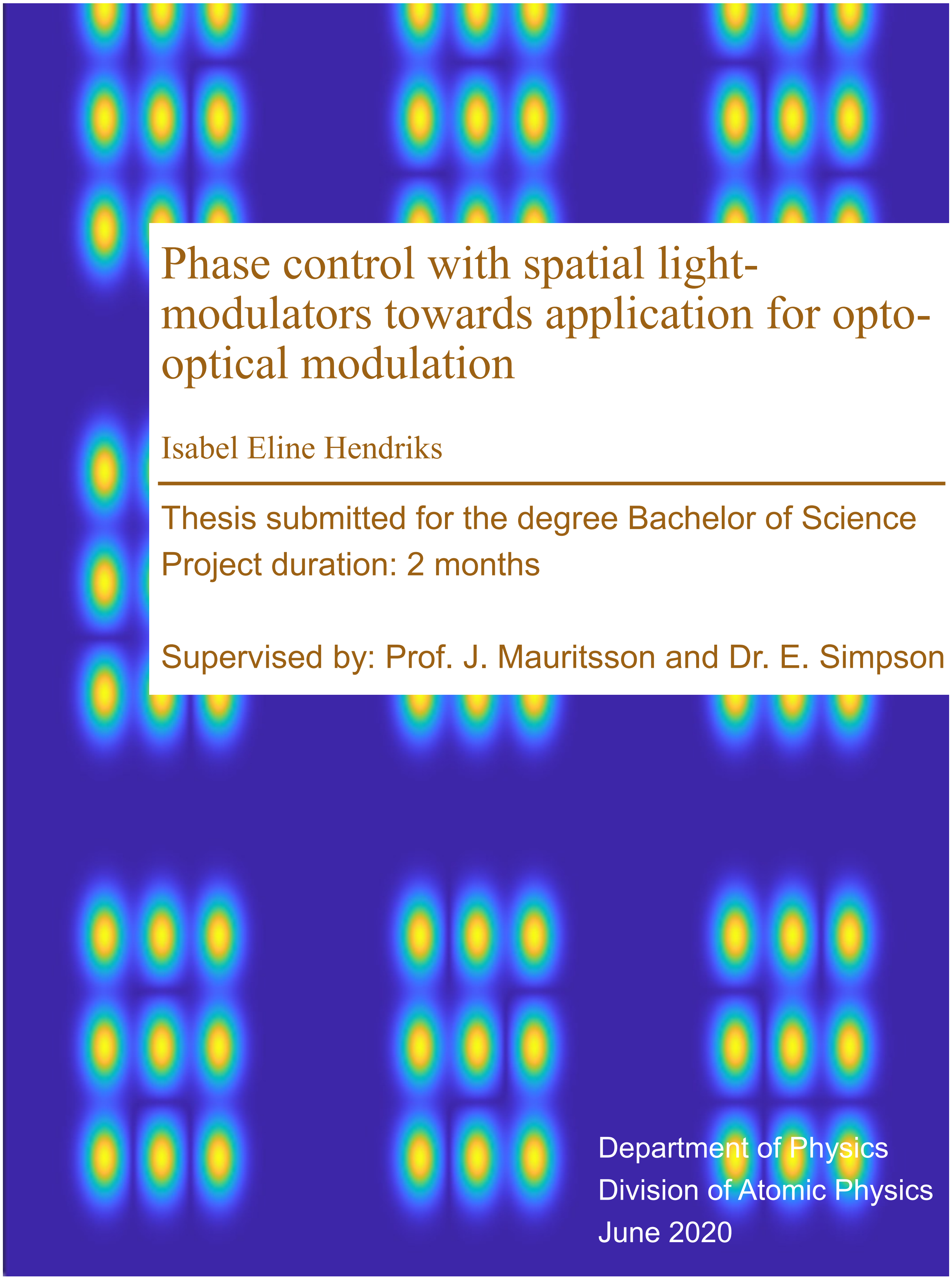
LUND UNIVERSITY
Faculty of Science



Phase control with spatial light-modulators towards application for opto-optical modulation

Isabel Eline Hendriks

Thesis submitted for the degree Bachelor of Science
Project duration: 2 months

Supervised by: Prof. J. Mauritsson and Dr. E. Simpson

Department of Physics
Division of Atomic Physics
June 2020

Phase control with spatial light-modulators towards application for opto-optical modulation

by Isabel Eline Hendriks

LRAP563 (2020)
Thesis submitted for the degree Bachelor of Science, General Physics
Project duration: Four months at half-speed
Thesis advisors: Johan Mauritsson and Emma Rose Simpson
Division of Atomic Physics

Abstract

This project is about focusing a Gaussian laser beam that has a spatial phase profile applied by a spatial light modulator. The goal is to be able to control the shape, position and phase of the peak at the focus and to split up the peak at the focus into several peaks. Controlling the profile of the beam at the focus has interesting applications for opto-optical modulation. A code was written in Matlab that simulates the focusing of a Gaussian beam. Within the code a phase profile can be applied to the Gaussian beam profile. The phase profiles that are tested are linear profiles and curved profiles. A function is written that combines phase profiles by alternating between pixels. This splits up the peak the focus into several peaks. It was found that by combining linear phases through alternating between pixels the peaks at the focus can be split up into several peaks. However, when doing so, a number of extra peaks appear at the focus. The origin of the extra peaks is not explained in this thesis, but their behaviour and consequences are studied. It was also found that the peaks can be controlled in time by applying a flat phase profile, and that the shape of the peak at the focus can be adjusted by applying a curved phase profile.

Abbreviations

DFT	Direct Fourier Transformation
FFT	Fast Fourier Transformation
IR	Infrared
OOM	Opto Optical modulation
SLM	Spatial Light Modulator
XUV	Extreme Ultraviolet

Table of Contents

1 Introduction

Ultrashort pulses of electromagnetic radiation in the extreme ultraviolet range (XUV) have extremely interesting applications. Since the pulses are in the attosecond range, they can be used to study ultra fast processes, such as the behaviour of electrons within an atom. Another interesting application of these pulses is quantum control. Due to the high photon energy they can be used for the control of short lived highly excited states [1]. In order to use XUV pulses, it is desirable to have a close control of the pulses. Because of their high photon energy, XUV pulses are absorbed by most materials. Therefore, regular devices to manipulate light are not applicable to XUV pulses. Opto-Optical Modulation (OOM) is a method to control XUV pulses. The XUV pulses are directed into a gas and get absorbed. Shortly after the XUV pulse, a control pulse follows at a non-resonant wavelength, which is in the infrared spectrum (IR). As the IR pulse passes through the gas it will temporarily shift the energy levels in the excited atoms, which will result in a phase shift in the XUV pulse that will be emitted when the gas de-excites. This allows us to control XUV pulses by controlling the shape of an IR pulse. Since IR pulses have a low photon energy, they are easy to control by conventional optical methods.

In order to get the desired shape of the XUV pulse, the IR pulse has to be shaped correctly. The IR pulse is shaped by a Spatial Light Modulator (SLM) that has 792 x 600 pixels. Each pixel can add a phase to a fraction of the beam. Every pixel can be controlled individually, which allows us to apply a 2 dimensional phase profile to the spatial profile of the beam. The profile of the IR beam is approximated as a Gaussian shape. After the phase profile is applied, the beam is focused into the gas. This focusing process is a 2 dimensional Fourier transformation. In order to create the desired profile at the focus, the correct phase has to be applied.

Figure 1 shows a possible setup that can be used for this. The gas cell is used to create the XUV pulses through high order harmonic generation and the iris is used to filter out the IR.

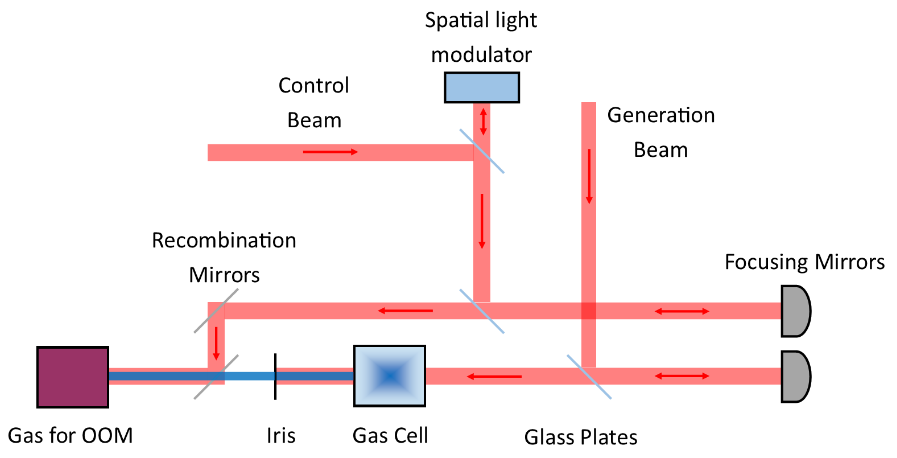


Figure 1: Possible setup

The aim of this project is to write a code that simulates this process. The input of the code is the phase profile, and the output is the two dimensional intensity profile at the focus. The code will first apply the phase shift to a Gaussian profile, and then focus this beam by applying a Fourier transformation. The code allows control over every pixel of the SLM, and can combine several phase profiles.

When there is no SLM and there is no phase shift applied, the spatial profile of the control pulse is approximately a single Gaussian peak. When this profile is used to control the XUV light, the part of the profile where the Gaussian shape is approximately a linear increasing line is used to to steer the XUV pulse. This will steer the pulse into a different direction. Approximating a Gaussian function with a linear increasing line is a rough approximation. With the code, different phase profiles can be used to change the shape of the control pulse, and get a shape that is closer to the desired shape. This will increase the control over the XUV pulses.

Besides controlling the shape of the peak, the SLM can also be used to control the position of the peak, or split the peak up into several peaks at the focus. When there are two peaks at the focus, one of the peaks can be delayed such that the peaks are out of phase. An extremely precise interferometer can be built when two peaks are travelling together while the phase difference between the two peaks can be controlled.

The goal of the project is to demonstrate the control over the pulses when only their phase is accessible. One goal is to split up the peak at the focus into several peaks and delay one of the peaks. To do this, first the pulses will be steered around using linear phase profiles. Then these linear phase profiles will be combined to split up the peak and create several peaks at the focus. After that a uniform phase profile is added to one of the peaks which will delay the peak. Another goal is to control the shape of the pulses by using curved phase profiles. The main focus of the project will be to create several peaks at the focus.

A limitation of this setup is that only the phase is accessible when shaping the IR pulse. The code will allow us to investigate what is possible when only the phase is accessible.

2 Theory

2.1 Generation of XUV pulses using High order Harmonic Generation (HHG)

The XUV pulses are created through a method called High order Harmonic Generation, which is based upon a semi-classical model. [2, 3] This technique uses a gas that is exposed to an alternating electric field. The strength of the electric field is similar to the binding energy of the nucleus and the electron in the gas. The alternating electric field is called the driving field, and influences the shape of the potential well of the atom. Due the changed shape of the potential well, the electron can tunnel through the potential barrier on certain occasions. After the electron has left the well, it will accelerate away from well because of the electric field. The shape of the electric field will change and the electron might accelerate back towards the well until it recombines with the nucleus. During the excursion the electron has gained kinetic energy during the acceleration. In this case the electron will the de-excite and emit a photon with energy $E_p = I_p + E_{kin}$, where E_p is the photon energy, I_p the ionization energy and E_{kin} the kinetic energy gained during the acceleration. An energy diagram showing this process is added in figure 22 in appendix A.1.1. [4, 5, 6, 7]

The frequency of the emitted radiation depends on how much energy the electron has when it recombines with the nucleus, which depends on the trajectory of the electron outside of the atom. This can be calculated using a classical approximation of the trajectory of the atom after it tunnels through the barrier. The acceleration of the electron after tunneling through the barrier is given by: [8]

$$\ddot{x} = -\frac{eE_0}{m_e}\sin(\omega_0 t) \tag{1}$$

Here E_0 is the electric field strength, ω_0 is the angular frequency of the IR laser pulse, e is the unit charge and m_e is the electron mass. By integrating it is found that the velocity is given by:

$$\dot{x} = -\frac{eE_0}{m_e\omega_0}[\cos(\omega_0 t) - \cos(\omega_0 t_i)] \tag{2}$$

Here t_i is the time of tunnel-ionization. If we now let t_r be the return time, or the time of recombination, we can find the kinetic energy in the following way.

$$E_k = \frac{1}{2}m_e\dot{x}^2 = \frac{e^2E_0^2}{m_e^2\omega_0^2}[\cos(\omega_0 t_r) - \cos(\omega_0 t_i)]^2 = U_p[\cos(\omega_0 t_r) - \cos(\omega_0 t_i)]^2 \tag{3}$$

Here U_p is the **pondermotive energy** [8], which represents the kinetic energy acquired by a free particle in an electric field. It can be shown that the maximum amount of energy that the electron can obtain during it's excursion is approximately 3.17 U_p. The value of U_p can be rewritten using the intensity I which is related to the amplitude by $I = c\varepsilon_0 E_0^2$, and the wavelength λ which is related to the angular frequency through $\omega = 2\pi c/\lambda$. This results in the following expression for the pondermotive energy:

$$U_p = \frac{e^2 \lambda^2 I}{8\pi^2 m_e \epsilon_0 c^3} \tag{4}$$

In order to find the total energy gained by the electron, the return time has to be determined. This is done by calculating the trajectory through integrating equation 2, and finding the time at which the displacements reaches zero again. Figure 23a) displays the trajectory that the electron follows dependent on the time at which the electron leaves the well (t_i). If the electron leaves while the potential is still increasing, the electron will gain too much energy and will not return to the potential well. These trajectories are marked in green. The blue trajectories are the electrons that recombine with the atom. The energy that the electrons gained can be calculated using equation 3 as a function of the travel time (T) which is the difference between the time of ionization (t_i) and the time of recombination (t_r).

Figure 23b) in appendix A.1.2 shows that an XUV pulse is not monochromatic, but contains a continuous spectrum of energies. When multiple pulses travel with short time intervals between them this will lead to interference, which will give discrete energies in the spectrum of the XUV light.

2.2 Opto-optical modulation

Opto-Optical modulation is a method that controls the phase of XUV pulses using a second control pulse, which is an IR pulse. The pulses are both directed into a gas of noble atoms. The first pulse is the XUV pulse. When the XUV pulse interacts with the gas, it gets completely absorbed and the gas ends up in a superposition of an excited state and the ground state. Without any external factors, the gas would de-excite, and the pulse would get emitted. However, the control pulse is focused into the gas right after the XUV pulse when the atoms are still in their excited state. The IR has a much lower energy than the excitation energy of the atoms, and is not resonant with the gas, meaning that absorption of the IR is negligible. The dominant effect of the IR on the atoms will be the so called AC-stark effect. [9]

The AC-stark effect describes the interaction between electromagnetic radiation and atoms that is also referred to as the *light shift*. [10] When the electromagnetic radiation hits the atoms, it will shift the energy levels in the atom due to interaction between the dipole moment and the non-resonant field. [1] The magnitude of the shift of the energy states can be calculated using equation 5 below:

$$\delta\omega = \frac{U_p}{\hbar} \tag{5}$$

Equation 5 shows us that $\delta\omega$ is proportional to U_p which implies that the shift of the energy states is proportional to the intensity of the electromagnetic radiation. This means that when an IR pulse is sent into a gas, the shift in the energy level will be proportional to the profile of the IR pulse.

The shift in the energy levels means that the electron will oscillate at a higher frequency when it is in a higher energy state. Since the control pulse is short it will result in a short change in frequency, which will cause the phase of the electron to change. When the electron de-excites, the emitted XUV light will have a different phase compared to the case of an atom that remained undisturbed. The phase shift can be calculated by integrating the energy shift over time, as shown in equation 6 below.

$$\Delta\Phi = \int \delta\omega(t)dt \tag{6}$$

The magnitude of the phase shift is approximately proportional to the intensity of the IR pulse, meaning that the intensity profile of the IR at the focus determines how the XUV pulse will be shaped. Therefore, it is important that the shape of the IR pulse can be closely controlled.

Shaping the infrared light is done with a Spatial Light Modulator (SLM). The SLM can add a phase to the light, without influencing the intensity or the polarization. This is done using liquid cells. The SLM can be controlled pixel by pixel, allowing a 2 dimensional phase profile to be added to the spatial profile of the beam. After the SLM, the beam is focused into the gas. This focusing process is a 2 dimensional Fourier transformation. [11]

When there is no phase shift applied, the profile of the IR pulse is approximately a Gaussian shape. The sides of a Gaussian profile can be approximated by a linear function. This linear function can then be used to steer the XUV pulse by adding or subtracting phase to the wavefront, as shown in figure 2 below.

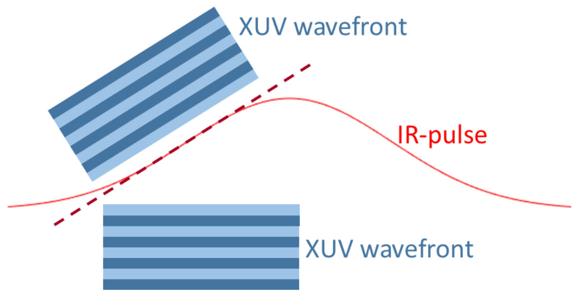


Figure 2: Changing the phase of the excited atoms using a Gaussian shaped control pulse

2.3 Numerical two dimensional Fourier transformation

The code has to be able to numerically perform a 2 dimensional Fourier transformation of the wave profile. A Fourier transformation transforms a function from the spatial to the frequency domain. The spatial domain is represented by x and the frequency domain is represented by ν. The analytical form of the one dimensional Fourier transformation is given in equation 7 below. Here $F(\nu)$ represents the Fourier Transformation of the original function $f(x)$.

$$F(\nu) = \int_{-\infty}^{\infty} f(x)e^{-i2\pi x\nu}dx \tag{7}$$

The general way to execute a Fourier transformation numerically is by using a Direct Fourier Transformation (DFT). The definition of the one dimensional DFT is given in equation 8. [12] It is similar to the definition of the analytical Fourier transformation, but instead of having a function as input, the input is a list with N discrete values.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-i\frac{2\pi}{N}kn} \tag{8}$$

Here X is the output list containing frequencies, which has indexes k and also has length N. The input list x describes the wavefront and can contain either real or complex numbers. The output list X contains complex numbers. The amplitude of the Fourier transformation is the absolute value of each value in X. An interesting property of the DFT is that it is a periodic over a period N, which is proven below.

$$X[k+N] = \sum_{n=0}^{N-1} x[n] \cdot e^{-i\frac{k}{N}2\pi n} \cdot e^{-i\frac{N}{N}2\pi n} = \sum_{n=0}^{N-1} x[n] \cdot e^{-i\frac{k}{N}2\pi n} = X[k]$$

Due to the periodicity of the Fourier transformation, the frequencies in X between $k = N/2$ to $k = N$ are identical to the samples between $k = -N/2$ to $k = 0$. Therefore, it is possible to shift the values between $k = N/2$ to $k = N$ to the beginning of the list. This operation is called a Fourier shift.

For each value in X, the function runs through x and sums the value of the the exponent at each point. This means that for a list with N values, the complexity of the computation is $\mathcal{O}(N^2)$. This becomes problematic when doing a Fourier transformation at a high resolution, since the code becomes very slow. A more practical approach is the Fast Fourier Transformation (FFT), which uses an optimization algorithm. This optimization is based upon the periodicity of the Fourier transformation. Because of this periodicity, the calculations for the values between $k = N/2$ to $k = N$ are redundant as these have already been done for the values between $k = 0$ to $k = N/2$. The FFT algorithm gives the same solution as the DFT, but it has a much shorter computation time. The FFT algorithm is fastest when N is a power of 2. In this case, the computation complexity is $\mathcal{O}(N \log N)$. [13] However, the FFT algorithm is still much faster than the DFT algorithm when N is not a power of 2.

In order to apply a Fourier transformation to a wave profile, it has to be done in two dimensions. The definition of the DFT in two dimensions is given in equation 9 below.

$$X[k,l] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m] \cdot e^{-i2\pi(\frac{kn}{N}+\frac{ml}{M})} \tag{9}$$

The DFT in two dimensions has the same properties as the one dimensional DFT. The dimensions of x and X are both M x N. The optimization algorithm for the one dimensional DFT can also be applied to the two dimensional DFT, meaning that this calculation can also be done for high resolutions.

3 Method

3.1 Writing the simulation code

The main purpose of the simulation code is to apply a two dimensional Fourier transformation to a spatial profile. Both Matlab and Python have built in libraries containing functions that can apply the FFT algorithm, but in order to really understand how a numerical two dimensional Fourier transformation works, a code was written from scratch that performs the DFT in two dimensions. First the DFT in one dimension was written and some tests were done to see if it worked correctly. The code also contains a function for an inverse Fourier transformation, which was helpful for testing, as the inverse Fourier transformation should be the same as the original function. The test showed that in order to get the right shape of the curve, the Fourier shift had to be performed. In one dimension the DFT would run within a reasonable amount of time for a list of 1000 points. As this is a high enough resolution, there was no need investigate in using the FFT algorithm in one dimension. The code was written in Python, and can be found in appendix H.1.

The code was expanded to two dimensions. The inverse Fourier transformation and Fourier shift were also expanded to two dimensions. This code can be found in appendix H.2. When testing with the two dimensional code, it would take about ten minutes to run the code using a profile with the dimensions of 30 x 30 pixels. This is not enough since the SLM has a size of 600 x 792 pixels. It became clear that the DFT algorithm was too slow, and that the FFT algorithm was needed. This optimization algorithm is quite complicated, and the focus of this project is not to understand optimization processes, so the built in libraries were used to perform the two dimensional FFT. Despite the fact that the FFT is fastest when using a number that is a power of 2 for the size of the matrix, it still runs within a reasonable amount of time when using a matrix with the size 600 x 792.

3.2 Simulation code

The code is written in Matlab and can be found in appendix B. The goal of the code is to simulate how the infrared pulse is shaped. The beam profile that is used is a Gaussian profile. The parameters of this profile can be adjusted. The input of the code is the phase profile. The code contains a function called `combphases` that can combine different phase profiles. This function contains a few default phase profiles. The output of the code is the beam profile at the focus.

3.2.1 Creating the beam profile

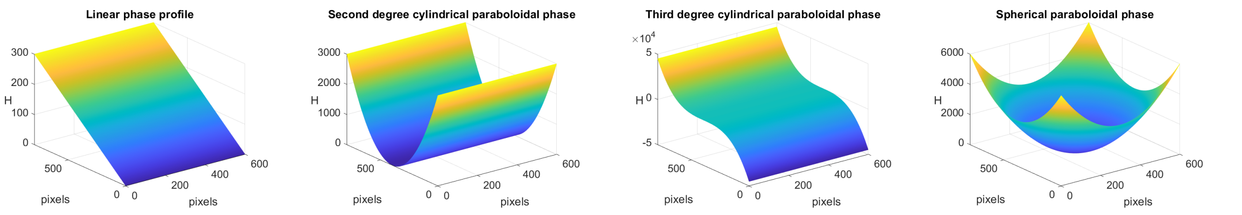
The first part of the code sets the amount of pixels, nX and mY, and the values between which the axis range, called aX, aY, bX and bY. The obvious choice is to set nX to 600 and mY to 792, as these are the dimensions of the SLM. The values of aX, bX, aY and bY are set to -10 and 10. These are dimensionless values that are used to create the Gaussian profile, but have no physical meaning. To create the Guassian profile, the parameters of the Gaussian function are set. The equation for a 2 dimensional Gaussian profile is given in equation 10 below.

$$A \cdot \exp\left(-\pi \cdot \frac{(x-x_0)^2 + (y-y_0)^2}{2\sigma^2} \right) \tag{10}$$

Here A is the amplitude of the peak, x_0 and y_0 are the distance to the center in x and y direction respectively, and σ is the peak width. The values of x_0 and y_0 are set to 0 and A and σ are set to 1. Two lists are created with the `linspace` command that represent the horizontal and vertical axis called `x` and `y`. Through element-wise operators a matrix is created with dimensions 792 x 600 that applies a the Gaussian function to these axis. This matrix represents a 2 dimensional Gaussian profile and is called `input_profile`.

3.2.2 Creating the phase profile

The phase profile that will be added to the Guassian profile is created next. The phase profile is a Matrix of dimensions `nX` by `mY` called `phase_profile`. Some examples of phase profiles that are of interest for this project are displayed in figure 3. Figure 3a) shows a linear phase. Figure 3b) and c) show a second degree and third degree cylindrical paraboloid and figure 3 d) shows a spherical paraboloid. The main interest of this project is to study the combination of several linear phases. By combining linear phases in different direction and with different gradients, the peak at the focus can be split up into several peaks. A function called `combphases` has been created that can combine these linear phases.



a) linear phase b) second degree cylindrical paraboloid c) third degree cylindrical paraboloid d) Spherical paraboloid

Figure 3: Examples of Phase profiles

3.2.3 combphases function

To combine several phase profiles, a function called `combphases` has been created, which is shown in appendix D. This function can combine several phase profiles by alternating between pixels. It can combine up to 16 phase profiles. These phase profiles are defined within the function. Initially, sixteen matrices are created filled with zeros called `M1` to `M16`. The phase profiles that are to be combined can then be put into the first few matrices, depending on how many phase profiles will be combined. For example, if four phase profiles are combined, these profiles are defined in `M1` to `M4`, and `M5` to `M16` remain zero matrices. Some default phase profiles are created in the beginning of the code, shown in appendix E.1. These default phase profiles can be used to create the desired phase profiles. These profiles can then be assigned to the matrices `M1` to `M16`, and the code will then combine these phase profiles by alternating between pixels.

The default phase profiles are four linear phases with height `H` that point in different directions, and three curved profiles that have different shapes. The variable `H` is equivalent to the phase added by the SLM, but is the not normalized value. This variable is used in the figures to make the shape of the phase profile better visible, since the shape of the phase profile is not obvious when subtracting the largest multiple of 2π from each value. The linear phases are created using the commands `linspace` and `repmat`. `linspace` creates a list of linearly increasing numbers ranging from 0 to `H`, and `repmat` repeats this list such that a matrix of the size nX x mY is created. These linear phases are called `Left`, `Right`, `Up` and `Down`. The curved profiles that are created have the shape of a second degree and third degree cylindrical paraboloid and a spherical paraboloid. The phase profile of the second degree and third degree paraboloid are created by making a list of linearly increasing numbers ranging from -H to H. Every element in the list is squared or tripled and then the list is repeated to become a matrix using the `repmat` command. The resulting matrices are called `Second_Degree_Cylindrical_Paraboloid` and `Third_Degree_Cylindrical_Paraboloid`. The spherical paraboloid is created using a double for loop that run through all x and y values. This matrix is called `Spherical_Paraboloid` Another phase called `Center` contains a matrix with only zeros. How these profiles are created in the code can be found in appendix E.1. By combining these phases all different kinds of patterns can be created. For example, by adding `Left` to `Up`, the phase points to the upper right corner.

The input values of this function are `nX` and `mY`, which represent the dimensions of the matrix. The number of linear phases that will be combined is determined by `numphases`. This number has to be between 0 and 16. The other input variables of the function are `shape`, `boxsize`, `H`, `Phase` and `theta`. These variables will be explained below.

The last three input variables determine the properties of the individual phase profiles that will be combined. The value of `H`, which stands for height, determines the gradient of the default phase profiles. The variable `theta` can be used to rotate the phases. For example in the following way: `sin(theta) * Left` + `cos(theta) * Up`. By changing the value of `theta`, the linear phase will rotate. This is useful when making the animations, or when creating a circular pattern for the peaks at the focus. The variable `Phase` is used to add a plane phase profile to one of the phase profiles. By doing so, a peak at the focus is delayed in time.

The input variable `boxsize` determines how big the set of pixels is between which the phase profiles alternate. The variable `shape` determines through which pattern the matrices are combined. The phase profiles can be combined through horizontal, vertical or diagonal lines. The input value for `shape` to do this are `'hor'`, `'ver'` and `'dia'`, respectively. This can be applied to any number of linear phases. When the number of phases is a multiple of two, three or four, the phases can be combined in squares. Figure 26d) portrays hows this is done. The input value for `shape` for this is `'sq2'`, `'sq3'` and `'sq4'`. The last number determines the vertical dimension of the square. When combining four linear phases, the square has to be dimension 2X2, do the input value `'sq2'` has to be used. When combining six linear phases, they can be combined through either a 3x2 square or a 2x3 square. To get a 3x2 square, the input value `'sq3'` has to be used, and to get a 2x3 square, the input value `'sq2'` has to be used.

Figure 4 shows how the phase profiles are combined. Each color represents a different phase profile. In this example, the number of phase profiles is four, but in principle it can be done for any number. The function supports a maximum of 16 phase profiles. Figure 26a) shows the horizontal shape, figure 26b) shows the vertical shape, 26c) shows the diagonal shape and 26d) shows the square shape. In this case, the input value for the square shape is `'sq2'`.

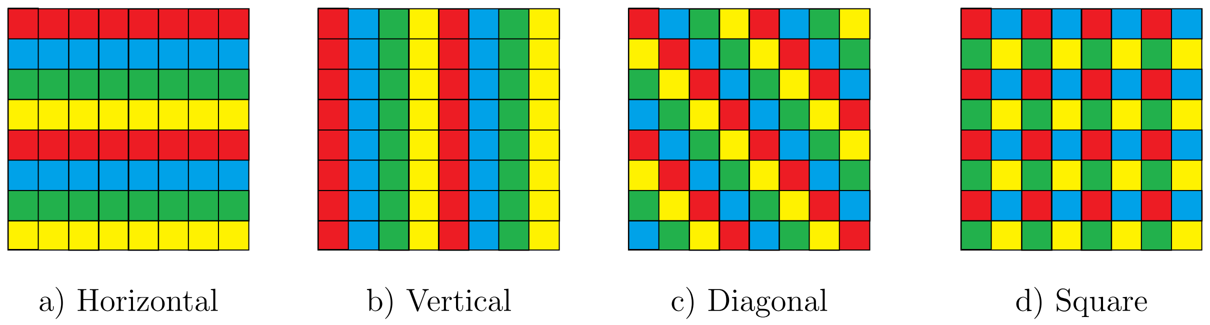


Figure 4: Different patterns through which four phase profiles can be combined. Every box represents a pixel on the SLM. The different colours represent different phase profiles.

After the Matrix with the phase profile has been created, the largest multiple of 2π is subtracted for each value in the matrix. This is because the phase is a periodic value, and the SLM can only add a phase between 0 and 2π. This is done by using the `fix` function which is built in by matlab. This function removes all the decimals of the input number and returns the decimal number rounded down to an integer. Subtracting `2*pi * fix(phase_profile/2*pi)` from `phase_profile` then subtracts the largest multiple of 2π from each element in the matrix. Figure 24 in appendix A.2.1 shows what a linear phase looks like after subtracting the largest multiple of 2π from each value. In this figure the value of `H` is 30.

Some test were run to confirm whether the profile at the focus was the same for the original phases profile and the profile after the largest multiple of 2π was subtracted.

3.2.4 Add profiles and execute Fourier transform

Now the phase is added to the list and the Fourier transformation is executed. In order to add phase to a Gaussian profile, equation 10 is expanded to equation 11 below.

$$A \cdot \exp\left(-\pi \frac{(x-x_0)^2 + (y-y_0)^2}{2\sigma^2} + i\varphi \right) \tag{11}$$

Here, φ is the phase. The matrix `phase_profile` is added to `input_profile` by element wise multiplication, where each element of `input_profile` is multiplied by $\exp(i\varphi)$ for φ in `phase_profile`. This matrix is called `input`

In order to execute the Fourier transformation, the FFT algorithm is applied in two dimensions. This is done by using the `fft2` command, which is built in by Matlab. This command takes in a matrix and returns a Fourier shifted matrix of the same size. The `fft2` command is applied to both `input` and `input_profile`. The results are called `focus_profile` and `focus_profile_nophase`. Both lists are made to compare what the focus looks like with and without the phase. This gives a better insight to what the exact effect is of the phase shift. The inverse Fourier transformation is then applied to `focus_profile` by using the `ifft2` command. This is to check if we get the same profile back. This matrix is called `inverse_focus_profile_plot`.

3.2.5 Plotting the profiles

The last part of the code plots the matrices. In order to plot the profiles at the focus, the amplitude of the complex number is needed and the Fourier shift has to be applied. The absolute value is taken with the `abs` command. For a complex number, this command adds the real part squared and the imaginary part squared, and takes the square root. The Fourier shift is then applied by a self written function called `fliplist2D`, which can be found in appendix F. This function divides the matrix in four evenly sized parts, and then exchanges the top left and lower right part with each other, and does the same for the top right and lower left part. In order to calculate the relative intensity, the maximum value of the profile at the focus with no phase shift is calculated. This value is called `max_nophase`. Every value in `focus_profile` is divided by `max_nophase` to plot the relative intensity. The matrices obtained after these operations are called `focus_profile_plot`, `focus_profile_nophase_plot` and `inverse_focus_profile_plot`.

Eventually the code plots 5 figures. Table The plots are made by using the `surf` command, which plots the matrix as a surface in 3 dimensions. By using the command `view([ 1 0 0])`, the angle is changed to look at the surface from directly above. This can be useful when analyzing the profiles. The `colorbar` command is used to show the intensities that correspond to the colours.

3.3 Testing phase profiles

The default input intensity for testing the different phase profiles has a Gaussian shape, and is oriented at the center. This profile is plotted in figure 5a). The value of σ is 0.5, and the amplitude is 1.
In order to combine the phases, they have to be defined in the function `combphases`. This is done by inserting them in line 20 in the code shown in appendix D. Appendix E shows the codes used to define these phases. Appendix E.1 shows the default phase profiles, and different combinations of these default phases are shown in the appendices below this. The default value of `H` is 300, the default value for `Boxsize` is 1 and for `theta` and `Phase` the default value is 0. For `nX` and `mY` the default values are the dimensions of the SLM, which are 600 by 792.

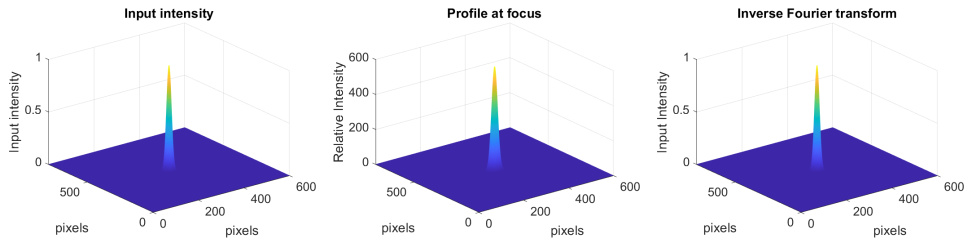
4 Results

This section shows the results that were obtained using the code. The intensity at the focus is plotted relative to the maximum intensity of the peak at the focus when no phase shift is applied. When the default input intensity profile is used, the peak at the focus has a height of 592 expressed in units of the input intensity.

When several subfigures are shown with one colorbar in the figure, this colorbar applies to all the subfigures. In this case, the colorbar will be plotted either on the right of all figures or below all figures. The units on the lower axis of the input intensity are pixels, the profile at the focus is expressed in arbitrary units that are in the inverse length Fourier plane. The size of these units depends on the setup.

4.1 Focusing the beam without a phase profile applied

To confirm that the code gives correct results the code was run with no phase added. By doing an inverse Fourier transformation afterwards it was checked if the same profile as the input profile was reconstructed. Figure 5a) below shows a Gaussian input profile with no phase applied. Figure 5b) shows the Fourier transform and figure 5c) shows the inverse Fourier transform. Since 5a) and 5c) are identical, the code gives correct results.



a) Input intensity   b) Profile at focus   c) inverse Fourier transform of profile at focus

Figure 5: Testing whether the code gives correct result with simple Gaussian profile

4.1.1 Changing input intensity profile

By changing the size of the input intensity profile, the size of the profile at the focus changes. Figure 6 shows different sizes of input intensities and the corresponding profile at the focus. The size of the input intensity is changed by changing the value of `sigma` in the code. The figures on top show the input intensity profile and the figures below show the profile at the focus for this input intensity. The dotted white lines represent where intensity has decreases to 10% of the peak intensities. These lines visualise how the peak size changes. The value of σ ranges from 0.3 to 2. As the value of sigma increases, the circumference of the peak in the input intensity increases and the circumference of the peak in at the focus decreases. When σ is equal to 0.5, the circumferences of the peak in the input intensity profile and the peak at the focus are approximately equal.

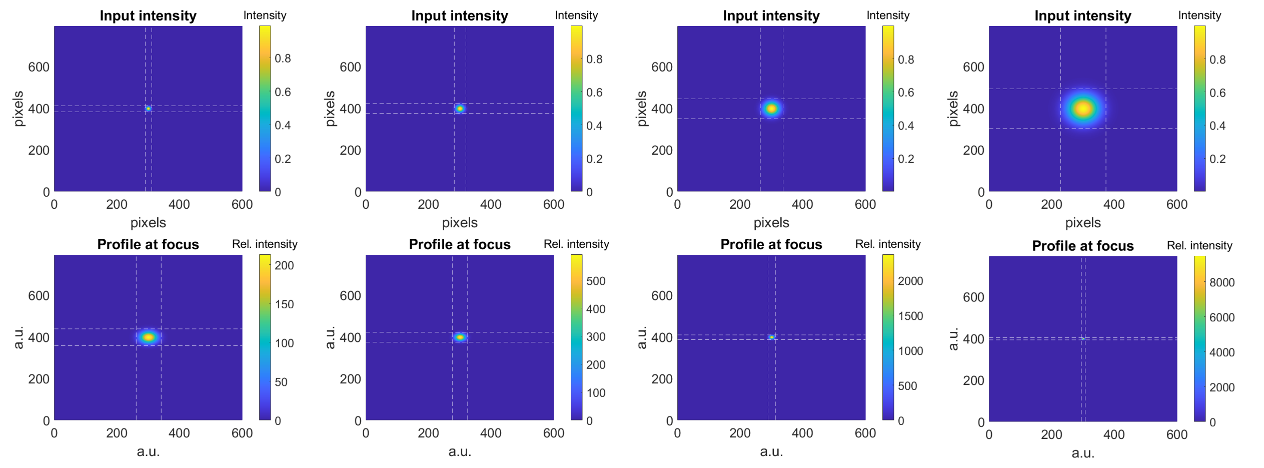

a) $\sigma = 0.3$ b) $\sigma = 0.5$ c) $\sigma = 1$ d) $\sigma = 2$

Figure 6: The figures on top show the input intensity profiles and the figures on the bottom show the corresponding profile at the focus. The intensity is expressed in units of the height of the input intensity.

4.2 Linear phase

Figure 7 shows the results of applying a linear phase. Appendix E.2 shows how the phase profile was built up in the function `combphases`. Figure 7a) shows the phase profile of a linear input phase. Figure 7b) shows the spatial profile of the beam at the focus with no applied phase. Figure 7c) shows the profile at the focus with a linear phase applied with H equal to 300. Figure 7d) shows the same but then with H equal 600. From these figures it can be concluded that the peak moves upwards when a linear phase pointing up is applied. The larger the value of H, the further the peak moves upwards.

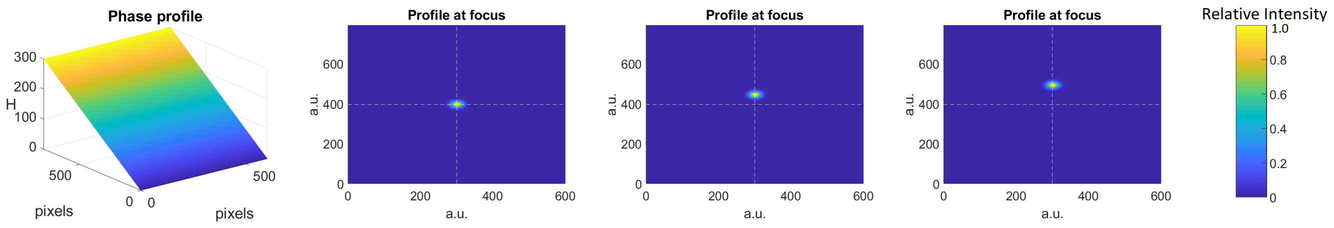

a) Input phase profile b) H = 0 c) H = 300 d) H = 600

Figure 7: The effect of a linear phase pointing upwards for different values of H. The colorbar applies to all three figures.

4.2.1 Adding Linear phases

Figure 7 above shows the result of one linear phase pointing up. By adding linear phases in different directions, the peak at the focus can be moved around. For example, a linear phase pointing to the left and a linear phase pointing up can be added. Appendix E.3 shows how this is done in code. Figure 29b) in appendix A.3.1 shows the result of adding two linear phases, one pointing up and one pointing left. Figure 29a) shows the profile focus with no phase as a reference. For both linear phases H is 600. In the focus, the effect of the linear phase pointing left and the linear phase pointing up are added, and the peak appears in the upper left corner.

The peak can follow a circular path by creating a linear phase that is the sum of `sin(theta) * Right` + `cos(theta) * Up`. By adjusting the value of `theta`, the peak moves in a circular path. Appendix E.4 shows how this is implemented in the code. Figure 30 in appendix A.3.2 shows how the peaks moves around. In this figure H is 600. The dashed white line represents the trajectory of the peak.

4.3 Curved phase profile

Figure 8 shows the result of applying different curved phase profiles. The shapes that are used are a second degree and third degree cylindrical paraboloids and a spherical second degree paraboloid. The upper figures show the phase profile and the lower figure shows the profile at the focus. Appendix E.5 shows how these profiles are created in the code.

Figure 8a) shows the result of applying a phase profile with the shape of a second degree cylindrical paraboloid as a phase profile. In this figure `H` is 3000. At the focus the peak stretches out along the axis orthogonal to the cylindrical axis. Figure 8b) shows the result for applying a phase profile with the shape of a third degree cylindrical paraboloid. In this figure H is 5000. Here, the peak is stretched out along the axis orthogonal to the cylindrical axis. In the direction where the sheet moves down, the peak does not stretch out, and in the direction where the sheet moves up the peak stretches out oscillating in intensity. Figure 8c) shows the result of applying a phase profile with the shape of a spherical paraboloid. H is 6000 in this figure. Here the peak stretches out in all directions.

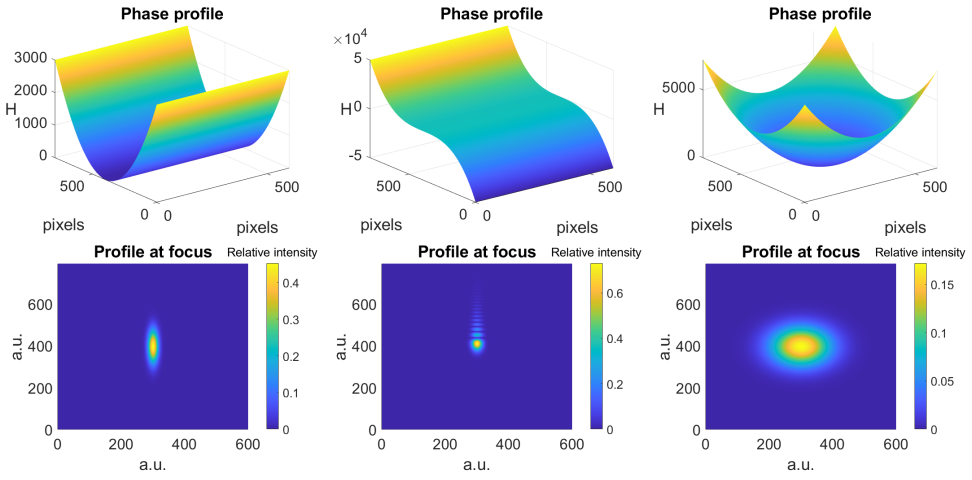
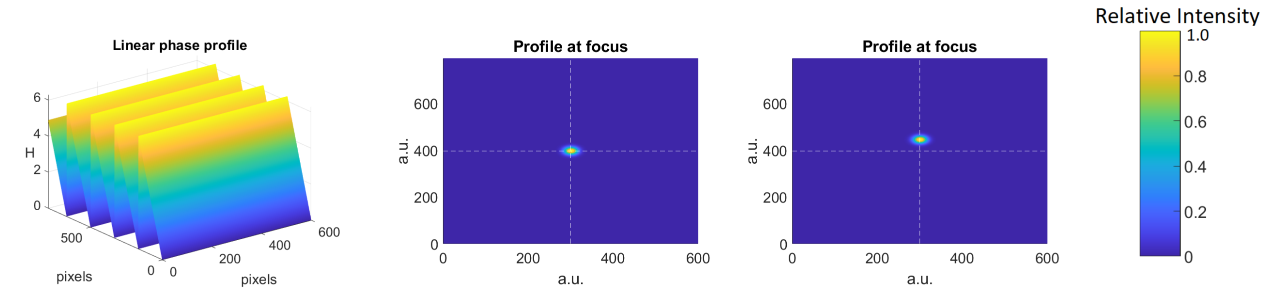


Figure 8: Different curved phase profiles

4.4 Subtracting the largest multiple of 2π

Figure 9 shows the results of subtracting the largest multiple of 2π from each element in the linear phase profile. Figure 9a) shows what the phase looks like when H is 30 and the largest multiple of 2π is subtracted. In figure 9b) and c), the value of H is 300. The focus at the profile is the same in figure 9c) and figure 7c). This shows that subtracting the largest multiple of 2π from each value in the phase does not affect the profile at the focus.



a) Phase profile b) No applied phase c) Profile at focus with phase profile figure a)

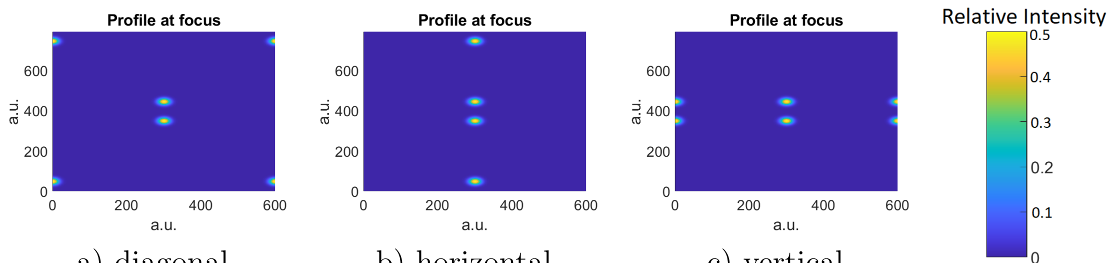
Figure 9: Testing whether subtracting largest multiple of 2π from phase gives different results

4.5 Combining linear phases

The function `combphases` can combine different phase profiles through alternating between pixels. By combining linear phases that point in different directions, the peak at the focus splits up into several peaks. The sections below shows some results obtained by combining linear phases.

4.5.1 Two Linear Phases

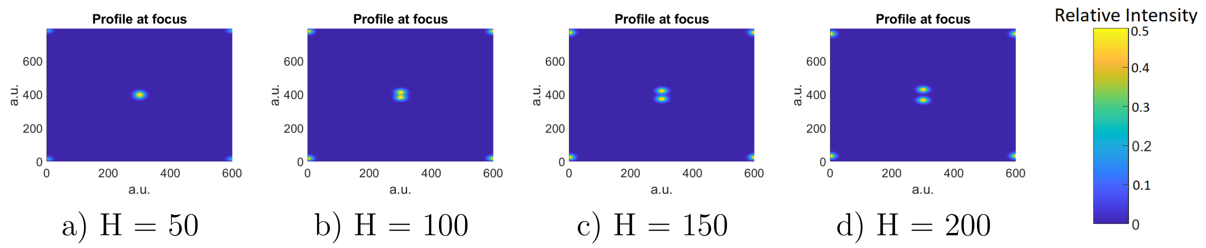
Figure 10 shows the results of using the `combphases` function to combine two linear phases in opposite direction, one pointing up and one is pointing down. Appendix E.6 shows how these linear phases are created. Figure 10 shows the profile at the focus for different combination patterns. The combination patterns are displayed in figure 25 in appendix A.2.2, where the different colours represent the two linear phases in opposite direction. As predicted, the two peaks move apart. However, extra peaks appear. Their position depends on the pattern through which the phases are combined.



a) diagonal b) horizontal c) vertical

Figure 10: Two linear phases combined through different patterns

In order to split up the peaks, the value of `H` has to be sufficiently large for the peaks to be considered as two separate peaks. Figure 11 shows the two patterns combined through diagonal lines with the value of `H` increasing. The peaks shows the same behaviour of a single peak in figure 7, namely they move further apart as the value of `H` increases.

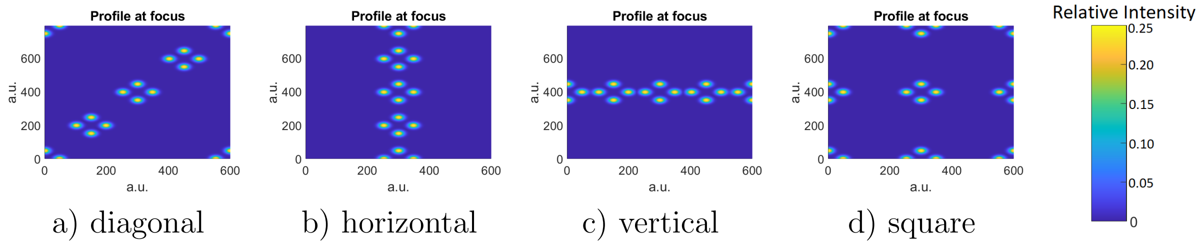


a) H = 50 b) H = 100 c) H = 150 d) H = 200

Figure 11: Two linear phases with increasing values of H

4.5.2 Four Linear Phases

Figure 12 shows the results of using the `combphases` function to combine four linear phases. The phases that are combined are pointing up, down, to the left and to the right. Appendix E.9 shows the how the linear phase profiles are created. The phases can be combined in four ways, namely through horizontal lines, vertical lines, diagonal lines and through squares of size 2x2. Figure 12 shows the beam profile at the focus for these four different combinations. Figure 26 in appendix A.2.2 shows how the phase profiles are combined, respectively. Each color represents a different phase profile, and each box represents a pixel. The figures below show the profile at the focus.



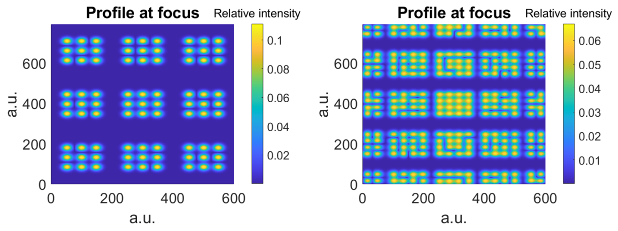
a) diagonal b) horizontal c) vertical d) square

Figure 12: Four Linear Phases combined through different patterns

Figure 12 shows how the pattern through which the phase profiles are combined influences the shape of the profile at the focus. The four peaks at the center, which are the peaks that will eventually be used in the experiment, are identical regardless of how the phase profiles are combined. The extra peaks appear in the same orientation with respect to each other as the center peaks. The center peaks appear in a diamond shape of four, and the extra peaks also appear in a diamond shape of four. In each picture, the total number of peaks is 16. The position of the peaks depends on the pattern through which the phase profiles are combined.

4.5.3 N linear phases

The function `combphases` can combine up to sixteen phase profiles which means that up to sixteen main peaks can be created at the focus. The position of every single peak can be controlled by changing the height and direction of the linear phase. This gives the possibility to create many different orientations of peaks at the focus. Figure 13 shows a square of nine peaks and a square of sixteen peaks. This is done through combining linear phases. Appendix E.14 and appendix E.16 shows how these phase profiles are built up in the code. The phase profiles are combined through squares. For nine peaks this is done by using `sq3` for the shape and H equal to 300, for sixteen peaks the shape is `sq4` and H is equal to 100. For bigger values of H the main peaks start to overlap with the extra peaks.

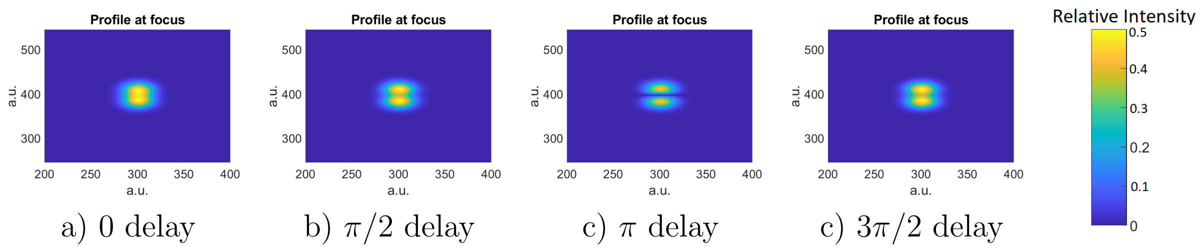


a) nine linear phases b) sixteen linear phases

Figure 13: Nine and sixteen linear phases combined to create squares

4.6 Time delay

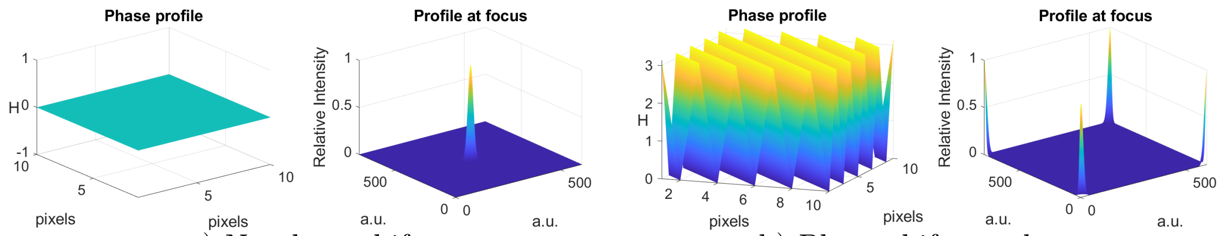
One of the peaks can be delayed in time by adding a plane phase to one of the linear phases. A plane phase is a uniform phase profile. A plane phase does not change the position of the peaks, but it does change the way the peaks interact with each other when they are close to each other. Figure 31 in appendix A.3.3 shows what the phase profile looks like for two linear phases, one pointing up and one pointing down. Figure 31a) shows the normal case, and in figure 31b) one peak is delayed with 11π. The figure shows the profile of the phases looking from the side. Since the phase is periodic, the effect from a delay of 11π is the same as the effect of a delay of π, which is equivalent to half a cycle. A delay of 11π is used in the figure since this makes the phase difference more visible in the figure. The results of the added plane phase are shown in figure 14. The figures are zoomed in to make the effect of the delay more visible.



a) 0 delay b) $\pi/2$ delay c) π delay c) $3\pi/2$ delay

Figure 14: Profile at the focus for different delays in lower peak

The peaks can be moved closer together such that they completely overlap. This is shown in figure 15 below. In this case H is zero, which means that there is no tilt in the phase profiles and they are flat. The two peaks then overlap at the center. The profiles are combined through diagonal lines. When there is no phase difference, the two linear phases are both flat phases at the same height, so the final phase profile will be a uniform phase. At the focus the peaks add up and the extra peaks disappear, creating the same profile as when there are no phase profile applied. When the two linear phases are half a cycle out of phase, the phase profiles are two uniform phases but at different heights. The final phase profile consists of diagonal lines alternating between 0 and π. When this phase profile is applied the peaks at the center disappear and the extra peaks double in intensity. In figure 15, the figure on top shows the phase profile, and the figure below shows the profile at the focus. The phase profile is zoomed to make the pattern better visible.



a) No phase shift b) Phase shift equal to π

Figure 15: Two linear phases combined with H=0

4.7 Extra peaks

When the `combphases` function is used to combine linear phases, extra peaks appear. In order to create several peaks at the focus, the behaviour of the extra peaks has to be well understood. The sections below show the behaviour of the peaks with regard to height of the linear phase, the number of peaks, the pattern through which the phases are combined and the phase of the peaks.

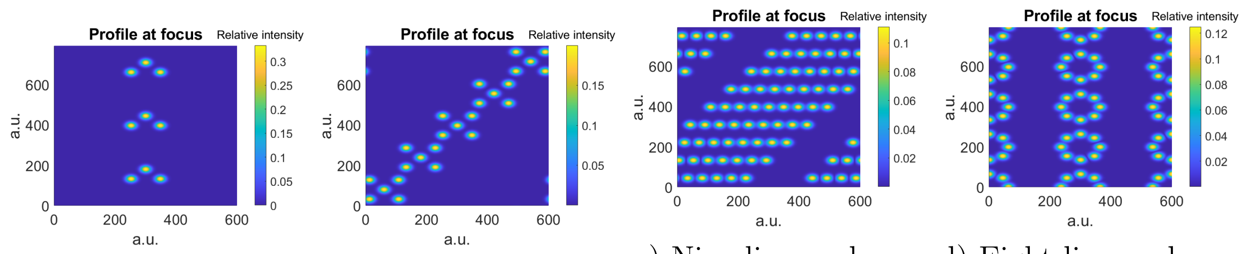
4.7.1 Gradient of the linear phase

Figure 10a), b) and c) shows the profile at the focus when combining two linear phases. In each case, two extra peaks appear. In figure 10c), there are four half peaks, giving two peaks in total. This figure is used to study the behaviour of the peaks with regard to the value of `H`. The two upper peaks and the two lower peaks have the same height. The distance of the two main peaks to the center is the same as the distance from the extra peaks to the top. As the two main peaks move up, the extra peaks move down. By increasing the gradient of the linear phase, which is done by increasing `H`, the main peaks move further apart and the extra peaks move closer together. To understand how this works, a new variable called the step size is introduced. The step size is the difference in height between each pixel. The step size can be adjusted by setting `H` to `(mY - 1) * S`, where `S` is the step size.

Figure 32 in appendix A.3.4 shows how the peaks move apart depending on the step size. It was found that the main peaks and the extra peaks were at the same height when the step size is $\pi/2$. When the step size is π the extra peaks merge in the middle and the main peaks disappear. This is the point where the peaks have a maximum separation. When the step size is $3\pi/2$, the same result is obtained as for a step size of $\pi/2$, meaning that the peaks move backwards. When the step size is 2π, the results is the same for no phase, and the extra peaks disappear.

4.7.2 Number of extra peaks

The number of extra peaks depends on the number of linear phases that is combined. In the case of two linear phases, there are four peaks in total. When four linear phases are combined there are sixteen peaks in total. Figure 16 demonstrates how the number of extra peaks changes. Figure 16a) shows three linear phases that create three peaks forming a triangle. The phase profiles are combined horizontally. Here a total of nine peaks appear. Figure 16b) show five linear phases in a square with one peak in the middle. The phase profile are combined diagonally. In this figure there are 25 peaks in total. Figure 16c) shows nine linear phases forming a line of peaks. Here the phase profiles are combined diagonally. This figure has 81 peaks in total. figure 16d) shows eight linear phases forming a circle. The phase profiles are combined trough rectangles of dimension 4x2, and H is equal to 400. This figure has 64 peaks in total. Figure 13b) displays sixteen peaks forming a square. This figure has 256 peaks in total. In all figures it holds that the total amount of peaks that appears at the focus is equal to the number of linear phases that is combined squared. The pattern through which the phases are combined is different in each case. Therefore, this can be regarded as a general rule.



a) Three linear phases b) Five linear phases c) Nine linear phases making a line d) Eight linear phases making a circle

Figure 16: Profile at the focus for different numbers of peaks

4.7.3 Pattern through which linear phases are combined

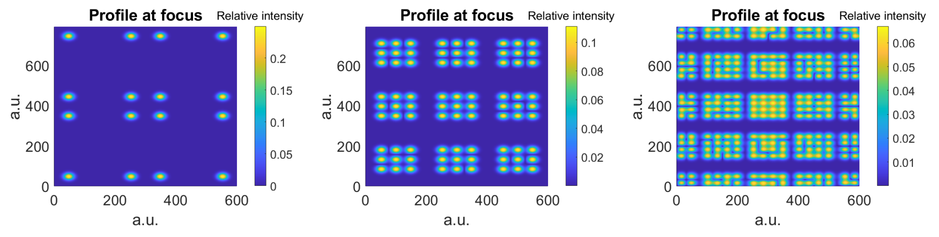
The linear phases can be combined through lines or through squares. The phases can be combined through lines for any number of phases, but in order to combine the linear phases through squares, the number of phases has to be a multiple of two, three or four. Figure 12 shows how the position of the extra peaks changes as a result of the pattern through which the phases are combined. The extra peaks always appear in the same shape as the main peaks, but their position depends on the pattern through which the phases are combined.

Different orientations of lines

The linear phases can be combined in lines that are either diagonal lines, horizontal lines or vertical lines. In figure 12a), the phase profiles are combined through diagonal lines running from the top left to the bottom right. The extra peaks in the focus are oriented in a line that runs from the top right to the bottom left. In figure 12b), the phase profiles are combined through horizontal lines and the extra peaks are oriented in a vertical line, and in figure 12c) the phase profiles are combined through vertical lines and the extra peaks are oriented in a horizontal line. The same behaviour is observed when using a different number of phases combined. Therefore, in the case of combining the linear phases through lines, the extra peaks align orthogonal to the direction of the lines that are combining the phases.

Different square shapes

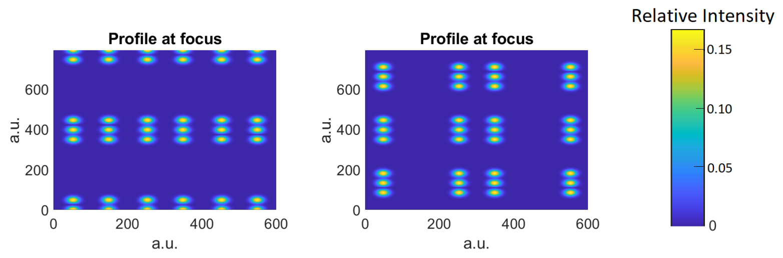
When the linear phases are combined through squares, the peaks appear in the corners and on the sides. In the case of square numbers such as four, nine and sixteen, the phases can be combined through a square that has the same height and width, as shown in figure 27 in appendix D. These combination patterns lead to symmetry in the patterns of the extra peaks, meaning that the pattern of the original peaks appears in the same amount in the horizontal direction as the vertical direction.



a) Four linear phases b) Nine linear phases c) Sixteen linear phases

Figure 17: Square numbers

For the case of non-square numbers, the squares through which the linear phases are combined can have different dimensions. For example, six phases can be combined through a 2x3 square or a 3x2 square. This will lead to different positions of the extra peaks. Figure 18 shows what the profile looks like in this case. When using a square of size 2x3, which is displayed in figure 18a), the pattern of six peaks appears three times in horizontal direction, and two times in vertical direction. When using a square of size 3X2, the opposite happens, namely the pattern appears three times in vertical direction and two times in horizontal direction. The combinations patterns of the phase profiles are displayed in figure 28 in appendix A.2.2

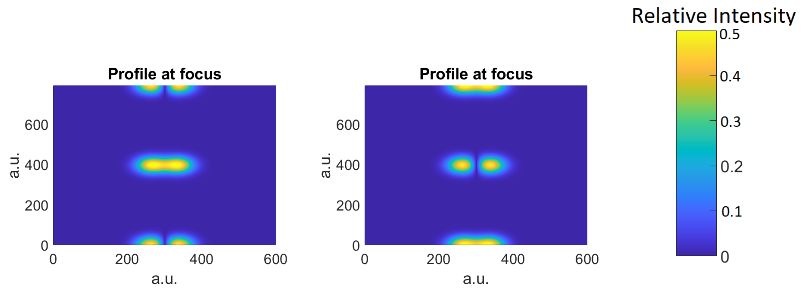


a) 2x3 a) 3x2

Figure 18: Different dimensions of squares through which the phase profiles are combined

4.7.4 Phase difference extra peaks

Section 4.6 shows how adding a time delay to one of the phase profiles causes the peaks at the focus to be out of phase. When the peaks are out of phase by exactly half a cycle, meaning that they have a phase difference of π, a diffraction pattern appears between the peaks. The phase difference between the extra peaks can be determined by studying the diffraction pattern between the extra peaks. Figure 19 shows the profile at the focus where the peaks are split up horizontally. The phase profiles are combined through horizontal lines are a sufficiently close such that the diffraction pattern can be observed. The value of σ in the input intensity profile is 0.2. This is done to make the peaks bigger and the diffraction pattern more visible. The value of H is 220. Figure 19a) shows two peaks that have the same phase at the center and figure 19b) shows two peaks that have a phase difference of π at the center. Appendix E.7 shows how the phases are created in the code. In figure 19a) the value of `Phase` is 0 and in figure 19b) the value of `Phase` is π. In figure 19a) the extra peaks show a diffraction pattern, which means that the peaks have a phase difference of π. In figure 19b) the extra peaks do not show a diffraction pattern, which means that the peaks have no phase difference.



a) no phase difference b) phase difference π

Figure 19: Phase difference extra peaks

4.8 Increasing the dimension of the SLM

The dimensions of the SLM are 600x792. The code can be used to investigate what the advantages are when using a SLM that has more pixels. The number of pixels of the SLM is changed in the code by changing the values of `nX` and `mY`. Figure 20 below shows sixteen linear phases forming a four by four square. Figure 20 shows how the figure changes when the number of pixels on the SLM increases. In this simulation, the total size of the SLM remains constant. This means that when the number of pixels on the SLM increases, the size of the pixels decreases. The figure shows that the space between the extra peaks and the original peaks increases as the amount of pixels in the SLM increases. This simulation is purely hypothetical since there is are SLM's with these dimensions in the lab. However, these results give an indication of what advantages a SLM with larger dimensions would have.

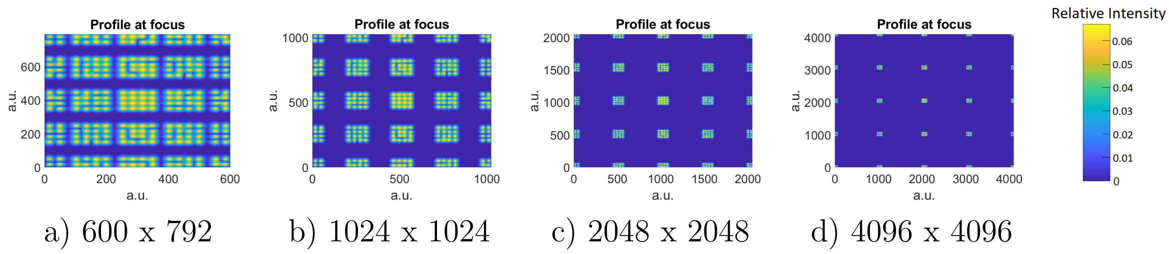


Figure 20: Increasing the dimensions of the SLM

4.9 Effect of size input intensity profile on multiple peaks

As shown in section 4.1.1, the size of the peak at the focus decreases as the size of the peak of the input intensity increases. Figure 21 below shows how increasing the size of the input intensity changes the size of the peaks at the focus when a phase profile is applied that consists of sixteen linear phases. The peaks become smaller but their position with respect to each other does not change.

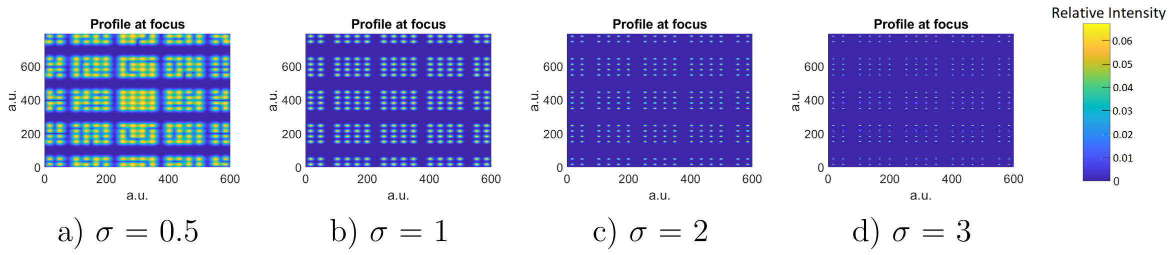


Figure 21: Increasing the size of the input intensity profile when applying phase profile combining sixteen linear phases

5 Discussion

5.1 Changing input intensity profiles

It is shown that when the size of the input intensity profile increases, the size of the peak at the focus decreases. This is the same as focusing a Gaussian beam with a lens. The relation between the beam diameter of the input intensity (D) and at the focus (D') is the following [11]:

$$2D \approx \frac{4\lambda f}{\pi D'} \tag{12}$$

This relation shows that the diameter of the beam at the focus is inversely proportional to the diameter of the input beam. This relation corresponds to the results obtained in section 4.1.1 and section 4.9.

5.2 Periodicity phase

In section 4.4 it was observed that subtracting the largest multiple of 2π from each value in the phase profiles matrix does not affect the profile at the focus. This is because the wave is periodic and has a phase of 2π. The advantage of this is that the SLM does not have to be able to add a phase larger than 2π. This explains why the extra peaks disappear in figure 32f). Here, the step size is equal to 2π, but since the largest multiple of 2π can be subtracted this is equivalent to a flat phase. Therefore the profile at the focus is the same as in figure 32a) where no phase is applied.

5.3 Origin extra peaks

The origin of the extra peaks can be found in the frequencies of the Fourier transformed phase image that represent intensity changes across the phase profiles. One way to understand the extra peaks is to see the pattern through which the phase profiles are combined as a diffraction grating. When combining the phase profiles through lines the peaks align orthogonal to the lines through which the phase profiles are combined, which is exactly how a diffraction grating behaves. Combining the phase profiles through squares can be seen as combining diffraction gratings. This explains why the peaks appear in the corners. A different way to explain the peaks is with Fourier theory. The extra peaks can be seen as a repetition of the Fourier domain.

5.4 Phase of the peaks

Section 4.6 shows how the interaction between two close peaks changes when one of them is delayed. When there is no phase difference and the peaks are close, their intensities are added in the areas where the peaks overlap. When the phase difference is π, the intensities cancel each other out. This is because one peak arrived half a cycle after the other peak, and is in the exact opposite phase of the earlier peak. Due to destructive interference, the peaks will cancel each other out in the parts where they overlap.
When combining two linear phase profiles and their gradient is zero, the two peaks have the same position. This can be seen in figure 15. When the phase difference is zero, the two phase profiles are the same and they form a flat phase at zero. Therefore, the profile at the focus is a Gaussian profile. When there is a phase difference, the phase profile has diagonal lines alternating between 0 and π. This works as a diffraction grating. At the focus, the peaks at the center cancel each other out and the extra peaks appear at the corners. This is the same result as when using a diffraction grating.

5.4.1 Phase of the extra peaks

Figure 19 shows the diffraction pattern between the extra peaks. In figure 19a), the main peaks are in phase and a diffraction line is visible between the extra peaks, which means that they are exactly π out of phase. In figure 19b) the main peaks are π out of phase and the extra peaks are in phase. This means that the phase difference between the extra peaks is exactly opposite of the phase difference between the main peaks. An explanation for why the extra peaks have the exact opposite phase of the main peaks is energy conservation. When the main peaks are in phase their intensities are added when the peaks overlap. The extra peaks are out of phase, so their intensities cancel out when they overlap. The energy of the extra peaks therefore gets conserved by the main peaks, and the total energy in the profile does not change. When the main peaks are out of phase and overlap, they cancel each other out while the intensities of the extra peaks add up, meaning that the energy of the main peaks is conserved in the extra peaks.

5.5 Controlling the shape of the peaks

The curved profiles change the shape of the peaks at the focus. The curved profiles that were tested in this project stretched out the peaks at the focus. For the third degree cylindrical paraboloid, the peak stretched out in an oscillating way. This can be explained by comparing the the third degree cylindrical paraboloid to a linear phase profile. A linear phase profile has a uniform gradient which adds the same shift to the entire peak. The third degree cylindrical paraboloid has a gradient that is positive or zero at all points, but is not uniform over the entire surface. Therefore the peak is stretched out but the phase is not uniform. This will lead to constructive interference at some points, and negative interference in other points, which results in the oscillating pattern.
The XUV pulses are steered by using the part of the Gaussian profile of the control pulse that can be approximated by a linearly increasing line. By stretching the pulse, the gradient of this straight line can be adjusted. With this, the direction of the XUV pulse can be controlled. The pulse can also be stretched out by using a different shape of the input intensity. However, by changing the shape of the intensity profile, only the size of the peak at the focus can be controlled. When using a curved profile, it is possible to stretch out the pulse in only one direction, which conserves energy.

5.6 Avoiding overlapping of center peaks and extra peaks

The extra peaks that appear when combining phase profiles are a disturbance when executing experiments. The number of extra peaks is equal to the square of the number of the original peaks, so the number of extra peaks can become very big. This makes it likely for the extra peaks to start overlapping with the peaks at the center. To avoid this the extra peaks must be oriented such that they are sufficiently far away from the center peaks to not disturb the experiment. The sections below discuss some ways to minimize this disturbance.

5.6.1 Reducing gradient Linear phase

The gradient is reduced by using a low value of `H`. In this case the peaks at the center are closer together and the extra peaks will stay further away from the peaks at the center. However, when the peaks are too close to each other they will start to interact with each other. The peaks will add up when they are in phase and cancel each other out when they are half a cycle out of phase. These interference patterns change the shape of the peaks.

Therefore, the value of `H` can be reduced only up until a certain point. In the results section an input intensity profile is used where σ is equal to 0.5. Figure 11 shows that for this input intensity the peaks at the center start to overlap with each other when `H` is smaller than 150. To combine sixteen linear phases, the value of `H` has to be 100, which means that the peaks are showing diffraction patterns. This can be seen in figure 13b). Here `H` can not be increased since the extra peaks will overlap with the peaks at the center.

5.6.2 Using most efficient pattern for combining phase profiles

To use the peaks at the center for the experiment the extra peaks can not overlap with the peaks at the center. It is desirable to have as much distance as possible between the center peaks and the extra peaks. This can be achieved by using the most efficient pattern through which the phase profiles are combined.

In general the center peaks form a pattern that has approximately the same height as length, such as a square or a circle. In this case square shapes are most efficient, since this will lead to the extra peaks being distributed over the whole profile and not on a line. Using a larger surface to distribute peaks over will keep the largest distance between the center peaks an the extra peaks. In order use square shapes to combine the phase profiles, the number of peaks has to be a multiple of two, three or four. Square numbers are most efficient since the phase profiles can be combined through squares that have the same height and width, which results in the sets of extra peaks being evenly distributed in horizontal and vertical direction, so that they are placed as far away as possible from each other.
Numbers that are multiples of two, three and four can best be combined through squares that have approximately the same height and width. For example, when combining twelve linear phases, it is more efficient to use squares of size 3x4 than using squares of size 2x6. When using squares of size 3x4, the set of peaks will appear four times in horizontal direction and three times in the vertical direction. When using squares of size 2x6, the set of peaks will appear six times along the in the horizontal direction and two times along the vertical direction. There will be more space between the peaks when using squares of size 3x4.
Prime numbers are least efficient since they can only be combined through lines. This means that the extra peaks will appear in a line which will leave little space between the sets of peaks making it likely for them to overlap.

When the center peaks form a pattern that does not have the same height and length, such as a line of peaks as in figure 16c), it can be more favourable to combine the phase profiles through lines. When creating a shape that is wide but not high, it is most efficient to combine the phase profiles through horizontal lines. The sets of extra peaks will then appear in a vertical line, which will prevent the peaks from overlapping horizontally.

5.7 Using lab equipment most efficiently

5.7.1 Changing intensity

In order to be able to combine more peaks, the peaks need to be brought closer together. This can be achieved by decreasing the size of the peaks at the focus. The size of the peak in the input intensity profile determines the size of the peaks at the focus. Therefore, to combine more peaks the size of input intensity profile has to be increased.

To use the SLM as efficient as possible, the peak of the input intensity profile has to be as large as possible. When using a small peak, the largest part of the intensity profile is flat and a big part of the pixels on the SLM is unused. Therefore, it is most efficient to use a big peak for the input intensity profile. This will ensure that the peaks at the focus are small enough to avoid peaks from overlapping, and the SLM will be used most efficient.

5.7.2 Resolution SLM

A large number of pixels on the SLM will result in more space between the main peaks and the extra peaks. The SLM that is used in this experiment has 792x600 pixels, which is enough for combining two or four linear phases. However, when more peaks are needed at the focus, the extra peaks will likely be a problem. This problem can be solved by using a SLM that has more pixels.

5.8 Possibilities and limitations

This project shows that it is possible to control the shape and position of a focused Gaussian beam using a spatial light modulator. The shape and position can be controlled, and the peak can be delayed in time. The peak can be split up into several peaks, and each of these peaks can be controlled individually. In total up to sixteen peaks can be created. It is possible to expand the code to combine more phase profiles, but in practice this will not be necessary. To create an interferometer, a profile is needed at the focus that contains two peaks of which the phase difference is adjustable. This project shows that this is achievable, meaning that it is possible to create an interferometer using a spatial light modulator.
A limitation of the project is the control over the shape of the peak at the focus. The peak at the focus can be stretched out or compressed when a curved profile is applied, but the shape will always remain a Gaussian shape. The amount of peaks at the focus is limited because of the extra peaks. Disturbance of the extra peaks can be avoided up until a certain number of peaks, but there is a limit to how many peaks can be created. With the size of the input intensity used in this project, sixteen peaks can be created. A larger number is not possible since too many extra peaks will appear and they will overlap with the main peaks. Increasing the size of the input intensity will make the peaks smaller which allows the peaks to be brought closer together. However, the number of extra peaks increases as a square of the number of main peaks, so it won't be possible to combine many more peaks. However, in practice there is no need for profiles containing many peaks, so this limitation does not affect the research.

6 Outlook

Future research could include developing a method to control the shape of the peaks at the focus. Ideally, a pulse having a triangular shape would be created, since a triangle contains a straight linear line. It is not possible to create a profile that contains discontinuities such as corners. However, it is possible to create a profile that has a shape closer to a triangular shape.

This project shows that it theoretically possible to create an interferometer. The next step would be to try this out in the laboratory.

Acknowledgements

I would like to thank my supervisor Johan Mauritsson and my co-supervisor Emma Rose Simspson for supervising my project. I would also like to thank Samuel Bengtsson, who has written a code similar to mine and has helped me with problems I encountered while writing the code for this project.

References

[1] Bengtsson S. 'Ultrafast opto-optical control of extreme ultraviolet light pulses'. Lund university. Lund; (2017).

[2] McPherson A, Gibson G, Jara H, Johann U, Luk TS, McIntyre IA, et al. J Opt Soc Am B. (1987);4:595.

[3] Ferray M, L'Huillier A, Li XF, Lompre LA, Mainfray G, Manus C. J Phys B. (1988);21:L31.

[4] Schafer KJ, B Yang LFD, Kulander KC. Phys Rev Lett. (1993);70:1599.

[5] Corkum PB. Phys Rev Lett. (1993);71:1994.

[6] Lewenstein M, Balcou P, Ivanov MY, L'Huillier A, Corkum PB. Phys Rev A. (1994);49:2117.

[7] L'Huillier A. Lecture notes about Atoms in strong Laser Fields;. Published through Canvas for students taking the course Light-matter interaction (FYST21) at Lund University.

[8] Ibrakovic N. 'Control of Coherent Extreme Ultraviolet Light and Light Sources'. Lund university. Lund; (2020).

[9] Bengtsson S, Larsen E, Kroon D. Space–time control of free induction decay in the extreme ultraviolet. (2017);11:252–258. Available from: http://dx.doi.org/10.1038/nphoton.2017.30.

[10] Foot CJ. Atomic Physics. New York, New York, United States of America: Oxford University Press; (2005).

[11] Saleh BEA, Teich MC. Fundamentals of Photonics. 2nd ed. Hoboken, New Jersey, United States of America: John Wiley Sons, Inc.; (2007).

[12] Smith SW. The Scientist and Engineer's Guide to Digital Signal Processing. San Diego, California, United States of America: California Technical Publishing; (1997).

[13] Redmon N. A gentle introduction to the FFT; (2002). Available from: https://www.earlevel.com/main/2002/08/31/a-gentle-introduction-to-the-fft/.

A Figures

A.1 High order Harmonic Generation

A.1.1 Energy diagram of three step model

Figure 22 shows the energy diagram of the three step model.

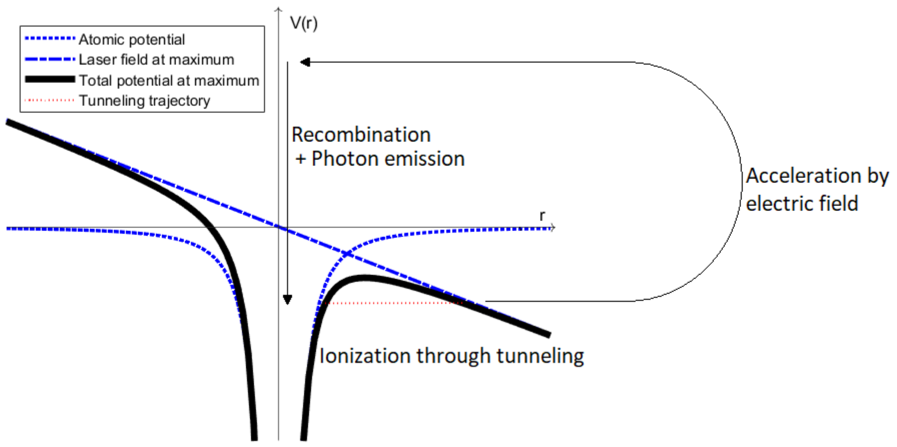
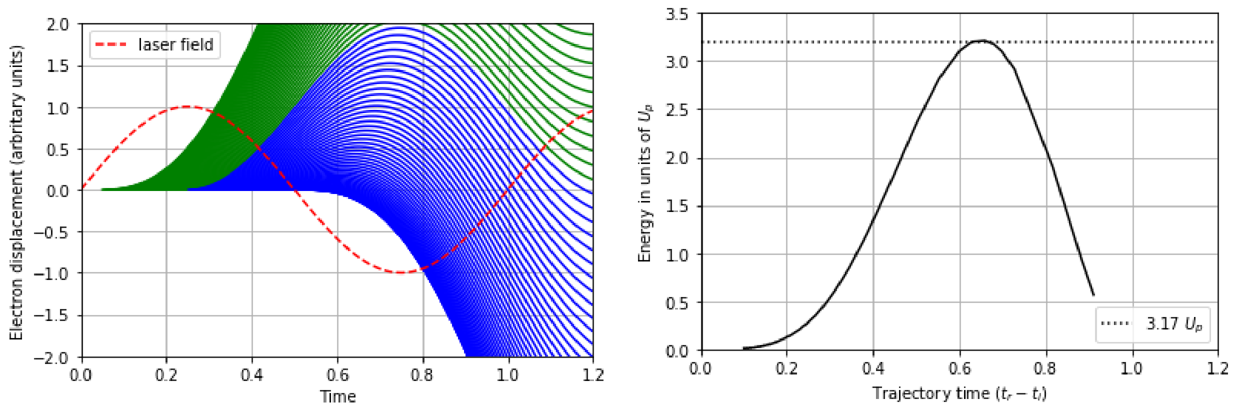


Figure 22: Energy diagram of three step model

A.1.2 Electron trajectories High Order Harmonic Generation

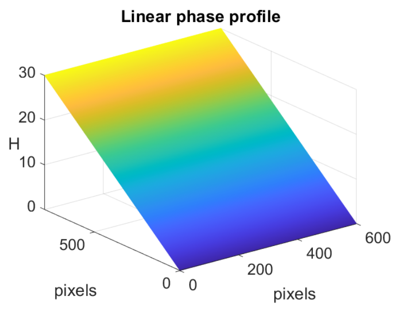


a) Electron trajectories b) Energies as a function of trajectory time

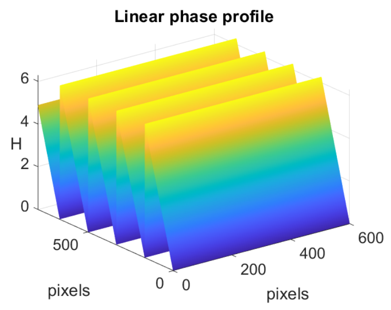
Figure 23: The trajectory (a) and corresponding energy (b) of the electron. The green lines represent the trajectories of the electrons that do not recombine with the atom, and the blue lines represent the the trajectories of the electrons do recombine.

A.2 Method

A.2.1 Subtracting largest multiple of 2π from a phase profile



a) Linear phase



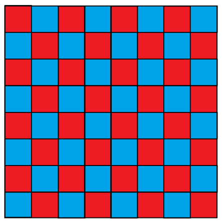
b) Linear Phase with maximum of 2π

Figure 24: Subtracting the biggest multiple of 2π from each value

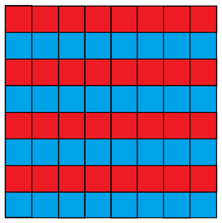
A.2.2 Combination patterns of phase profiles

The figures below show the possible combination patterns of phase profiles.

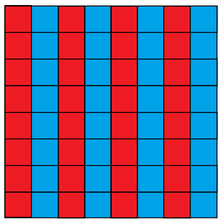
Two phase profiles



a) diagonal



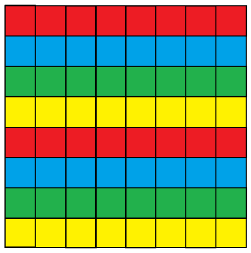
b) horizontal



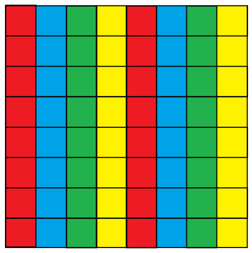
c) vertical

Figure 25: Different patterns through which two phase profiles can be combined. Every box represents a pixel on the SLM. The different colours represent different phase profiles.

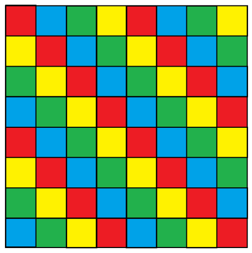
Four Phase profiles



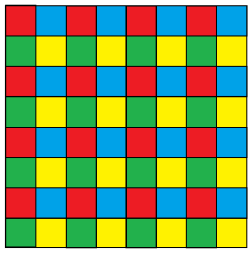
a) Horizontal



b) Vertical



c) Diagonal



d) Square

Figure 26: Different patterns through which four phase profiles can be combined. Every box represents a pixel on the SLM. The different colours represent different phase profiles.

Four, Nine and Sixteen Phase profiles combined through squares

1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4

a) Four linear phases

1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5
7	8	9	7	8	9	7	8
1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5
7	8	9	7	8	9	7	8
1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5

b) Nine linear phases

1	2	3	4	1	2	3	4
5	6	7	8	5	6	7	8
9	10	11	12	9	10	11	12
13	14	15	16	13	14	15	16
1	2	3	4	1	2	3	4
5	6	7	8	5	6	7	8
9	10	11	12	9	10	11	12
13	14	15	16	13	14	15	16

c) Sixteen linear phases

Figure 27: The combination patterns for combining four, nine and sixteen phase profiles through square shapes. Every box represents a pixel on the SLM. The numbers represent different phase profiles.

Six phase profiles combined through different rectangles

1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5
1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5
1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5
1	2	3	1	2	3	1	2
4	5	6	4	5	6	4	5

a) rectangles 2x3

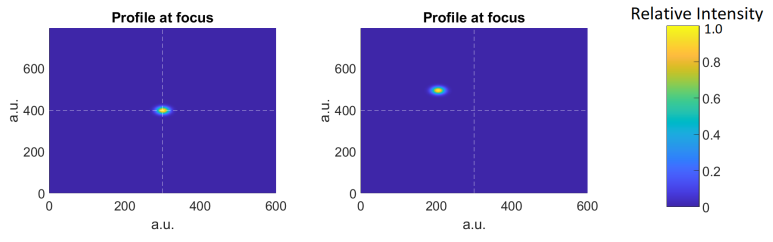
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
5	6	5	6	5	6	5	6
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
5	6	5	6	5	6	5	6
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4

b) rectangles 3x2

Figure 28: The combination patterns for combining six phase profiles through rectangles of dimension 2x3 or 3x2. Every box represents a pixel on the SLM. The numbers represent different phase profiles.

A.3 Results

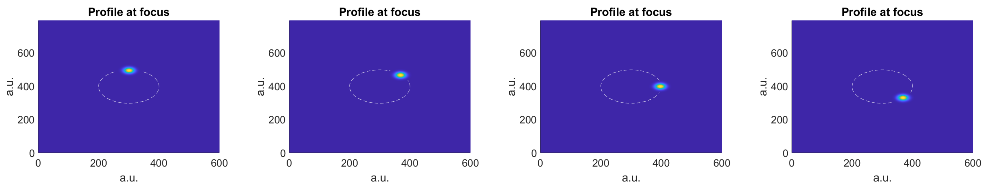
A.3.1 Adding linear phases



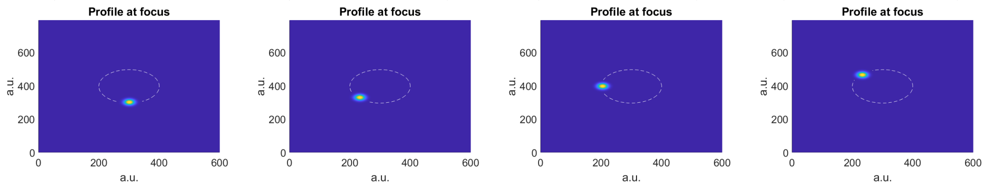
a) No phase   b) Sum of phase pointing up and left

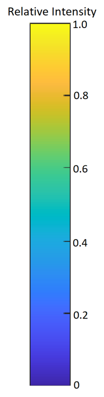
Figure 29: Adding linear phases in different direction

A.3.2 Move peak around circular trajectory



a) theta $= 0$ b) theta $= \pi/4$ c) theta $= \pi/2$ d) theta $= 3\pi/4$



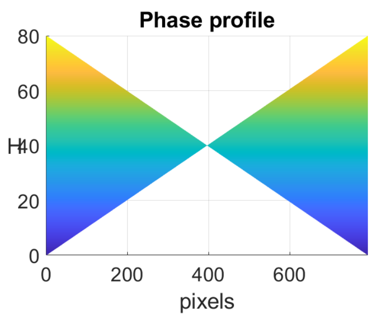


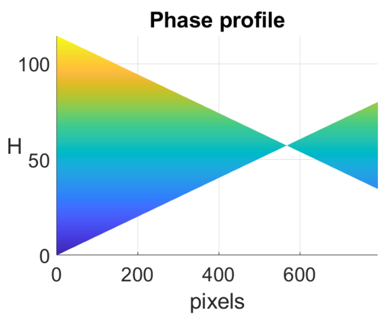
e) theta $= \pi$ f) theta $= 5\pi/4$ g) theta $= 3\pi/2$ h) theta $= 7\pi/4$

Figure 30: Circular trajectory

A.3.3 Time delay

Figure 31 shows how two phase profiles combined look when observed from the side. In figure 31a) there is no time delay, but in figure 31b) a time delay is added.

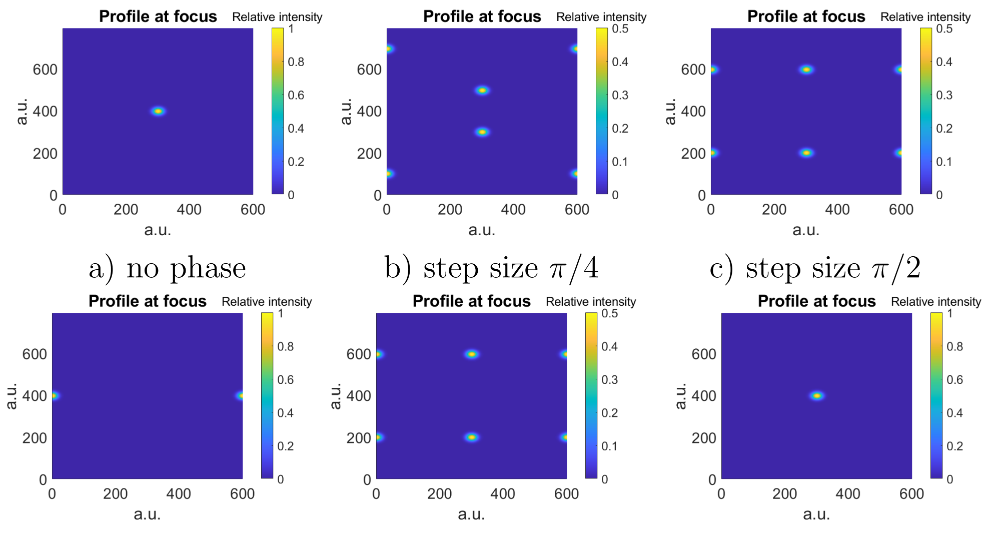




a) two linear phases b) two linear phases, one delayed by half a cycle (π)

Figure 31: The effect of delaying one peak by adding a plane phase

A.3.4 Changing the step size for 2 linear phases



a) no phase b) step size $\pi/4$ c) step size $\pi/2$

d) step size π e) step size $3\pi/2$ f) step size 2π

Figure 32: Profile at the focus for two linear phases with different step sizes

B Simulation code (Matlab)

```
%1. Set Paratmeters
nX = 600;
mY = 792;

aX = -10;
bX = 10;
aY = -10;
bY = 10;

x = linspace(aX, bX, nX);
y = linspace(aY, bY, mY)';

%2. Create Input intensity profile
x0 = 0;           % center
y0 = 0;           % center
sigma = 0.5;          % width
A = 1;            % amplitude

ang = -pi * ((x-x0).^2 + (y-y0).^2)./(2*sigma^2);
input_profile = A * exp(ang);

% f(sqrt(x.^2 + y.^2)<0.5) = 0; %Create a hole
%3. Create Input Phase profile
%Setting the Parameters
```

```matlab
numphases = 16;
shape = 'sq4';
boxsize = 1;
% H = (mY - 1) * (2*pi);
H = 300;
Phase = 0;
theta = 0;

% For shape there are 6 options:
% 1. 'dia' - Diagonal lines
% 2. 'hor' - Horizontal lines
% 3. 'ver' - Vertical lines
% 4. 'sq2' - Squares of size 2 x (numwaves/2)
% 5. 'sq3' - Squares of size 3 x (numwaves/3)
% 6. 'sq4' - Squares of size 4 x (numwaves/4)

% Phase_profile = zeros(mY, nX); %No phase
Phase_profile = combphases(mY, nX, numphases, shape, boxsize, H, Phase,
 ↪  theta); %Linear Phase in several directions

% Phase_profile = Phase_profile - fix(Phase_profile/(2*pi)) * 2*pi;
 ↪  %substracts
% % the largest multiple of 2pi as this is the physical limit of the SLM

%4. Execute Fourier Transform and Inverse
input = input_profile.*exp(1i * Phase_profile);

focus_profile = fft2(input);
focus_profile_phase = angle(focus_profile);
focus_profile_nophase = fft2(input_profile);
inverse_focus_profile = ifft2(focus_profile);

%5. Plot Figures
focus_profile_plot = abs(fliplist2D(focus_profile));
focus_profile_nophase_plot = abs(fliplist2D(focus_profile_nophase));
inverse_focus_profile_plot = abs(inverse_focus_profile);
max_nophase = max(focus_profile_nophase_plot(:));

figure
surf(Phase_profile)
grid on
title('Phase profile')
xlabel('pixels');
ylabel('pixels');
zlabel('H');
```

```matlab
hYLabel = get(gca,'zLabel');
set(hYLabel,'rotation',0,'VerticalAlignment','middle')
shading interp
ylim([0 mY])
xlim([0 nX])
ax = gca;
ax.FontSize = FS;
% view([ 1 0 0 ])
% hc=colorbar;
% title(hc,'H');

figure
mesh(input_profile)
grid on
str = sprintpf('Input intensity profile with \sigma = %d', sigma);
title('str')
title('Input intensity')
xlabel('pixels');
ylabel('pixels');
zlabel('Input intensity');
ylim([0 mY])
xlim([0 nX])
ax = gca;
ax.FontSize = FS;
view([0 0 1])
hc=colorbar;
title(hc,'Intensity');

figure
surf(focus_profile_plot/max_nophase)
shading interp
grid on
title('Profile at focus');
zlabel('Relative Intensity');
ylabel('a.u.');
xlabel('a.u.');
ylim([0 mY])
xlim([0 nX])
ax = gca;
ax.FontSize = FS;
view([0 0 1])
ax = gca;
% cb = colorbar('Location','eastoutside');
% cb.FontSize = 16;
% title(cb,'Relative intensity');
```

```
figure
mesh(focus_profile_nophase_plot)
grid on
title('Focus intensity profile for no phase shift')
view([0 1 0])
ylim([0 mY])
xlim([0 nX])

figure
mesh(inverse_focus_profile_plot)
grid on
xlabel('pixels');
ylabel('pixels');
zlabel('Input Intensity');
ax = gca;
ax.FontSize = FS;
title('Inverse Fourier transform')
ylim([0 mY])
xlim([0 nX])
```

B.1 Output Summary

Table 1 summarizes the matrices that the code plots.

`phase_profile`	Phase profile
`input_profile`	Input Gaussian profile
`focus_profile_plot`	Fourier transformation of the input profile with a phase profile applied to it
`focus_profile_nophase_plot`	Fourier transformation of the input profile without the phase profile applied to it
`inverse_focus_profile_plot`	Inverse Fourier transformation of the focus profile

Table 1: Matrices made by the code

C Animation code (Matlab)

```
numphases = 6;
shape = 'sq3';
boxsize = 1;
H = 300;
Phase = 0;

loops = 20;
F(loops) = struct('cdata',[],'colormap',[]);

```

```
i = 1;

v = VideoWriter('1 peak rotating.avi');
v.FrameRate = 12;
open(v)

for theta=linspace(0, 6*pi, loops)
    Phase_profile = combphases(mY, nX, numphases, shape, boxsize, H, Phase, theta);
    focus_profile = focus_beam(input_profile, Phase_profile);
    surf(focus_profile);
    str = sprintf('angle = %d', theta);
    title(str)
    shading interp
    view([0 0 1])
    drawnow
    A = getframe(gcf);
    writeVideo(v, A);
    F(i) = getframe(gcf);
    i=i+1;
end

close(v);
fig = figure;
movie(fig,F,2,6)
```

D combphases (Matlab)

```
function [ M1 ] = combphases( nX, mY, numphases, shape, boxsize, H, Phase,
 ↪  theta)

M1 = zeros(nX, mY);
M2 = zeros(nX, mY);
M3 = zeros(nX, mY);
M4 = zeros(nX, mY);
M5 = zeros(nX, mY);
M6 = zeros(nX, mY);
M7 = zeros(nX, mY);
M8 = zeros(nX, mY);
M9 = zeros(nX, mY);
M10 = zeros(nX, mY);
M11 = zeros(nX, mY);
M12 = zeros(nX, mY);
M13 = zeros(nX, mY);
M14 = zeros(nX, mY);
```

```
M15 = zeros(nX, mY);
M16 = zeros(nX, mY);

%Define phases that are to be combined here

L = [M1; M2; M3; M4; M5; M6; M7; M8; M9; M10; M11; M12; M13; M14; M15; M16];

if rem(nX, boxsize) ~= 0 || rem(mY, boxsize) ~=0
    error('boxsize has to be multiple of nX and mY');
elseif numphases > 16 || numphases < 0
    error('numphases has to be between 0 and 16')
end

if numphases == 0
    M1 = zeros(nX, mY);
elseif numphases==1

elseif shape == 'dia'
    for i=1:nX/boxsize
    for j=1:mY/boxsize
        for n = 1:numphases
                if (rem(i, numphases) + rem(j, numphases) == n - numphases)
                 ↪  || (rem(i, numphases) + rem(j, numphases) == n +
                 ↪  numphases) || (rem(i, numphases) + rem(j, numphases) ==
                 ↪  n)
                    M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                     ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                     ↪  boxsize*i-(boxsize-1) : (n-1)*nX + boxsize*i,
                     ↪  boxsize*j - (boxsize-1):boxsize*j);
                end
        end
    end
    end

elseif shape == 'hor'
for i=1:nX/boxsize
    for j=1:mY/boxsize
        for n = 1:numphases
                if rem(i, numphases) == n - 1
                    M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                     ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                     ↪  boxsize*i-(boxsize-1) : (n-1)*nX + boxsize*i,
                     ↪  boxsize*j - (boxsize-1):boxsize*j);
                end
        end
    end
end

elseif shape == 'ver'
for i=1:nX/boxsize
    for j=1:mY/boxsize
        for n = 1:numphases
                if rem(j, numphases) == n - 1
                    M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                    ↪   (boxsize-1):boxsize*j) = L((n-1)*nX  +
                    ↪   boxsize*i-(boxsize-1) : (n-1)*nX + boxsize*i,
                    ↪   boxsize*j - (boxsize-1):boxsize*j);
                end
        end
    end
end

elseif shape == 'sq2'
    if rem(numphases, 2) ~= 0
        error('numphases has to be multiple of 2')
    else
        for i=1:nX/boxsize
            for j=1:mY/boxsize
                for n = 1:numphases
                    if n <= numphases/2
                        if rem(j, numphases/2) == n - 1 && rem(i, 2) == 1
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                            ↪   (boxsize-1):boxsize*j) = L((n-1)*nX  +
                            ↪   boxsize*i-(boxsize-1) : (n-1)*nX +
                            ↪   boxsize*i, boxsize*j -
                            ↪   (boxsize-1):boxsize*j);
                        end
                    elseif n > numphases/2
                        if rem(j, numphases/2) == n - numphases/2 - 1&&
                        ↪   rem(i, 2) == 0
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                            ↪   (boxsize-1):boxsize*j) = L((n-1)*nX  +
                            ↪   boxsize*i-(boxsize-1) : (n-1)*nX +
                            ↪   boxsize*i, boxsize*j -
                            ↪   (boxsize-1):boxsize*j);
                        end
                    end
                end
            end
        end
    end

    elseif shape == 'sq3'
    if rem(numphases, 3) ~= 0
        error('numphases has to be multiple of 3')
    else
        for i=1:nX/boxsize
            for j=1:mY/boxsize
                for n = 1:numphases
                    if n <= numphases/3
                        if rem(j, numphases/3) == n - 1 && rem(i, 3) == 1
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                            ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                            ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                            ↪  boxsize*i, boxsize*j -
                            ↪  (boxsize-1):boxsize*j);
                        end
                    elseif n > numphases/3 && n <= 2*numphases/3
                        if rem(j, numphases/3) == n - numphases/3 - 1 &&
                        ↪  rem(i, 3) == 2
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                            ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                            ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                            ↪  boxsize*i, boxsize*j -
                            ↪  (boxsize-1):boxsize*j);
                        end
                    elseif n > 2*numphases/3
                        if rem(j, numphases/3) == n - 2*numphases/3 - 1 &&
                        ↪  rem(i, 3) == 0
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                            ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                            ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                            ↪  boxsize*i, boxsize*j -
                            ↪  (boxsize-1):boxsize*j);
                        end
                    end
                end
            end
        end
    end

    elseif shape == 'sq4'
    if rem(numphases, 4) ~= 0
        error('numphases has to be multiple of 4')
    else
```

```
        for i=1:nX/boxsize
            for j=1:mY/boxsize
                for n = 1:numphases
                    if n <= numphases/4
                        if rem(j, numphases/4) == n - 1 && rem(i, 4) == 1
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                             ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                             ↪  boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j);
                        end
                    elseif n > numphases/4 && n <= 2*numphases/4
                        if rem(j, numphases/4) == n - numphases/4 - 1 &&
                         ↪  rem(i, 4) == 2
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                             ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                             ↪  boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j);
                        end
                    elseif n > 2*numphases/4 && n <= 3*numphases/4
                        if rem(j, numphases/4) == n - 2*numphases/4 - 1 &&
                         ↪  rem(i, 4) == 3
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                             ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                             ↪  boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j);
                        end
                    elseif n > 3*numphases/4
                        if rem(j, numphases/4) == n - 3*numphases/4 - 1 &&
                         ↪  rem(i, 4) == 0
                            M1(boxsize*i-(boxsize-1):boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j) = L((n-1)*nX  +
                             ↪  boxsize*i-(boxsize-1) : (n-1)*nX +
                             ↪  boxsize*i, boxsize*j -
                             ↪  (boxsize-1):boxsize*j);
                        end
                    end
                end
            end
        end
    end
%       Phasevector = Phasevector - fix(Phasevector/(2*pi)) * 2*pi;
end
```

E Different phase combinations

E.1 Default linear phases

```
Right = repmat(linspace(0, H, mY), nX, 1);
Left = repmat(linspace(H, 0, mY), nX, 1);
Up = repmat(linspace(0, H, nX)', 1, mY);
Down = repmat(linspace(H, 0, nX)', 1, mY);
Center = zeros(nX, mY);
Second_Degree_Cylindrical_Paraboloid = 10 * repmat(linspace(-H, H, nX)', 1, mY).^2 / H;
Third_Degree_Cylindrical_Paraboloid = 150 * repmat(linspace(-H, H, nX)', 1, mY).^3 / H^2

Phase_profile = zeros(nX, mY);

ii = 1;
jj = 1;
for x=linspace(-H, H, nX)
    jj = 1;
    for y = linspace(-H, H, mY)
        Phase_profile(ii, jj) = x^2 + y^2;
        jj = jj + 1;
    end
    ii = ii + 1;
end
Spherical_Paraboloid = 20 * Phase_profile / (2*H);
```

E.2 One Linear Phase pointing up

```
M1 = Up;
```

E.3 One Linear Phase pointing up and left

```
M1 = Left + Up;
```

E.4 One Linear Phase on boundary circle

```
M1 = sin(theta) * Right + cos(theta) * Left;
```

E.5 Curved phaseprofiles

E.5.1 Second degree cylindrical paraboloid

```
M1 = Second_Degree_Cylindrical_Paraboloid;
```

E.5.2 Third degree cylindrical paraboloid

```
M1 = Third_Degree_Cylindrical_Paraboloid;
```

E.5.3 spherical paraboloid

```
M1 = Spherical_Paraboloid;
```

E.6 Two Linear Phases

```
M1 = Up;
M2 = Down;
```

E.7 Two Linear Phases, one of them has a phase difference

```
M1 = Left;
M2 = Right + Phase;
```

E.8 Three Linear Phases in a triangle

```
M1 = Up;
M2 = Right;
M3 = Left;
```

E.9 Four Linear Phases in Diamond shape

```
M1 = Up;
M2 = Down;
M3 = Left;
M4 = Right;
```

E.10 Four Linear Phases in a square

```
M1 = Up + Left;
M2 = Down + Left;
M3 = Up + Right;
M4 = Down + Right;
```

E.11 Five Linear Phases, a square with one peak at the center

```
M1 = Up;
M2 = Right;
M3 = Left;
M4 = Down;
M5 = Center;
```

E.12 Six Linear Phases in a 3x2 square

```
M1 = Left;
M2 = Right;
M3 = Left + Up;
M4 = Right + Up;
M5 = Left + Down;
M6 = Right + Down;
```

E.13 Eight linear phases in a circle

```
M1 = sin(0) * Up + cos(0) * Left;
M2 = sin(pi/4)  * Up + cos(pi/4)  * Left;
M3 = sin(pi/2) * Up + cos(pi/2) * Left;
M4 = sin(3*pi/4)  * Up + cos(3*pi/4)  * Left;
M5 = sin(pi) * Up + cos(pi) * Left;
M6 = sin(5*pi/4)  * Up + cos(5*pi/4)  * Left;
M7 = sin(3/2*pi) * Up + cos(3/2*pi) * Left;
M8 = sin(7*pi/4)  * Up + cos(7*pi/4)  * Left;
```

E.14 Nine Linear Phases in a Square

```
M1 = Up + Left;
M2 = Down + Left;
M3 = Up + Right;
M4 = Down + Right;
M5 = Center + Phase;
M6 = Right;
M7 = Left;
M8 = Down;
M9 = Up;
```

E.15 Nine linear phases in a line

```
M1 = 4 * Left;
M2 = 3 * Left;
M3 = 2 * Left;
M4 = Left;
M5 = Center;
M6 = Right;
M7 = 2 * Right;
M8 = 3 * Right;
M9 = 4 * Right;
```

E.16 16 Linear Phases in a Square

```
M1 = Up + Left;
M2 = Down + Left;
M3 = Down + Right;
M4 = Up + Right;
M5 = 3 * Up + Left;
M6 = 3 * Up + Right;
M7 = 3 * Up + 3 * Right;
M8 = 3 * Up + 3 * Left;
M9 = Up + 3 * Left;
M10 = Down + 3 * Left;
M11 = 3 * Down + 3 * Left;
M12 = 3 * Down + Left;
M13 = 3 * Down + Right;
M14 = 3 * Down + 3 * Right;
M15 = Down + 3 * Right;
M16 = Up + 3 * Right;
```

F Fourier shift (Matlab)

```
function [ t ] = fliplist2D( f )
[n, m] = size(f);
t = zeros(n,m);
nhalf = floor(n/2);
mhalf = floor(m/2);

f1 = f(1:nhalf, 1:mhalf);
f2 = f(nhalf+1:n, 1:mhalf);
f3 = f(1:nhalf, mhalf+1:m);
f4 = f(nhalf+1:n, mhalf+1:m);

t(1:nhalf, 1:mhalf) = f4;
t(nhalf+1:n, 1:mhalf) = f3;
t(1:nhalf, mhalf+1:m) = f2;
t(nhalf+1:n, mhalf+1:m) = f1;
end
```

G Focus beam (Matlab)

```
function [focus_profile_plot] = focus_beam(input_profile,Phase_profile)
input = input_profile.* exp(1i * Phase_profile);

focus_profile = fft2(input);
```

```
focus_profile_plot = abs(fliplist2D(focus_profile));

end
```

H DFT (Python)

H.1 One dimension

```
def fourier1D(f):
    N = len(f)
    F = []
    for v in range(N):
        Sum = 0
        for n in range(N):
            c = 2 * np.pi * n * v/N
            Sum = Sum + (f[n] * np.e**(-1j * c))
        F.append(Sum)
    return (F)

def invfourier1D(F):
    n = len(F)
    f = []
    for v in range(n):
        Sum = 0
        for t in range(n):
            c = 2 * np.pi * t * v/n
            Sum = Sum + 1/(n+1) * (F[t] * np.e**(1j * c))
        f.append(Sum)
    return (f)
```

H.2 Two dimensions

```
def fourier2D(f):
    M = len(f[0])
    N = len(f)
    F = array([[complex(0)]*N]*M)
    for k in range(M):
        for l in range(N):
            Sum = 0
            for m in range(M):
                for n in range(N):
                    c = 2 * np.pi * (m * k / M + l * n / N)
                    Sum = Sum + (f[m][n] * np.e**(-1j * c))
```

```python
                F[l][k] = Sum
    return(F)

def invfourier2D(f):
    M = len(f)
    N = len(f[0])
    F = array([[complex(0)]*N]*M)
    for k in range(M):
        for l in range(N):
            Sum = 0
            for m in range(M):
                for n in range(N):
                    c = 2 * np.pi * (m * k / M + l * n / N)
                    Sum = Sum + 1/(M*N) * (f[m][n] * np.e**(1j * c))
            F[l][k] = Sum
    return(F)
```