel approaches. Yang et al. improve upon temporal resolution by making use of a novel hybrid scanning approach to stimulate more than 80 neurons with sub 10 ms latencies in a three dimensional volume while imaging surrounding neuronal activity in layer 2/3 of the mouse visual cortex [17] . Their approach has a relatively low power requirement as one major drawback to lasers as illumination beams is the possibility of neuronal photodamage (too much power fed into target area). Pegard et al., on the other hand, present a new excitation method through three-dimensional Scanless Holographic Optogenetics

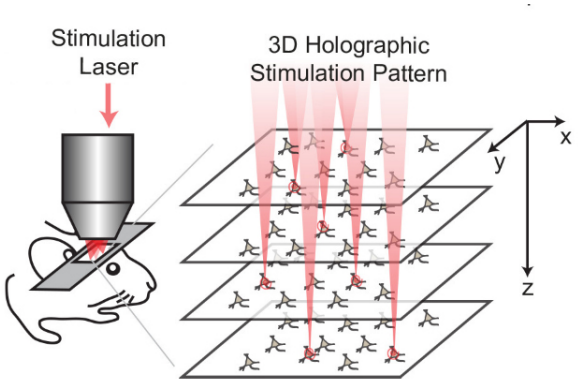

Figure 5: Illustration of volumetric illumination of mouse neurons with two-photon illumination

with Temporal focusing (3D-SHOT), which allows simultaneous illumination of 21 neurons in a 3D volume. They combine CGH with temporal focusing, a method relying on a diffraction grating that spatially separates spectral components of pulses into a "rainbow beam" and recombines these components only at the spatial focus of the objective lens [18]. Up to 300-600 neurons per SLM frame illumination is predicted to be achievable, but this is unrealistic due to the physical amount of power that can be shone into a mouse's brain, which follows the same constraints of neuronal photodamage stated above.

A thought can also be spared to alternative methods of neuronmodulation, which is the alteration of nerve activity through targeted delivery of a stimulus. Prior to optogenetics, the prime method for neuromodulation was known as Deep Brain Stimulation (DBS), an invasive procedure where a neurostimulator module is implanted inside the brain to deliver electrical signals to the brain [19]. DBS has been used to help treat Parkinson's disease [20], epilepsy [21], amongst others. DBS, however, lacks the spatial resolution that optogenetics offers [22]. Another area of interest lies in Brain-machine interfaces (BMI). Rather than acting directly on neurons through stimuli, the brain is connected to and communicates with a computer system. BMI are more commonly used in order to provide motor assistance to disabled patients. BMI differs from optogenetics and DBS in that it can provide bidirectional communication such as by giving proprioreception with the movements of a prosthetic arm, for better control [23].

1.4 Aim of the project

As such, the scope of this project revolves around creating an illumination source that isn't affected by power constraints or lowered temporal resolution like those imposed by two-photon excitation, while also accomplishing simultaneous single plane illumination of thousands of neurons in the field of optogenetics. This is possible with single-photon illumination by diffracting an input beam in such a way as to result in speckle and noise in non-targeted planes (non constructive illumination), whereas targeted planes would receive a proper distribution. Then, when generated at a specific frequency (high speed holographic projection), high resolution and targeted planar excitation/inhibition would be accomplished, allowing the creation of specific depolarization currents. This is due to the non-linearity of a neuron's response to light as action potentials are triggered not just by illumination intensity (i.e., how much light is received), but also by the way the intensity is distributed (i.e., in what "pattern" light is received). This means that even though we cannot avoid shining light onto a volume of neurons, by ensuring that non-targeted neurons receive degraded light distribution, we are effectively bypassing the problem posed by full field illumination. This is illustrated in figure 6.

Overall, this is achievable by work surrounding the Gerchberg-Saxton algorithm, which makes up the bulk of this project. As will be elaborated on, the 3D version of the GS algorithm allows the creation of holograms that are capable of causing one light source to reconstruct different images (and by extension, illumination patterns) at different z depths, or stacks, by treating a 3D target object as multiple 2D image "slices". First a rundown of basic Fourier image theory will be presented, followed by a breakdown of the basic Gerchberg-Saxton algorithm for one plane. An extension into 3D GS will then be elaborated on, which will lead into a discussion about differing methods that can be expected to achieve desired results.

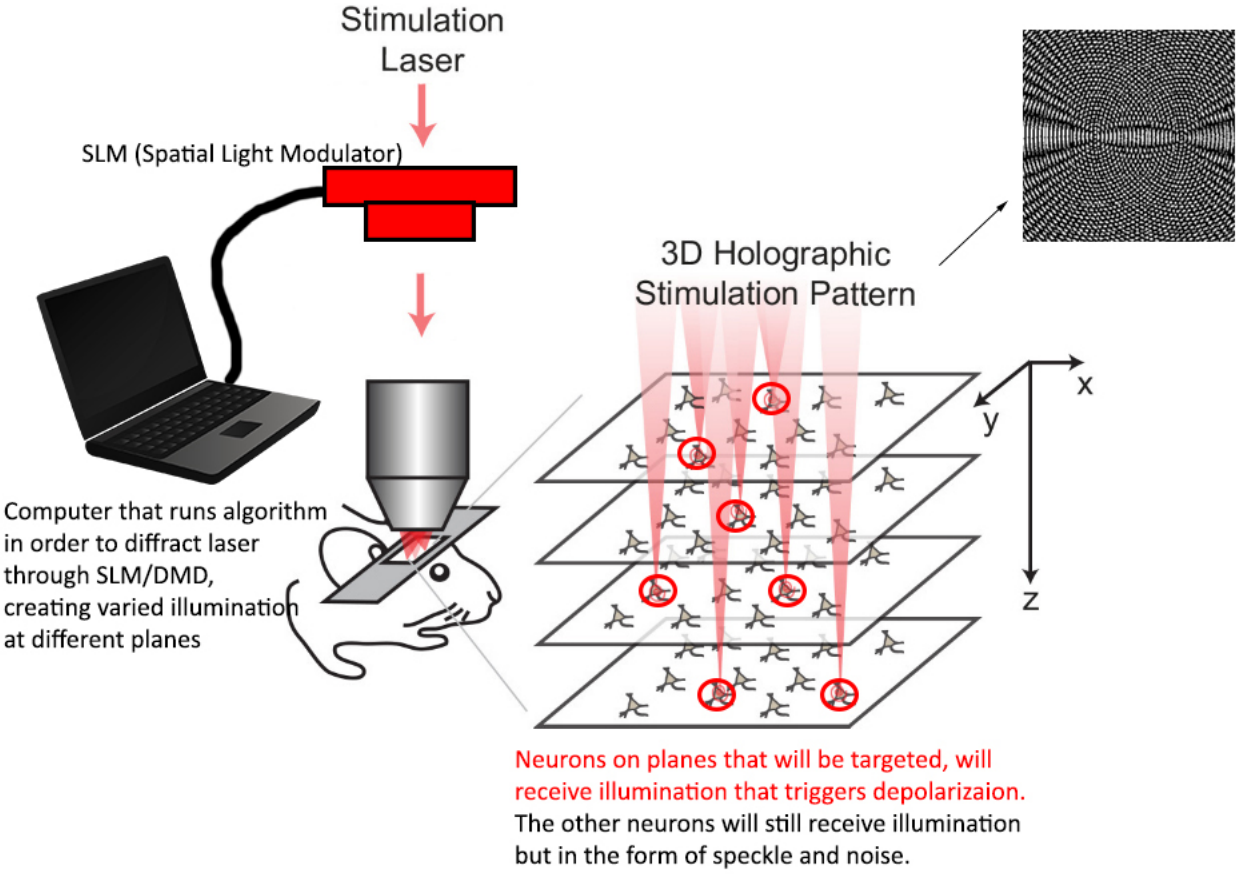

Figure 6: Diagram of desired setup. First the algorithm running on a computer will calculate a hologram able to allow for diffraction of a specific pattern by the SLM or other DOEs (Diffractive Optical Elements), after which the stimulation laser goes through a mouse that has undergone craniotomy. Stimulated at a specific frequency, the targeted neurons circled in red will be triggered due to constructive illumination but the non-targeted neurons will not be triggered due to destructive illumination and/or speckle and noise. Diagram adapted from Yang et al. paper [17].

2 Theory

2.1 Fourier theory

Before proceeding into algorithm methods used, some necessary background needs to be provided regarding Fourier image theory. In signal processing, Fourier theory allows for any signal that is a function of time to be decomposed into core sinusoidal components with varying frequencies and amplitudes. Similarly, a 2D image is composed of pixels in the x and y dimension that are a function of pixel intensity, so the Fourier transform allows for similar decomposition into the frequency/Fourier domain. Once in the Fourier domain, we may extract both phase and intensity information of the image's frequency components to allow for varying degrees of image reconstruction, which is what allows the most basic version of the phase hologram GS algorithm to work. For example, let us consider that the amplitude after Fourier Transform of the following two images (Pikachu and Pikachu without a tail) in figure 7 can be split into their intensity and phase components as according to the relation:

$$A = Ie^{j\varphi} \tag{1}$$

Where A is amplitude, I is intensity (complex magnitude) given by $abs()$, φ is phase (complex phase) given by $angle()$, e is the exponential operator and j is the imaginary number.

If neither intensity nor phase is varied and we simply take the inverse Fourier transform (IFT) of the amplitude, assuming no loss of information, we would get a perfect reconstruction of both images. By taking the inverse Fourier transform (IFT) of only the phase term, we get a partial reconstruction of largely positional information, where the tail is obviously missing from the reconstruction of the second Pikachu (figure 8). We may even combine the intensity information of one with the phase information of the other to get a close reconstruction of the first, and an approximation of the second one with an additional Pikachu tail (figure 9), albeit with a notable loss of color information. Note that reconstruction via intensity information only is only possible if the original image is perfectly symmetric around the center point, as asymmetry (and most positional information) is contained in the phase term, which is not the case in these two images.

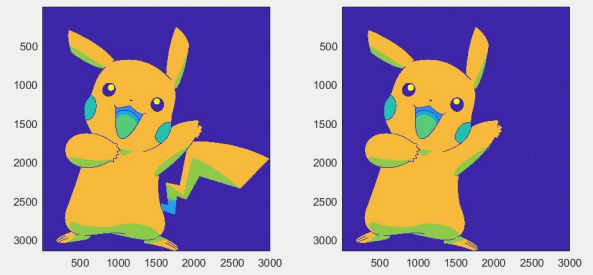

Figure 7: Original input images of Pikachu and Pikachu without tail

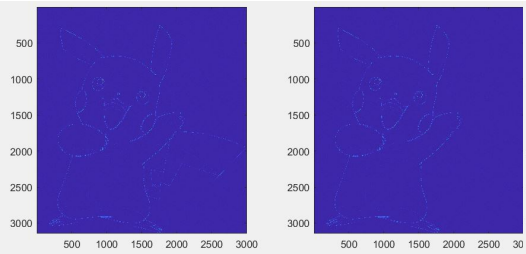

Figure 8: Phase only reconstruction

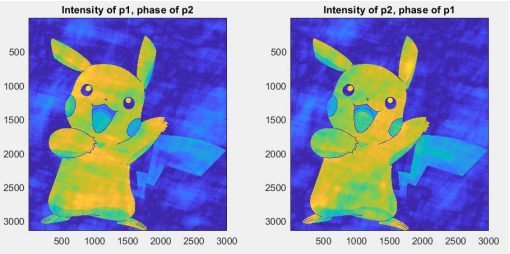

Figure 9: Mixed reconstruction

This is relevant as the GS algorithm is rooted in Fourier image theory - for example the most basic version of GS, elaborated on in section 3.1, is considered a phase modulated algorithm as it reconstructs input images through iterative loops where phase information in the Fourier plane is updated, but intensity information is discarded. This opens up many possibilities for the creation of weighted GS, intensity modulated GS, 3D GS, simultaneous phase and intensity modulation, and so on, which will be discussed.

3 Methods

The methods section will be split into mainly 2D GS and 3D GS methods, with an in-between section on DMD and SLM simulation. The results and discussions relating to the GS algorithm variants will respectively be covered in sections 4 and 5. All code used in methods is included in the appendix.

3.1 2D GS, methods

Work initially begun from an existing Gerchberg-Saxton algorithm by Musa Aydin [24], which follows the exact pseudo code outlined by that of the Wikipedia article (figure 10).

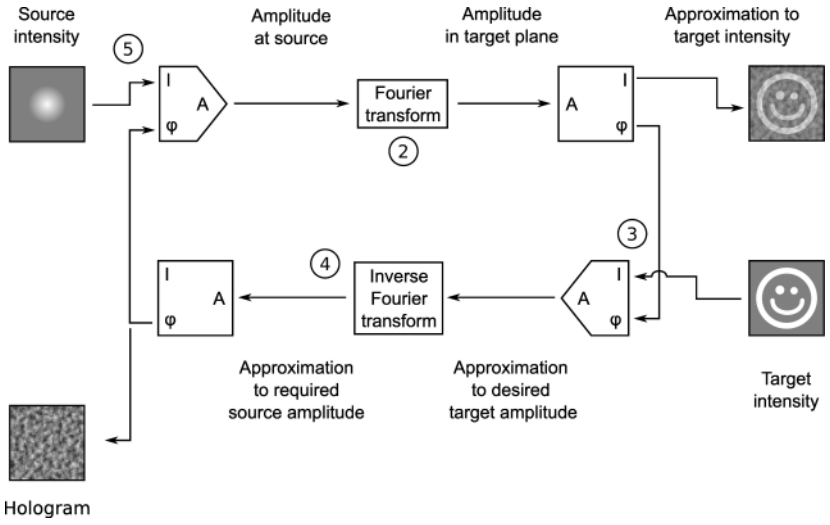

Figure 10: Block diagram for basic Gerchberg-Saxton algorithm [25]

In their original paper, Gerchberg and Saxton state that if intensity if known for 2 coherent light distributions, their phase distribution can be found by running an iterative algorithm with an error criterion. While their original intent was simply the retrieval of phase information of a pair of light distributions through propagation functions (such as the Fourier transform), their algorithm applies remarkably well to digital holography reconstruction. In the context of a digital image input, the principle of the algorithm is thus: given enough iterations, an adequate reconstruction of an input image can

be made through a source intensity beam and a phase hologram, which is sometimes also referred to as a phase mask.

First, an input beam intensity, in the sample code case a Gaussian beam (figure 11), and a randomly generated phase, are combined to get amplitude at the source (focal) plane according to equation 1 (top left of block diagram). From [26], the intensity of the Gaussian beam profile (or normal distribution) is given as:

$$I = I_0 e^{-2\frac{r^2}{\sigma^2}} \tag{2}$$

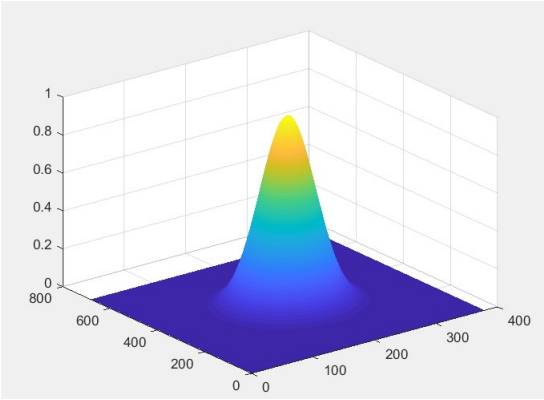

Figure 11: Gaussian beam generated in MATLAB

Where I is the intensity profile, I_0 the beam peak, r the beam radius, and σ the beam waist.

Note that this ignores the fact that the incident beam might include a phase term. The first phase term is also random in order to avoid constant phases over iterations if the input intensity pattern is centrosymmetric, which would fail to change the phase term with Fourier transforms. A forward Fourier transform is applied in order to get amplitude in the focal plane, and the phase term of this amplitude term is retrieved, while the intensity term forms the approximation to the target intensity and improves in fidelity as iteration number increases. The retrieved phase term is then combined with the target intensity, which is $abs()$ of the input image we desire to reconstruct through a hologram. An inverse Fourier transform is then applied to this approximation of the desired target amplitude to get back into the focal plane, from which the phase term serves as:

1. The phase hologram which, when coupled with the input beam intensity, serves to give a reconstruction of the target image.

2. The phase term of the source amplitude of the next iteration, replacing the randomly generated phase of the very first iteration.

The algorithm then runs until error converges, or however many times we specify for it to iterate. For basic images convergence occurs very fast, while for more complicated images, running more images gives more detail in reconstruction, as is shown in section 4.1.

Different input beam types were also considered in order to compare error evolution for different cases. In the first case, a uniform plane wave was generated as in figure 12 by setting the exponential term of equation 2 to being equal to 1.

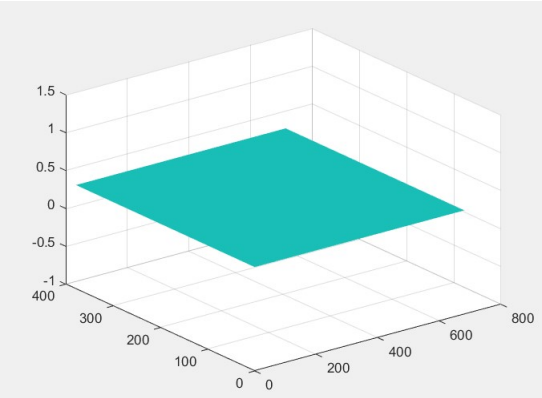

Figure 12: Uniform plane wave generated in MATLAB

Other potential input beams that could be used include, but are not limited to: uniform waves with random phase, Airy beams, Bessel beams, and so on.

The code for 2D GS is provided in appendix A.

3.2 DMD and SLM simulation, methods

The use of DMDs (Digital Micromirror Devices) and SLMs (Spatial Light Modulators) are both ways light sources may be diffracted in order to create our desired light distributions. As such, simulating their workings in conjunction with the 2D GS is worthwhile in order to compare performance.

A DMD is a MOEMS (micro-opto-electromechanical system) consisting of several hundred thousands of mirrors that can be rotated in particular angles at a high frequency (¿20 kHz) as to allow for light to be distributed in a binary way: an ON/1 state consists of the mirrors allowing light to be reflected into the lens, and an OFF/0 state consists of light being directed elsewhere, such as onto a heat-sink [27]. In order to simulate its workings, which can allow for binary image displays via particular arrangement of mirrors, the same algorithm as before was applied to a image converted to a binary black and white image using *imbinarize*. Due to the nature of DMDs only being able to display in binary patterns, no phase modulation is possible - in this case only intensity can be modified over iterations.

An SLM is an optical object capable of diffracting an input intensity by imposing phase or amplitude modulation on it. In order to simulate its workings, it is necessary to discretise phase going from $-\pi$ to $+\pi$ to going from 0 to 255, according to the following equation:

$$p_{discretize} = 255 \times \frac{p_{circular} + \pi}{2\pi} \tag{3}$$

Where $p_{discretize}$ is discretized phase and $p_{circular}$ is circular phase.

Due to time and knowledge limitations, however, these simulation methods were not fully implemented. The non-complete implementations are provided in appendix C.

3.3 3D GS with Fresnel diffraction, methods

As explained at the end of section 1.4, the goal is simultaneous stimulation of neurons in multiple planes, and as such it became necessary to develop a three-dimensionalised version of the basic GS algorithm. To do so, a Fresnel diffraction method was employed. Fresnel diffraction simulates the behaviour of wave diffraction in the near field - given a certain input field (image), Fresnel diffraction shows what the input field looks like if propagated a certain distance z. This allows for the previously used 2D GS to be used in 3D. It is done by splitting a 3D object into a discrete set of two dimensional planes and propagating a set amount with Fresnel diffraction for each plane. Essentially, for every iteration, it runs the 2D version of GS as many times as there are stacks. The version of Fresnel propagation used was adapted from [28], and is shown to work adequately on the input image shown below. It is based off of the convolutional form of Fresnel diffraction, given by the formula [29]:

$$E(x, y, z) = E(x, y, 0) * h(x, y, z) \tag{4}$$

$$h(x, y, z) = e^{\frac{ikz}{(i\lambda z)}} e^{i\frac{k}{2z}(x^2+y^2)} \tag{5}$$

$$k = \frac{2\pi}{\lambda} \tag{6}$$

Where E is the field at a given z position, z is the distance propagated in pixels, $*$ is the convolution operator, h is the convolution "mask" we convolve the input field with, k is wavenumber, and λ is size of pixel we propagate by.

This version of Fresnel propagation applies a forward Fourier transform to the input field and multiplies it by the convolution mask/kernel function (essentially convolving both phase and intensity of every pixel on an image), and the product is inverse Fourier transformed back. The effect on an input image is a thickening and progressive blurring of features. For example, an image of a binary "1" is the input field and is converted to a double before applying the Fresnel propagation. Figure 13 shows the result after a 100 pixel unit propagation forward and figure 14 shows the result after a 1000 pixel unit propagation forward. It is clearly noticeable that the farther the propagation, the more thick and blurry the features of the original input image are. The key to achieving a 3D version of the GS algorithm lies in the fact that propagation also works backwards - in this case, if we propagate

back exactly 100 and 1000 micrometers back, we get the exact same image as the input "1". As such, it is possible to create one hologram which goes through forward and backward Fresnel propagation n number of times each iteration, where n is the number of planes/stacks that the 3D input image is split into. The code for Fresnel propagation is provided in appendix D.

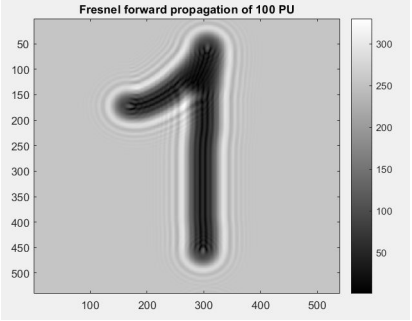

Figure 13: Forward Fresnel, 100 pixel units

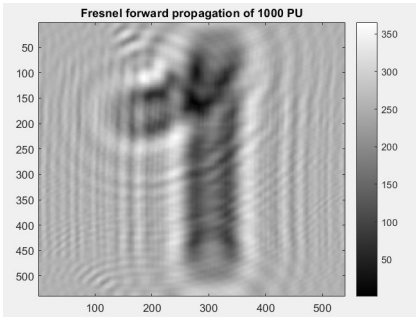

Figure 14: Forward Fresnel, 1000 pixel units

The block diagram for the 3D version of the GS algorithm is shown below (figure 15).

Like the original 2D GS, the intensity of an input beam (again, a Gaussian beam) is taken and associated with a random phase to get amplitude at the source/focal plane. A forward Fourier transform is applied in order to get amplitude in the focal plane. At this step, Fresnel diffraction is applied to the amplitude field n times at a linearly increasing propagation distance each time, and the phase term of each one is retrieved and combined with the target intensity for that specific propagation distance. For example, the input images used for this algorithm were the images 1 to 9 in black and white, converted to double. The target intensity of "3" (simply $abs()$ of the image) is combined with the amplitude in the target plane that has been propagated 3 x 100 pixel units forward. Similarly, the target intensity of "9" is combined with the amplitude in the target plane that has been propagated 9 x 100 pixel units forward. For nine planes, we will get nine different amplitude fields in 9 different target planes. Note that as with the 2D version, the intensity term for every post-Fresnel propagation plane forms the intensity approximation of each stack/plane. Then, a backward Fresnel propagation is applied to every stack/plane at a distance equal to that from the forward Fresnel step, followed by an inverse Fourier transform back to the focal plane. At this

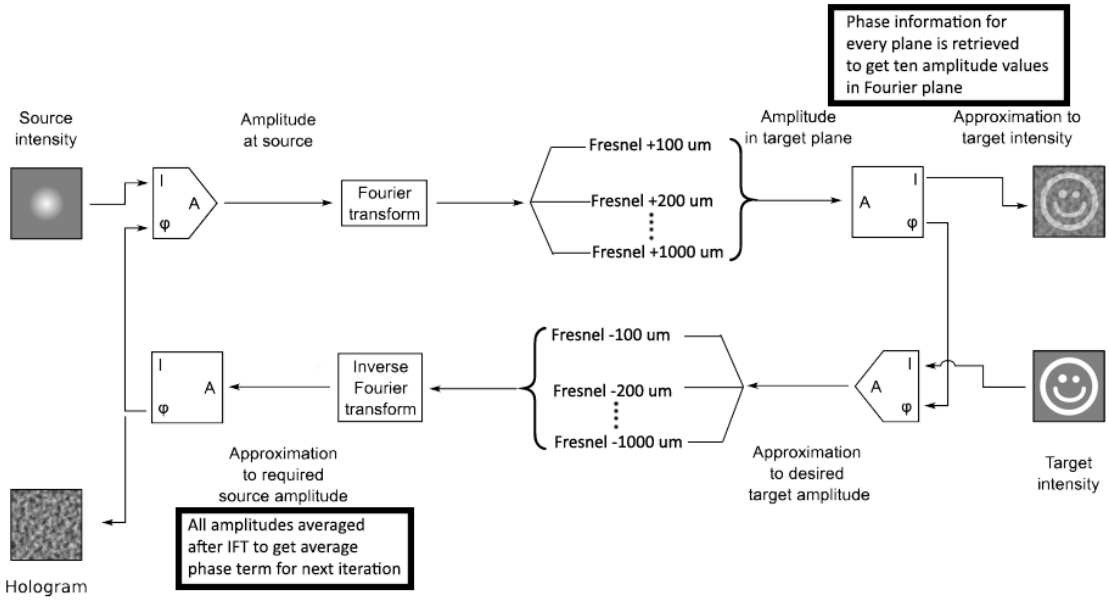

Figure 15: Block diagram for 3D Gerchberg-Saxton algorithm

point there should be nine fields at the focal plane, one for each stack/plane, which can then be averaged to get one field amplitude. At this point, the phase term of this field serves as:

1. The phase hologram which, when coupled with the input beam intensity, and the correct forward Fresnel propagation distance, serves to give a reconstruction of one of the target images from the 3D input stack.

2. The phase term of the source amplitude of the next iteration, replacing the randomly generated phase of the very first iteration.

The code for 3D GS is provided in appendix B.

4 Results

All algorithms used were implemented in MATLAB R2017b, unless stated otherwise, without the use of extra toolboxes.

4.1 First iteration of normal 2D GS, results

Figure 16 shows the Gaussian beam reconstructions of the target image over increased iterations by taking the intensity of the field after step 2 in the block diagram, and figure 17 shows the phase hologram for this particular run of the algorithm. Figure 18 shows the evolution of error over iterations for a Gaussian beam as input, while figure 19 shows the same for a uniform plane wave. The error is obtained by taking the RMS (root mean squared) error of the reconstruction by using *immse* with respect to the input image. Using the *tic* and *toc* tools on MATLAB, it is shown that this version of the algorithm takes approximately 5 seconds to run 100 iterations.

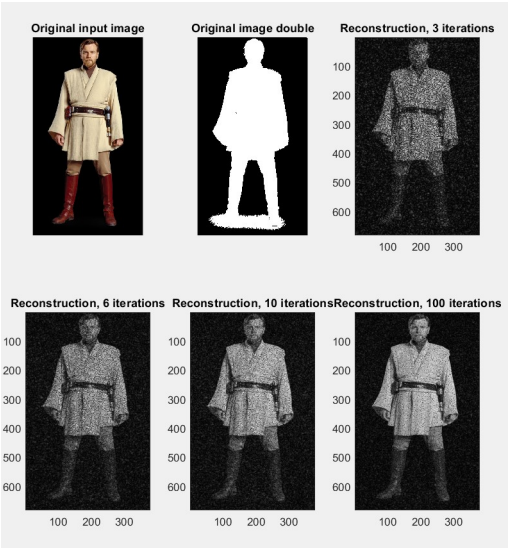

Figure 16: Reconstructions over increasing iterations of Obi-Wan input image

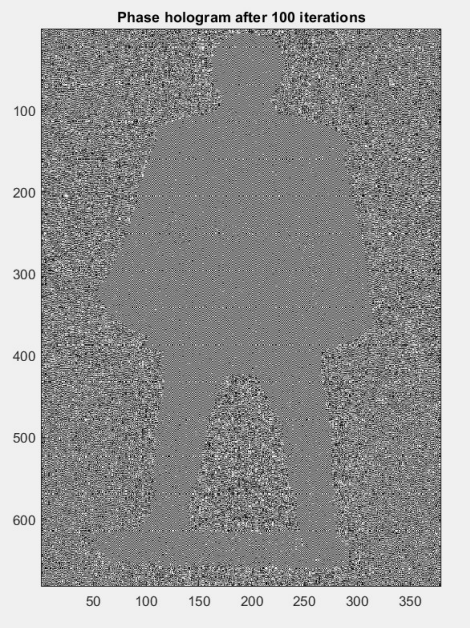

Figure 17: Phase hologram of Obi-Wan input image

Thus, the most basic version of the algorithm converges within less than

ten iterations, as enhanced reconstruction is not as significant beyond the first ten iterations, as shown by error converging to sub 1000. However, for a more complex input image (more varied shape and color), an increased number of iterations is still beneficial to the overall reconstruction's finer details, such as with better shading shown by the 100th iteration reconstruction (figure 16).

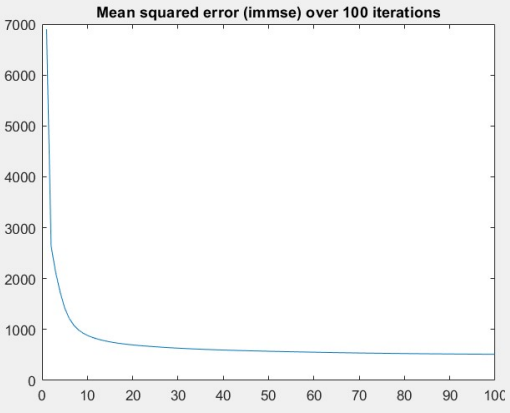

Figure 18: Error convergence with basic GS, Gaussian beam input

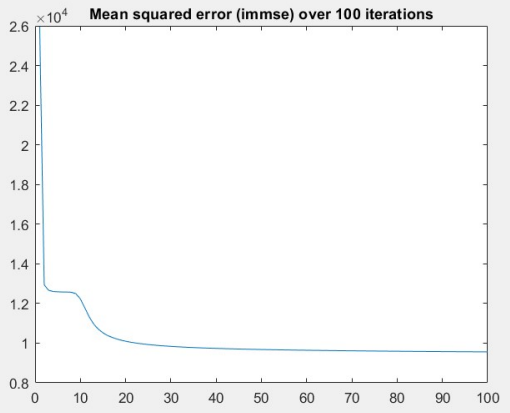

Figure 19: Error convergence with basic GS, Uniform plane wave input

4.2 DMD and SLM simulation, results

Attempts were made to incorporate intensity modulation as well as phase discretization into the basic 2D GS, but lack of expertise and time yielded no successful reconstructions, so no concrete results are presented here.

4.3 3D GS with Fresnel diffraction, results

Figure 20 and 21 show two of the nine planes used as target images for reconstruction and figure 22 and 23 show their reconstruction after 20 iterations. The error evolution of every plane is also shown over 50 iterations in figure 24. The phase hologram given by the average of the phase holograms of all planes is shown in figure 25. Using the *tic* and *toc* tools on MATLAB, it is shown that this version of the algorithm takes approximately 51 seconds to run 50 iterations.

This version of the algorithm does not converge, as opposed to the basic 2D GS version. MSE evolves erratically but all are under 5E7 exhibiting similar, non converging errors besides over iterations with no clear discernible pattern. Note that reconstructions of planes "under" the current plane can also be observed for any plane other than plane 1 (such as in figure 23, where the "8" reconstruction is slightly visible under that of the "9"). Additionally, as error converged faster and was lower for the Gaussian beam input, the uniform plane wave was expected to not perform as well in the 3D GS, either, so the results are not shown here.

Figure 20: Input image for plane 1

Figure 21: Input image for plane 9

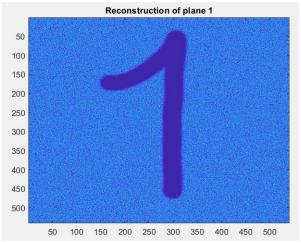

Figure 22: Reconstruction of plane 1 after 20 iterations

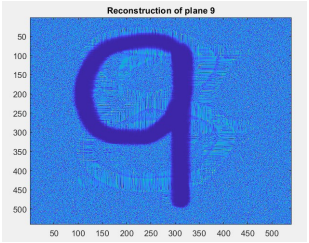

Figure 23: Reconstruction of plane 9 after 20 iterations

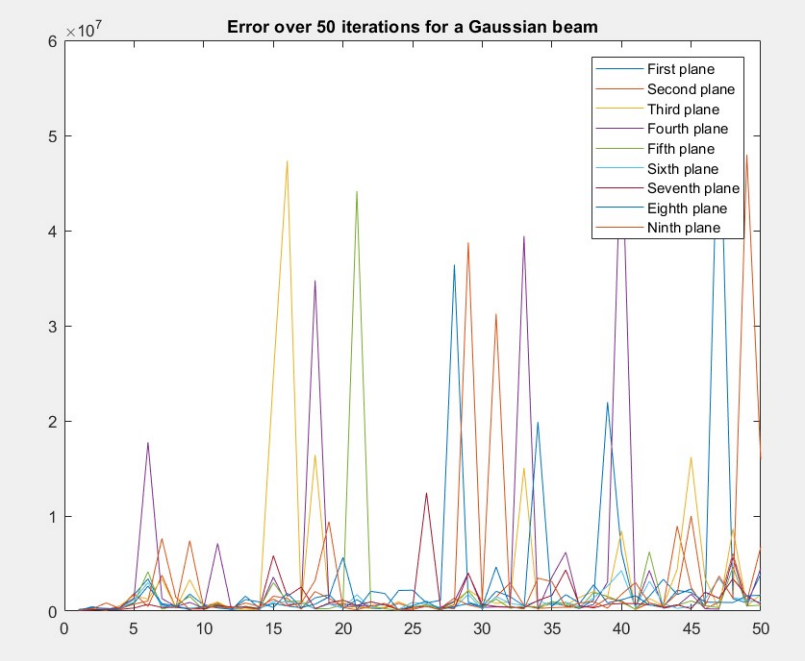

Figure 24: Error evolution for all planes of the 3D GS algorithm, zoomed in

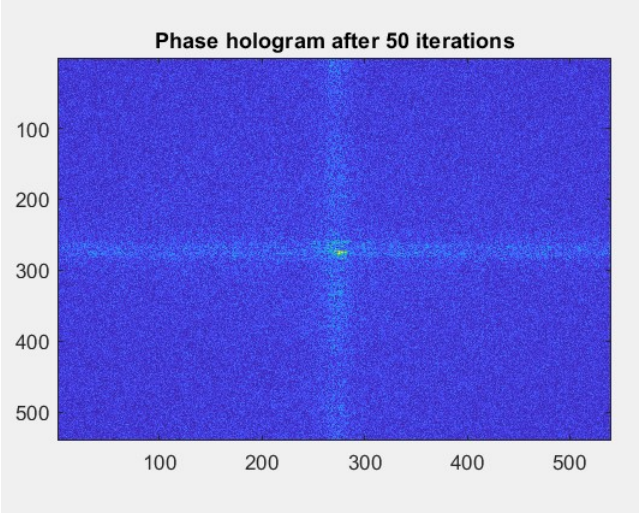

Figure 25: Average of phase holograms of all planes

5 Discussion

5.1 First iteration of normal 2D GS, discussion

The 2D version of the GS algorithm works as expected - the reconstructions appear adequate enough and error converges relatively fast in both cases of input beams (typically under less than 10 iterations) but MSE values are one order of magnitude lower in the case of a Gaussian beam, which is why reconstructions were not displayed for the uniform plane wave. The phase hologram also shows the outline of the input image as expected, as phase based holograms will contain the positional information of the input image.

2D holograms do not offer much for the purposes of volumetric neural stimulation, however, as they are limited by their capacity to reconstruct one two-dimensional image. As such, it became necessary to extend them into a third dimension.

Note that the algorithm is referred to as a phase modulated GS algorithm due to the intensity term of the approximation to required sources amplitude being discarded and not made use of for the next iteration. There are possibilities in making use of the intensity information to create an intensity modulated GS, which is touched upon in section 5.4.

5.2 DMD and SLM simulation, discussion

Unfortunately, no conclusive algorithms were created that allowed the full simulation of DMDs and SLMs, as such no notable error evolutions or reconstructions may be compared for discussion.

5.3 3D GS with Fresnel diffraction, discussion

The 3D version of the GS algorithm Reconstruction of planes 1 (figure 22) and 9 (figure 23) after 20 iterations is accurate enough and might not converge due to the simplicity of the input images (perfect black on white). It is clear, however, that unlike the 2D version, error does not decrease as iteration number increases, sometimes even spiking up up to tenfold. This is likely due to inaccuracies either in the input beam or the Fresnel propagation, where the residual reconstructions from previous planes (such as the ”8”

behind the reconstruction of the "9") might influence the mean squared error calculations. This version of the 3D GS works as a "naive" algorithm and works for most simple cases. In theory, however, since we are taking the average of 10 phase holograms, there is a possiblity that the phases can circularly average to zero, thereby causing another phase hologram related concern, as an intensity based reconstruction would not be affected (there is no "negative" intensity). Using the *tic* and *toc* tools on MATLAB, it is shown that this version of the algorithm takes approximately 51 seconds to run 50 iterations.

5.4 Further work and alternatives considered

Had the algorithm been fully developed, it would have been have been considered successful for these hologram parameters:

1. Resolution(x-y plane): 2-3 micrometers
2. Axial resolution (z-plane): 10 micrometers
3. Wavelength: 450-490 nanometers
4. Neural bandwidth(projection speed): 10 Hz (1 kHz as ultimate goal)
5. Power efficiency: 1 percent, few millimeters squared illumination (milliwatts per centimeter squared)

A notable strategy which has not been implemented due to lack of time and knowledge is the weighted GS algorithm, where the weighting of light on every pixel is considered differently for parts we would like to stimulate differently. For a non-weighted GS, weighting of light is considered as infinite over every pixel. An ideal weighted GS would consider important parts to illuminate, and non-important parts we want to avoid illumination on, such as non targeted neurons, and apply different weightings onto them by redistribution of light (for example by taking light on non-important parts and shifting them onto important parts).

Other alternatives that have been considered for implementation are intensity modulated algorithms, or both phase and intensity modulated algorithms, which tie in to the simulation of DMDs and SLMs.

6 Conclusion

To conclude with, this report has presented the need for new variations on the traditional Gerchberg-Saxton algorithm, which is popular method in computer generated holography for optogenetics, an up and coming field with huge potential in treatment of neural related diseases. Traditional implementations lack a combination of spatial and temporal resolution, and two-photon approaches that do achieve both are held back by power limitations, restricting the number of photons that may be influenced without overexposing subjects to dangerous doses. To this end, methods for basic 2D and 3D GS algorithms were implemented in MATLAB, with potential improvements also being considered but not brought to fruition due to lack of time or expertise.

References

[1] Edward S. Boyden, Feng Zhang, Ernst Bamberg, Georg Nagel, and Karl Deisseroth. Millisecond-timescale, genetically targeted optical control of neural activity. *Nature Neuroscience*, (8), 2005.

[2] Michiel van Wyk, Justyna Pielecka-Fortuna, Siegrid Löwel, and Sonja Kleinlogel. Restoring the ON Switch in Blind Retinas: Opto-mGluR6, a Next-Generation, Cell-Tailored Optogenetic Tool. *PLoS Biology*, 13(5):1–30, 2015.

[3] Antoine Chaffiol, Romain Caplette, Céline Jaillard, Elena Brazhnikova, Mélissa Desrosiers, Elisabeth Dubus, Laëtitia Duhamel, Emilie Macé, Olivier Marre, Patrick Benoit, Philippe Hantraye, Alexis Pierre Bemelmans, Ernst Bamberg, Jens Duebel, José Alain Sahel, Serge Picaud, and Deniz Dalkara. A New Promoter Allows Optogenetic Vision Restoration with Enhanced Sensitivity in Macaque Retina. *Molecular Therapy*, 25(11):2546–2560, 2017.

[4] Alexander Erofeev, Olga Zakharova, Stanislav Terekhin, Polina Plotnikova, Ilya Bezprozvanny, and Olga Vlasova. Future of Optogenetics: Potential Clinical Applications? *Opera Med Physiol*, 2(2):117–121, 2016.

[5] André Fiala, Anna Suska, and Oliver M. Schlüter. Optogenetic approaches in neuroscience. *Current Biology*, 20(20):897–903, 2010.

[6] Francis Crick. The impact of molecular biology on neuroscience. *Philosophical Transactions of the Royal Society B*, 354(1392):2021–2025, 1999.

[7] Boris V. Zemelman, Georgia A. Lee, Minna Ng, and Gero Miesenböck. Selective photostimulation of genetically chARGed neurons. *Neuron*, 33(1):15–22, 2002.

[8] Karl S.Sohal, Vikas; Zhang, Feng; Yizhar, Ofer; Deisseroth. Parvalbumin neurons and gamma rhythms enhance cortical circuit performance. *Nature*, 2009.

[9] Ofer Yizhar, Lief E. Fenno, Thomas J. Davidson, Murtaza Mogri, and Karl Deisseroth. Optogenetics in Neural Systems. *Neuron*, 71(1):9–34, 2011.

[10] Dennis Gabor. A New Microscopic Principle. *Nature Publishing Group*, 1948.

[11] Elizabeth Gibney. Physicists create Star Wars-style 3D projections — just don't call them holograms. `https://www.nature.com/articles/d41586-018-01125-y`, 2018. [Online; accessed 06-June-2019].

[12] Chris Slinger. Computer-Generated Holography as a Generic Display Technology. 2005.

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

[14] Rajiv Desai. Hologram. `http://drrajivdesaimd.com/2017/09/17/hologram/`, 2017. [Online; accessed 06-June-2019].

[15] Bob Mellish. Holograph-record. https://commons.wikimedia.org/wiki/File:Holography-record.png, 1 August 2002. [Online; accessed 1-November-2018].

[16] J. P. Rickgauer and D. W. Tank. Two-photon excitation of channelrhodopsin-2 at saturation. *Proceedings of the National Academy of Sciences*, 106(35):15025–15030, 2009.

[17] Weijian Yang, Luis Carrillo-Reid, Yuki Bando, Darcy S. Peterka, and Rafael Yuste. Simultaneous two-photon imaging and two-photon optogenetics of cortical circuits in three dimensions. *eLife*, 7:1–21, 2018.

[18] Guanghao Zhu, James Van Howe, Michael Durst, Warren Zipfel, and Chris Xu. Simultaneous spatial and temporal focusing of femtosecond pulses. 13(6):2153–2159, 2005.

[19] Vittorio A. Sironi. Origin and Evolution of Deep Brain Stimulation. *Front Integr Neurosci.*, 5:42, 2011.

[20] C. A. Pagni, M. G. Altibrandi, A. Bentivoglio, G. Caruso, B. Cioni, C. Fiorella, A. Insola, A. Lavano, R. Maina, P. Mazzone, C. D. Signorelli, C. Sturiale, F. Valzania, S. Zeme, and F. Zenga. Extradural motor cortex stimulation (EMCS) for Parkinson's disease. History and first results by the study group of the Italian neurosurgical society. *Acta Neurochir. Suppl.*, 93:113–119, 2005.

[21] Medtronic receives fda approval for deep brain stimulation therapy for medically refractory epilepsy. http://newsroom.medtronic.com/phoenix.zhtml?c=251324&p=irol-newsArticle&ID=2345882, 2018. [Online; accessed 6-January-2018].

[22] Jean Delbeke, Luis Hoffman, Katrien Mols, Dries Braeken, and Dimiter Prodanov. And Then There Was Light : Perspectives of Optogenetics for Deep Brain Stimulation and Neuromodulation. 11(December):1–20, 2017.

[23] Brain-machine interfaces: Bidirectional communication at last. https://www.sciencedaily.com/releases/2017/02/170222131442.htm, 2017. [Online; accessed 6-January-2018].

[24] Musa Aydin. Gerchberg-Saxton algorithm. https://uk.mathworks.com/matlabcentral/fileexchange/65979-gerchberg-saxton-algorithm, 2018. [Online; accessed 26-October-2018].

[25] Pseudo-code for gerchberg-saxton algorithm. https://en.wikipedia.org/wiki/Gerchberg2018. [Online; accessed 13-October-2018].

[26] Beam diameter. https://en.wikipedia.org/wiki/Beam_diameter, 2019. [Online; accessed 13-June-2019].

[27] Digital Micromirror Device. https://en.wikipedia.org/wiki/Digital_micromirror_device, 2019. [Online; accessed 13-June-2019].

[28] Fresnel diffraction in two steps. https://stackoverflow.com/questions/20971945/fresnel-diffraction-in-two-steps, 2018. [Online; accessed 02-February-2019].

[29] Fresnel diffraction. https://en.wikipedia.org/wiki/Fresnel_diffraction, 2018. [Online; accessed 02-February-2019].

A 2D GS code

```
clear all; close all; clc;
tic;

x = linspace(-10,10,379);
y = linspace(-10,10,682);
[X,Y] = meshgrid(x,y);


x0 = 0;
y0 = 0;
sigma = 2;
A = 1;
res = ((X-x0).^2 + (Y-y0).^2)./(2*sigma^2);
% res = ones(682,379); replace previous line with this one if uniform plane wave required
input_intensity = A  * exp(-res);
surf(input_intensity);
shading interp;

input_target = rgb2gray(imread('Obi_wan.png'));
target = double(input_target);
A = ifft2(ifftshift(target));
error = [];

figure;
subplot(2,3,1);
imshow('Obi_wan.png');
title('Original input image');

for i = 1:100
    B = abs(input_intensity) .* exp(1i*angle(A));
    C = fftshift(fft2(B));
    D = abs(target) .* exp(1i*angle(C));
    A = ifft2(ifftshift(D));
    error = [error; immse(abs(C),target)];
```

```
35      if i == 3
36          subplot(2,3,2);
37          imshow(target);
38          title('Original image double');
39          subplot(2,3,3);
40          imagesc(abs(C/mean(C(:))));
41          colormap(gray(256));
42          title('Reconstruction, 3 iterations');
43      end
44      if i == 6
45          subplot(2,3,4);
46          imagesc(abs(C/mean(C(:))));
47          colormap(gray(256));
48          title('Reconstruction, 6 iterations');
49      end
50      if i == 10
51          subplot(2,3,5);
52          imagesc(abs(C/mean(C(:))));
53          colormap(gray(256));
54          title('Reconstruction, 10 iterations');
55      end
56      if i == 100
57          subplot(2,3,6);
58          imagesc(abs(C/mean(C(:))));
59          colormap(gray(256));
60          title('Reconstruction, 100 iterations');
61      end
62  end
63
64  figure;
65  i = 1:1:i;
66  plot(i,('error'));
67  title('Mean squared error (immse) over 100 iterations')
        ;
68
69  figure;
70  imagesc(abs(angle(fftshift(fft2(A)))));
71  title("Phase hologram after 100 iterations");
```

```
colormap(gray(256));
toc;
```

B 3D GS code

```
clear all; close all; clc;
tic;

myFolder = 'C:\Users\....\onetoten'; % replace with own
    folder pathway
if ~isdir(myFolder)
  errorMessage = sprintf('Error: The following folder
     does not exist:\n%s', myFolder);
  uiwait(warndlg(errorMessage));
  return;
end

filePattern = fullfile(myFolder, '*.jpg');
jpgFiles = dir(filePattern);
imageArray = cell(1,10);

for k = 1:length(jpgFiles)
  baseFileName = jpgFiles(k).name;
  fullFileName = fullfile(myFolder, baseFileName);
  fprintf(1, 'Now reading %s\n', fullFileName);
  imageArray{k} = imread(fullFileName);
  imageArray{k} = double(imageArray{k})/255;
end

x = linspace(-10,10,540);
y = linspace(-10,10,540);
[X,Y] = meshgrid(x,y);

x0 = 0;
y0 = 0;
sigma = 2;
A = 1;
res = ((X-x0).^2 + (Y-y0).^2)./(2*sigma^2);
```

```
% res = ones(682,379); replace previous line with this one if uniform plane wave required
input_intensity = A  * exp(-res);
surf(input_intensity);
shading interp;

planes = length(imageArray);
iteration_n = 50;
intensity_array = cell(1,planes);
amplitude_array = cell(1,planes);
rng(1);
average_amplitude = ifft2(ifftshift(imageArray{randi([1 9])}));

error = zeros(planes,iteration_n);

for iterations = 1:iteration_n

  focal_amplitude = abs(input_intensity).*exp(1i*angle(average_amplitude));
  lens_field = fftshift(fft2(focal_amplitude));

  for m = 1:planes
      fresnel_forward = fresnelpropagateft(lens_field,m,0.6,0.39,0.39);

      intensity_approximation = abs(fresnel_forward/mean(fresnel_forward(:)));
      intensity_array{m} = intensity_approximation;

      fourier_amplitude = abs(imageArray{m}).*exp(1i*angle(fresnel_forward));

      lens_field = fresnelpropagateft(fourier_amplitude,-m,0.6,0.39,0.39);

      focal_amplitude = ifft2(ifftshift(lens_field));
      amplitude_array{m} = focal_amplitude;
```

```
63
64          if m > 1
65            error(m,iterations) = immse(abs(intensity_array
                {m}),imageArray{m-1});
66          end
67      end
68
69      sum_amplitude = amplitude_array{2};
70      for i = 3:planes
71          sum_amplitude = sum_amplitude + amplitude_array{i
                };
72      end
73      focal_average = sum_amplitude/9;
74
75      average_amplitude = fftshift(fft2(focal_average));
76      phase_hologram = exp(1i*angle(average_amplitude));
77
78      sum_intensity = intensity_array{2};
79      for i = 3:planes
80          sum_intensity = sum_intensity + intensity_array{i
                };
81      end
82      average_intensity = sum_intensity/9;
83
84  end
85
86  figure;
87
88  subplot(2,1,1);
89  imagesc(abs(ifft2(ifftshift(phase_hologram))));
90  title("Phase hologram after " + iterations + "
        iterations");
91
92  subplot(2,1,2);
93  imagesc(average_intensity);
94  title("Average of all reconstructions after " +
        iterations + " iterations");
95
```

```
figure;
for i = 2:planes
    imagesc(intensity_array{i});
    title("Reconstruction of plane " + (i-1));
    drawnow;
    pause;
end

figure;

for e = 2:size(error,1)
    i = 1:1:iterations;
    plot(i,(error(e,:)'));
    hold on;
    title("Error over " + iterations + " iterations for
        a Gaussian beam");
    legend('First plane','Second plane','Third plane','
       Fourth plane','Fifth plane','Sixth plane','
       Seventh plane','Eighth plane','Ninth plane');
end

toc;
```

C DMD and SLM modifications

```
clear all; close all; clc;

%% Read in images and take Fourier domain info
pikachu1 = double(rgb2gray(imread('pikachu.png')));
pikachu2 = double(rgb2gray(imread('pikachu_no_tail.png'
    )));

figure;
subplot(1,2,1);
imagesc(pikachu1);
subplot(1,2,2);
imagesc(pikachu2);

```

```
ft_pikachu1 = fftshift(fft2(pikachu1));
ft_pikachu2 = fftshift(fft2(pikachu2));

intensity_p1 = abs(ft_pikachu1);
intensity_p2 = abs(ft_pikachu2);
phase_p1 = angle(ft_pikachu1);
phase_p2 = angle(ft_pikachu2);

%% DMD simulation: binary image
binary_pikachu1 = imbinarize(pikachu1);
binary_pikachu2 = imbinarize(pikachu2);

figure;
subplot(1,2,1);
imagesc(binary_pikachu1);
subplot(1,2,2);
imagesc(binary_pikachu2);

% remaining work needs to be done in incorporating
    intensity modulation in the GS algorithm

%% SLM simulation
phase_p1_discrete = ((phase_p1+pi)/(2*pi))*255;
phase_p2_discrete = ((phase_p2+pi)/(2*pi))*255;

% this now needs to be integrated in a full GS
    algorithm
```

D Code for Fresnel propagation

```
function out = fresnelpropagateft(U0, z, lambda, dx, dy
    )
    % input is U0 at wavelength lambda and returns
        field U after distance z
    % using Fresnel approximation. dx and dy are
        spatial resolutions.

    k = 2*pi/lambda; % wavenumber
```

```
6       [nx,ny] = size(U0); % nx = x dimension of U0, ny =
            y dimension of U0.
7
8       Lx = dx*nx;
9       Ly = dy*ny;
10
11      dfx = 1./Lx;
12      dfy = 1./Ly;
13
14      u = ones(nx,1)*((1:nx)-nx/2)*dfx;
15      v = ((1:ny)-ny/2)'*ones(1,ny)*dfy;
16
17      O = fftshift(fft2((U0)));
18
19      H = exp(1i*k*z).*exp(-1i*pi*lambda*z*(u.^2+v.^2));
20
21      out = ifft2(ifftshift(O.*H));
22  end
```