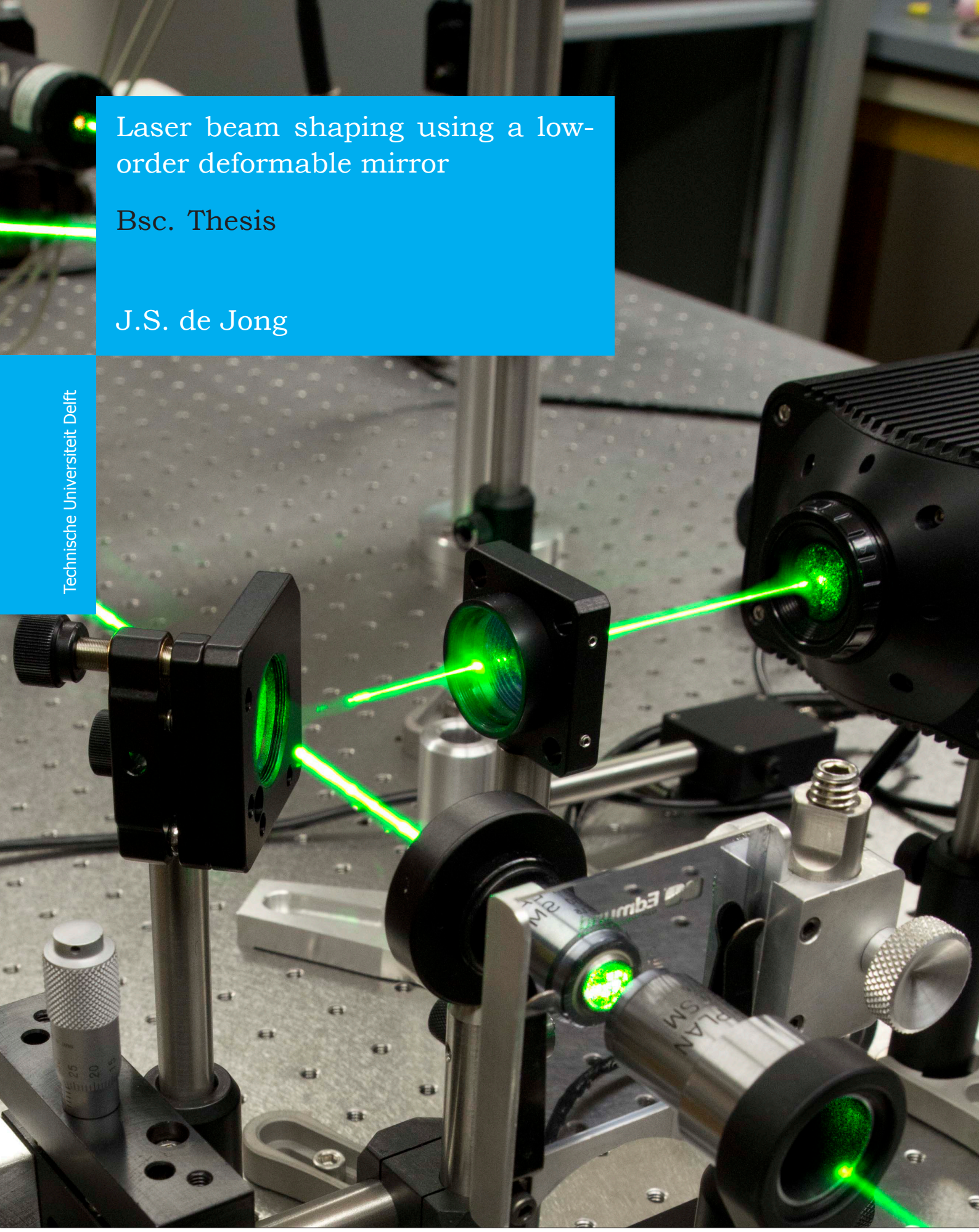
Laser beam shaping using a low-order deformable mirror

Bsc. Thesis

J.S. de Jong

Technische Universiteit Delft



TU Delft Delft University of Technology

Challenge the future

Laser beam shaping

Using a low-order deformable mirror

By

J.S. De Jong

in partial fulfilment of the requirements for the degree of

Bachelor of Science
in Applied Physics

at the Delft University of Technology,
to be defended publicly on Thursday February 25, 2016.

Supervisor:	Prof. dr. ir. M. Verhaegen	
Mentor:	Mr. T. Agbana	DCSC, TU Delft
Thesis committee:	Prof. dr. ir. M. Verhaegen,	DCSC, TU Delft
	Dr. O. Soloviev	Flexible Optical B.V.
	Dr. S.F. Pereira	ImPhys, TNW, TU Delft

This thesis is confidential and cannot be made public until December 31, 2016.

TU Delft

Abstract

Using a low order deformable mirror (DM), laser beam shaping is both simulated and physically applied. Although laser beam shaping has been done before using a DM, it has not been done with a low order DM. In this report, the beam shaping that is done is transforming a laser beam with a Gaussian intensity profile in one plane into a beam with a uniform or top hat intensity profile. Only radially symmetric problems are considered and intensity is treated in a relative sense. This is done for multiple radii of the uniform intensity profile. The beam shaping process is limited by both theoretical and practical limitations, including limitations arisen due to the low order characteristics of the DM.

The shaping process is done using two methods; calculating weights for the DM using its impulse response functions and optimizing for a predefined intensity profile while treating the setup as a black box. A random walker algorithm is used to optimize the problem. The first method does not give favourable results whatsoever due to static aberrations introduced by the system. The second method gives favourable results for a radius of 0.25 mm to 0.5 mm. β, an important parameter that incorporates the scale and many geometrical properties of the beam shaping problem, is a good measure of the viability of shaping problem. It is found that a value of β between 10 and 20 gives favourable results using the optimization process.

Further research can be done on both methods. For the first method, the static aberration can be minimized using the linear properties of the DM. For the second method, a more advanced algorithm can be used which allows for constraints on the input signal of the DM. Furthermore, intensity can be treated in an absolute sense.

Acknowledgements

I would like to thank two persons, without whom this research would not have been possible:

- Temitope Agbana, Msc.

For always helping me out with practical problems, for keeping me in the right direction and for the general supervision.

- Dr. Oleg Soloviev.

For explaining every theoretical question that I could conceive, for giving me a nudge in the right direction whenever I got stuck and for actually supervising me as well.

List of contents

1 Introduction

This is a report on the research on laser beam shaping done for a bachelors thesis.

The goal of the research is to determine if a low-order deformable mirror (DM) can be used for laser beam shaping.

Laser beam shaping is the act of changing the properties of a laser beam's intensity or phase distribution between different planes. For this research, only the intensity profile is considered. It has many applications, including industrial applications as laser machining[1], micro photolithography [2, 3] and medical applications [1].Laser beam shaping has been done for many years [1, 2], but most applications included a passive shaping element. This allows only for single usage. An active element allows for multiple applications and on the fly alteration of the element. This can be done using adaptive optics elements such as a deformable mirror[2].

Using a deformable mirror for laser beam shaping has been done before [2, 3] but mostly with a membrane mirror. In this research, a low order piezoelectric deformable mirror (PDM) is used, with a relatively low amount of degrees of freedom. To my knowledge, this is not done before. A PDM can be coated for usage with a high-energy laser beam, something which is not possible with a membrane surface.

Simulations are done to provide a backbone to measurements and to understand the theory. The simulations are done in MATLAB using a software package called Lightpipes developed by the company OKOtech. With this software light propagation, diffraction and general other elements can be simulated.

The simulations are put to the test by implementing the system on an optical table in the optics lab of the Delft Centre for Systems and Control (DCSC) at TU Delft, Netherlands. The implementation of the DM as the shaping element is done by two different methods. In one approach, the impulse response functions of the DM are used to calculate the optimal shape of the DM.

The other approach treats the optical system as a black box and optimizes for a predefined image using a random walker algorithm.

The theory used in this research and provided in the next chapter relies heavily on the book *Laser beam shaping – Theory and techniques* by *F. Dickey* [1]. It is a go-to source for explanation of laser beam shaping and recommended for any reader interested in a more thorough understanding of laser beam shaping.

2 Theory

This chapter consist of the theory behind laser beam shaping. For completeness, a summary on the theory of geometrical optics is given, since it provides a theoretical foundation for laser beam shaping. Although the diffractive theory on the wavelike nature of light offers a more robust way to derive the theory behind laser beam shaping, it has beam shown [1] that geometrical optics is sufficient. This method is used since it is much easier. It suffices to state that the result using both methods is the same. As such, diffractive optics can be used to validate the result of the derivation using geometrical optics. The interested reader is referred to *Laser beam shaping – Theory and Techniques* by *F. Dickey [1]* for further reading and especially for the treatment using diffractive optics.

2.1. Geometrical Optics

'Geometrical optics is the theory of light in the paraxial approximation'. This is an explanation of geometrical optics that often is given. Although not entirely correct, it does show the important correlation between geometrical optics and limit of small angles or the paraxial approximation, in which geometrical optics is valid. The limit $\lambda/a \to 0$ in which λ is the wavelength of light, and a is some characteristic length of the problem, is actually the region in which geometrical optics is valid. Most macroscopic problems in which the angles from the optical axis are small are valid in the geometrical sense, with examples of exemptions being the resolving power of a system, diffraction or interference[4]. As the frequency and wavelength of light are inversely proportional, geometrical optics can also be derived as a high-frequency limit of the Maxwell equations [1].

The laws of geometrical optics can be derived from Fermat's principle. For two points *a* and *b*, the path that light follows between the points must be stationary. Or described in formula:

$$\delta \int_a^b n(s)ds = 0 \qquad (2.1)$$

With $n(s)$ being the refractive index of the propagation medium and ds being an element of the followed path.

Fermat's principle can be stated as such[1]: the path from *a* to *b* is that path which minimizes the travel time of the light. Although not entirely correct, and in some cases[4] simply wrong, the statement is valid in many approximations. It will also be valid in the presented theory of beam shaping[1].

As is clearly visible from equation (2.1), in a homogeneous medium (with constant refractive index) , a path followed by light will always be straight. Therefore, it is very useful to introduce the concept of light rays – hypothetical lines that describe the path of light. They have many useful connections with wave theory of light: they are perpendicular to wave fronts and for homogeneous media they are perpendicular to both the direction of energy flow and the wave vector of light. The density of the rays is a direct measure of the energy density and thus intensity of light [4]. This last property will be very useful in the theory of laser beam shaping.

2.2. Laser beam shaping

In broad terms, laser beam shaping can be described as the act of changing the intensity distribution of a laser beam from one form into another. This can be done in multiple ways, the most direct method is implementing an aperture in the laser beam. For instance, a Gaussian intensity distribution can be changed into a small, relatively uniform distribution by putting a small aperture at the central axis of the Gaussian. However, this method has two major drawbacks: the area of applicability is very narrow, and the method is very inefficient in terms of energy conservation. Much of the initial energy will be lost, which is highly undesirable in many applications. This report will not treat this method.

Another method to accomplish the shaping is to change the phase-distribution of the beam. It uses a phase element to delay the phase in the plane z=0 (by convention) such that at some plane (*z=f*) the intensity distribution takes the desired shape.

A quick and enlighting way to think about this is to realize that changing the phase distribution alters the shape of the wave front. Since (geometrical) light rays are perpendicular to the wave front, they change direction as well. Therefore, they will be mapped onto the plane z=f in a different way than when the phase element would not be used. Thus the intensity distribution in this plane is changed, which is the goal.

To be more precise, the goal is as such: for an initial intensity distribution $I(\vec{s})$ in the plane z=0, and a desired intensity distribution $Q(\vec{a})$ in the plane $z = f$, with $\vec{s}$ and $\vec{a}$ being two-dimensional coordinates describing their respective planes, find the function $\varphi(s)$ such that the phase delay β $\varphi(s)$ applied in the plane z=0 gives the desired result. Here β is a constant that incorporates all of the geometrical properties of the problem, including size of both $I(\vec{s})$ and $Q(\vec{a})$ and f. It will be computed differently for different distributions. However, it is an important measure in the viability of the shaping problem. The higher the value, the better the result. Both coordinates $\vec{s}$ and $\vec{a}$ are normalized. This allows for a treatment of the function phi which is entirely independent of scale. This way, the viability of the problem is independent of function phi; and completely determined by the parameter β. Also, the problem is very easily scaled.

In the treatment of the theory, it is assumed that a lens of focal length f is situated in the plane z=0 as well. It is also assumed that the incoming beam is collimated, meaning that it has constant phase and a wavefront that is perpendicular to the direction of the beam. Therefore, without a phase shaping element, the beam would be focused perfectly (diffraction limited) in the plane z=f. As such this plane will be referred to as the focal plane from now on. The plane z=0 is referred to as the incident or object plane.

It is useful to separate a lens from the phase delay function for a very practical reason: it allows to recreate the same distribution at different planes by simply replacing the lens with another. As the phase delay element can be hard to manufacture this is a simple but elegant way to allow for multiple uses of the same element. However, it should be taken into account that the parameter beta will change as well since it depends on f.

2.3. Derivation of phase function φ using geometrical optics

The theory behind the derivation of the phase function φ is now given without a thorough derivation. Only radially symmetric problems are treated. Then, the coordinates a and s are only the radial coordinates. As such, the problem becomes one dimensional.

To ensure conservation of total energy, the intensity distribution in the focal plane is multiplied by a constant A. This constant is calculated by dividing the total energy of the incoming beam by the total energy of the desired beam. In formula form:

$$AaQ(a)\frac{da}{d\xi} = \xi I(\xi) \tag{2.2}$$

The function $a(\xi)$ is needed in the next and last step to determine the function $\varphi(\xi)$. To determine this function, the fact that the lights rays' path must be stationary is used. It is sufficient to assume that the total travel time of the light, derived with respect to the coordinate ξ, is zero.

The total travel time is given by the time it takes the light to travel from its origin to the object plane t_L, plus the time the light spends in the phase delay object and the object plane lens $t_{object\ plane}$, plus the time it takes the light to reach the focal plane t_f.

As such, the following equation must hold:

$$\frac{\partial t}{\partial \xi} = \frac{\partial t_L(\xi)}{\partial \xi} + \frac{\partial t_{object\ plane}(\xi)}{\partial \xi} + \frac{\partial t_f(\xi, a)}{\partial \xi} = 0 \tag{2.3}$$

$\frac{\partial t_L(\xi)}{\partial \xi}$ can be regarded as zero if the source is sufficiently far enough from the object plane.
$\frac{\partial t_{object\ plane}(\xi)}{\partial \xi}$ is equal to $-\xi\frac{R^2}{fc} + \frac{RD}{fc}\frac{\partial}{\partial \xi}\varphi(\xi)$, with the first term coming from the lens and the second term coming from the phase delay function.

$\frac{\partial t_f(\xi,a)}{\partial \xi}$ is equal to $\frac{R}{fc}(R\xi - Da)$. Combining the last two expressions, the result is very simple:

$$\frac{d\varphi}{d\xi} = a(\xi) \tag{2.4}$$

This is a very simple first order differential equation that is readily solved by integrating both sides with respect to ξ.

2.4. Radial Gaussian to uniform distribution

Now take $I(s) = e^{-s^2}$ and $Q(s) = 1\ for\ s < 1$ and zero otherwise. Then, $A = \frac{1}{2\pi}$ and therefore using equation (2.2) the following expression for $a(\xi)$ must hold:

$$(2.5) \qquad a(\xi) = \sqrt{2\pi}\sqrt{1 - e^{-\xi^2}}$$

Using equation (2.4) and integrating gives the following expression for $\varphi(\xi)$:

$$(2.6) \qquad \varphi(\xi) = \sqrt{2\pi}\int_0^{\xi} \sqrt{1 - e^{-s^2}}\, ds$$

This non-analytic equation describes the shape the phase delay element must take in order to shape a Gaussian distribution into an uniform distribution. Together with the parameter β it solves the whole problem of beam shaping.

2.4. The parameter β

β is a parameter that incorporates the scaling properties and (most) geometrical properties of the shaping problem . It is a dimensionless scaling factor which will not change value if the ratios between characteristic length in the problem change. What this means is that if a shaping element works for a particular shaping problem, it will also work for scaled versions of the problem, as long as the ratios, and as such the value of β, does not change.

For a radial Gaussian to flat-top conversion, the expression for β is as follows:

$$(2.7) \qquad \beta = 2\pi\frac{R_u R_g}{f\lambda}$$

With R_u the radius of the uniform distribution in the focal plane, R_g the Gaussian radius (at which the intensity is $\frac{1}{e}$ of the maximum) of the distribution in the object plane, λ is the wavelength of light used and f is the focal length.

From this equation it is quite visible that β relies heavily on the wavelength of the light used. This is a limiting factor in the manufacturing process and in the development of a phase shaping element. Also, this is one of the main reasons why the theory behind laser beam shaping only allows for one wavelength at a time.

When dealing with diffraction, β is a good measure to determine whether the problem will be diffraction limited or not. If β is too small, diffractive properties of light will overshadow the shaping process. It is good to realize that if f gets too small, there will be no shaping possible.

The parameter β is important for determining the phase delay that should be introduced by the shaping element, but is had another very important function. Its value is a direct evaluation of the viability of the shaping process. The higher the value of β is, the better the shaping process will be. What is meant is that the obtained intensity distribution will resemble the desired distribution more closely for higher

values of β. As such, smaller values of β will distort the shaping process and eventually, for even smaller values, the shaping process breaks down due to diffraction and other limitations of the theory behind beam shaping. The obtained intensity distribution will not resemble the desired distribution anymore. Therefore, it is important to note the role of β as a quality factor, always pay attention to its value. Based on this fact, in this report the conclusions will be based on values of β, and then on values of the desired beam size. This emphasizes the fact that β is an important quality factor.

2.5. Using a Deformable mirror as the phase delay element

To introduce a phase delay, one can alter the optical path length that the light has to travel. If this path length is different in different points in the object plane, there is a difference in phase between the light waves in the two points. This is, in concept, the desired objective. The optical path length and the phase change of a light wave are related by the very simple formula:

$$\Delta\varphi = k\Delta x_{op}; k = \frac{2\pi}{\lambda} \tag{2.8}$$

With Δx_{op} the optical path length difference, $\Delta\varphi$ the change in phase, and k the wavevector of the light, calculated as 2π over the wavelength λ. The shape that the DM should take is therefore easily computed using this linear correlation.

The DM used in this report is a piezoelectric deformable mirror, or PDM. This means that the shape of the mirror surface is changed by piezoelectric actuators that are attached to the surface at different places. They retract the surface at the point where they are attached. Due to the stiffness of the surface, this shapes the mirror into a certain shape. The shape that the surface takes when only one of the actuators is retracted is called the impulse response function of the respective actuator. The impulse response function depends on which actuator is used, in other words this means that the various actuators have different impact on the surface of the mirror. The impulse response functions are smooth, which can be expressed as them being modal. This is an important characterization of the DM that gives it its articular properties.

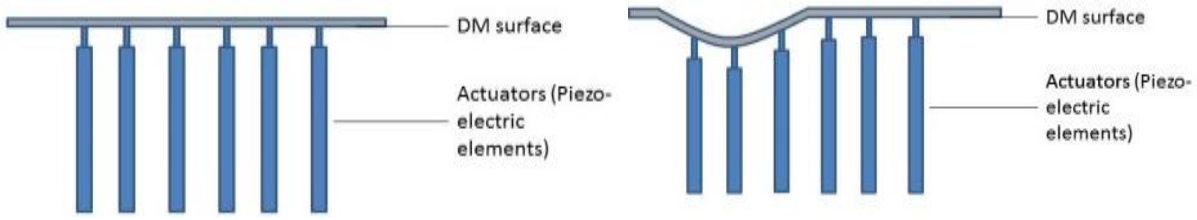


Figure 1 – Left: 2D representation of the DM in its zero state, with no actuators retracted. The surface is entirely flat. Right: One of the actuators is now retracted, therefore the shape of the surface changes according to the impulse response function of the actuator.

The different response functions are mutually linear; the response functions of two actuators combined is the actual result of retracting these two actuators. This linearity applies to any amount of actuators.

When all the actuators are retracted mutually, the entire surface is shifted laterally. All this does is change the optical path length of the entire beam, not just portions of the cross section with respect to other portions. This accounts for an constant phase change over the entire cross-section of the beam. In other words, it defocuses the beam very slightly, since the optical path length travelled is not exactly the focal distance anymore. Although in theory this could be used in other applications, the difference in path length is so small that it has no real effect. This effect is used to allow for both negative and positive values of displacement. To achieve this, all the actuators are retracted halfway initially. This then corresponds to the zero state of the DM. To make this more clearly, usage is made of a control signal instead of expressing the weights in voltages.

To retract the actuators, a voltage is applied to them. The actuators make use of piezo electric components, thus by applying a voltage to the actuator, it changes size and therefor retracts the surface conform its impulse response function. This behavior is linear, half of the maximum voltage corresponds to half of the maximum displacement.

Special note should be taken to the fact that the optical path length difference caused by the shape of the DM will be twice the displacement, since the light travels both ways in the DM.

The shape of the DM, which should equal the phase delay, can be described as a linear combination of all the impulse response functions, since they are mutually linear. Thus, the following equation must hold.

$$\sum_{i=1}^{N} B_i R_i = \frac{1}{2}\beta\varphi(\xi) = \frac{1}{2}\sqrt{2\pi}\int_0^{\xi}\sqrt{1-e^{-s^2}}ds \tag{2.9}$$

With R_i and B_i the response functions and its respective weight, *N* the number of actuators, and all the other variables as described above. This equation is not exact, the DM's response function will never be able to mimic every function, since they are not a complete set of functions. Therefore the impulse response functions will have to be fitted to the function.

3 Simulation

Simulation is done on the setup for three purposes, besides understanding the process better. The first and foremost reason is to determine the theoretical optimal voltages that should be applied to the DM. These are calculated by fitting the impulse response function to a simulation of the phase delay function as calculated in the previous chapter.

The second reason for this simulation is to obtain a theoretical optimal intensity distribution in the focal plane. Since the determined phase delay function of the previous chapter is a non-analytic function, it can only be approximated. Therefore, the intensity distribution in the focal plane will not be perfectly uniform, but will deviate slightly. This distribution can then be used as the optimal distribution in the optimizing process.

There is yet another reason why the simulation is favourable. Using the fitted impulse response function, a preliminary result can be obtained by simulating the actual DM instead of 'just' the normal phase delay function. Some preliminary conclusions can be made to check the usefulness of the particular DM, and whether it's viable to use in this set up.

3.1 Lightpipes

Lightpipes is a software tool that is developed exclusively for the simulation of light. It takes diffraction, coherence and interference into account. It consist of a set of functions written in C for Unix. It was also ported to MATLAB (which version is used in this report). The software was developed by OKOtech (Flexible Optical BV), a small Dutch company that manufactures high-end adaptive optics equipment. The Lightpipes software toolbox as a set of MATLAB functions is available for purchase on the website. For more information, see [5].

Although Lightpipes offers many more functions, only 7 of them were used. For completeness, a small explanation of these commands are given in appendix A. For a more thorough explanation of all the commands used, see the manual provided by OKOtech [6].

3.2 Simulating the phase delay function

For a gaussian to flat-top (uniform) intensity distribution a phase function is needed that has the following function (see par. 2.3) in the radial direction:

$$\varphi(\xi) = \int_0^{\xi} \sqrt{1 - e^{-s^2}}\, ds \tag{3.1}$$

To simulate this, first an array is made that reproduces the normalized variable ξ. This is a radial coordinate which is equal to one at the radius of the Gaussian aperture. At the boundary of the matrix which is filled by the array the value is thus higher than 1; the value is dependent on both the Gaussian aperture and the size of the initiated field. The rectangular matrix with sides *gridsize* is filled entirely.

Since *gridsize* must be even, there is no centre of the matrix. Using a for loop, starting at (gridsize/2,gridsize/2) the value of each entry (gridsize/2+1 ,gridsize/2+1) is calculated by a simple implementation of Pythagoras theorem. Only the bottom-right quadrant is calculated, the other three are copied.

The phase delay itself is stored in a *gridsize* x *gridsize* matrix. First the function is specified, and then, using a for loop for each entry in the bottom-right quadrant of the matrix, the MATLAB function *integrate* is used with as the upper bound ξ the respective entry in the previous matrix. Then again the quadrant is copied into the other three.

To take the finite aperture of the mirror into account, the entered value is set to 0 whenever $2\xi/A_{DM}$ exceeds 1.01, with A_{DM} the diameter of the DM.

3.3. Simulating the field using lightpipes

When starting the field using the *LPBegin* command, three things should be taken into account which greatly affect the rest of the simulation and the end result. The most obvious one is that the wavelength of the light should be specified. Although the laser has a very narrow bandwidth [7], it is indeed finite in size. Lightpipes simulates the light as being truly monochromatic, something which is not physically possible[4]. Therefore, this first step is an unavoidable simplification of the problem. The simulation will be better than what is physically possible in that it filters out some interference which could occur.

Of more practical concern is the gridsize. The gridsize impacts both the memory usage and computation time on one side, and the validity of the simulation result on the other side. For practical considerations later explained, only multiples of 320 are chosen as grid sizes. It should be noted that the demo version of lightpipes software can't use a gridsize greater than 64. This value is grossly too low for this particular application.

The corresponding size of the field should be chosen as small as possible, but should still be able to harbour all of the components in their real physical size. If the size is too big, the physical distance corresponding the distance between two grid points grows as well. It is not necessary to keep the distance between grids within one wavelength. Since the largest parameter in the setup is the aperture of the DM, the diameter of the DM can be used.

To keep the simulation error's to a minimum, as little propagation as possible is done. Therefore, both the Gaussian aperture, the phase delay element and the lens are all simulated in the initial plane. Only thereafter the light is propagated to the focal plane of the lens. Lightpipes offers three different propagation commands; the one used in this simulation is based on the Fresnel approach on diffractive optics. It has less computational burden than the *LPForward* command, of which the calculation time scales with N^4, as opposed to N^2[6]. The third option, *LPForvard*, has the disadvantage that since it uses a spatial Fourier transform it needs broad boundaries, which is undesirable in this case. For more comment, see the manual[6]. *LPFresnel* has another clear advantage; it allows for backwards propagation. This is needed as explained in the next section.

The Gaussian aperture used in the simulation is …mm. The only impact this value has on the final result is scale; only β is dependent on this value. The transmission coefficient is set to 1. Directly after the Gaussian aperture the phase delay is introduced using the *LPMultPhase* command. Then a lens with a focal length of 50 cm is simulated, centred at the optical axis. The field is propagated by 50 cm, the focal length of the lens. Then, using the *LPIntensity* command, the intensity is calculated at the focal plane.

3.4 Fitting the response functions of the deformable mirror to the phase delay functions

The impulse response functions of the DM are provided as 8 bit bitmap files with a 320 x 320 resolution. This is the reason that the gridsize is initially chosen as a multiple of 320; it allows for easy subsampling of the phase delay function for the fitting process. The impulse response functions can be calculated using *MrFit* software, a tool developed by OKOTech. Extra files can be downloaded for the specific DM used in this report. Both the software and the extra files are available from the OKOTech website[8].

The response functions are provided as 8 bit bitmap files, which when imported in MATLAB become matrices with values ranging from 0 to 255. To translate this into the response functions expressed in microns, they are first normalized. Then, they are multiplied with their displacement at maximum input signal. It should be noted that this displacement is different for different actuators, the values can be obtained using the *MrFit* software. Then, a weight in the fit of 1 corresponds to maximum displacement of its respective actuator. This is also a good measure to determine if the fit is possible; if the weight exceeds 1, the displacement needed is too big for the DM.

The impulse response functions cannot be fitted straightforward to the phase delay function. Although for the simulation purposes having the phase delay elemnt and lens in the same plane sufficed, in the real world they are very much in different planes. The phase distribution of the light just before the lens will be different than the one just after the phase delay element, because it is propagated.

To take this into account, an extra phase delay function needs to be applied to the DM. The total phase delay function should be as such:

$$\sum_{i=1}^{N} B_i R_i = \frac{1}{k}\left(\varphi_p + \beta\varphi_s\right) \tag{3.2}$$

With R_i the impulse response functions of the DM in radians, B_i their respective weights in the fit, k the wavevector, φ_p the phase correction for the finite spatial difference between the DM and the lens, and φ_s the phase delay to shape the beam to the desired distribution. Although depicted as an exact equality, this generally will not be the case; although the impulse response functions are a linear set, they are not a complete set and as such will not be able to reconstruct every phase distribution perfectly. Therefore, a least square fit is applied to calculate the optimal weights.

To the determine the phase correcting function φ_p reverse engineering is used. Since the final result is known, this could be used in theory, but more straightforward and less prone to errors is to use the field between the DM and the lens as calculated in the previous section. This field can then be backwards

propagated using the *LPFresnel* command among the desired distance (the real physical distance between the DM and the lens, in this case 270 mm). Then in the plane of the DM the *LPPhase* command can be used to calculate the phase distribution of the field. Since the input to the DM is the collimated beam, with a constant phase distribution, this reverse engineered phase distribution is exactly the one that should be applied to the DM. As such, this distribution can be used to determine the weights B_i and then the voltages applied to the DM.

To determine the optimal weights, the following cost function is minimized:

$$(3.3) \quad \int \left(\sum_{i=1}^{N} B_i R_i - (\varphi_p + \beta \varphi_s) \right)^2 dA$$

With the integral over the entire surface of the aperture. Since the signals in question are discrete, it is actually a sum over all the entries in the matrix.

Taking the derivative, setting to zero and rearranging results in the following set of equations:

$$(3.4) \quad A\vec{b} = \vec{c}; \quad A_{ij} = \int R_i R_j \, dA, \qquad c_i = \int R_i \left(\varphi_p + \beta \varphi_s \right) dA$$

Both the integrals are actually summations of all the elements in the matrix. The multiplications are done in a pointwise manner.

$\vec{b}$ is then a vector containing the weights on all the response functions expressed in the control signal as described in the next chapter. It can be readily applied to the DM.

4 Experimental setup

To validate the simulation results, an experimental setup was designed and built. This chapter deals with both the equipment used in the setup and the process of setting up the optical table. First, the concept is explained. Then a list of the used equipment is presented. Thereafter the setup process is explained in more detail.

Furthermore, the optimization procedure used in this report is presented in this chapter. During the optimisation, the setup is used as a black box-system without known impulse responses. The random walker optimization algorithm was used to find an optimal fit to a pre-calculated image.

4.1. Concept

Light emitted by a fibre coupled laser source(1) is collimated by a collimating lens(2) shown in figure (2). A DM(4), functioning as the phase shaping element, then reflects the incoming beam, and the reflected beam is then focused by the focus lens(5). The intensity distribution in the focal plane of the focus lens(5) is measured using a camera(6).

To reduce the number of optical components which contribute to the aberration of the imaging system, the DM is implemented under an angle (from top view), thus eliminating the need for a beam splitter. As such, two different optical axes exist: one from the laser source to the DM, and one from the DM to the camera. To minimize aberrations introduced by the inclined DM, the angle is kept as small as possible. This eliminates the need for a beam splitter.

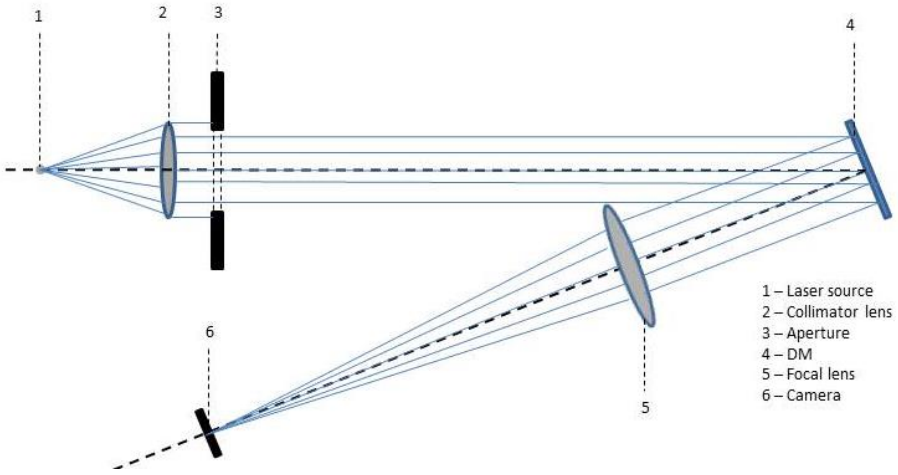


Figure 2 - Experimental setup. The light is emitted form the source(1) in the top left. It then is collimated (2) and the edges are cut off (3). It propagates to the phase shaping element which is the DM(4). It reflects on its surface under a slight angle, and is then focused by the focusing lens(5) in the plane of the camera(6).

The DM is controlled by the computer. Using MATLAB code its surface can be shaped by applying voltages to the DM. The camera can be accessed and operated using MATLAB as well.

4.2. List of optical components used for the optical setup

The following is a list of all the major components used in the setup, sorted as in figure(2).

- 1. Laser source – Thorlabs HLS635 handheld laser source

The Laser source is a handheld source manufactured by Thorlabs. It acts as a point source laser with a centre wavelength of 635 nm; the FWHM is 1-2 nm. It uses a fibre to guide the output. The power output at the centre wavelength is 1 mW.[7]

- 2. Collimating lens – Thorlabs AC254-200-A-ML

To collimate the laser source a lens with focal length 200 mm is used. The diameter is 25.4 mm. It is coated to allow transmission of light in the 400 nm – 700 nm range. [9]

- 4. Deformable mirror – OKOtech 30mm 19ch PDM

The deformable mirror itself is manufactured by OKOtech, also known as Flexible Optical B.V. It is a 19 channel DM with piezoelectric actuators. This amount of channels is relatively low, which makes it a low-order DM. It has a diameter of 30 mm. The impulse response functions can be calculated using the *MrFit* software provided by OKOtech, as mentioned earlier. The high voltage amplifier that delivers high voltages to the DM actuators is controlled using the computer via a Digital Analog Converter (DAC). Due to the smooth surface, and the fact that it has modal impulse response functions, it can be coated with various dielectrics for high energy laserbeams. [10]

- 4(a). DAC – 40-channel 16-bit Ethernet DAC from OKOtech

The DAC is also developed by OKOtech, especially for their adaptive optics systems including the DM. It is controlled by the computer using an Ethernet connection; this is preferable over USB since ethernet allows for a higher output resolution of 16 bit as opposed to 12 bit. Therefore, it has $2^{16} = 65536$ separate levels. The DAC has 40 separate channels, of which 20 are connected to the DM. It outputs a voltage between -12 and 12 Volt, but this can be changed to any bandwidth within these boundaries. For the purpose of controlling the DM, only positive voltages are used, since its response functions lie in this domain. [11]

- 4(b). Amplifier – High-voltage 20 channel amplifier from OKOtech

The amplifier has a gain of 81. It only amplifies positive voltages; when a negative voltage is applied the amplified signal is 0 Volt. The maximum output is 300 Volts, which is also the maximum input the DM allows. It amplifies each of the 20 signals individually. [12]

- 5. Focusing lens – Thorlabs AC508-500-A-ML

The focusing lens is manufactured by Thorlabs as well. It has a focal length of 500 mm. The diameter is 50.8 mm. Just as the collimating lens, it is coated to ensure propagation of light only in the 400 nm – 700 nm bandwidth. [13]

- 6. Camera – Thorlabs DCC1545M CMOS

The camera used is manufactured by IDS and distributed by Thorlabs. It boast a monochrome CMOS sensor with 1280 by 1024 square pixels with a width of 5.2 microns. It is connected to the computer via an USB 2.0 cable. The uEye software developed by iDS can be used to capture the camera's image; it is also accessible in MATLAB using a couple of commands provided by the uEye software via a *.mex* file. [14]

4.3. System design and alignment

The lens source is placed on an optical table. As the source is the end of a fibre, it is placed in a holder which is attached to the collimating lens via a rod. The holder and the lens can slide independently on the rod, so that the distance between them can be adjusted freely. Since the lens acts as a collimating, and it has a focal length of 200 mm, this is also the distance between the laser source and the lens.

To make sure that the laser beam is as best collimated as possible, the source is put on the far left of the optical table. This way, the light can travel as far as possible before hitting the DM. Before introducing the DM, the beam is measured in multiple planes; one very near the collimating lens, one at the end of the optical table and a third plane which even extends beyond the table. The diameter of the beam is measured in all three planes and the distance between the source and the collimating is changed whenever the measured diameters are not the same. Also, the first and second planes can be used to align the optical axis parallel to the optical table. This will make sure that no tilt is needed in the reflective properties of the DM. This is done by measuring the height of the bottom of the beam with respect to the optical table surface. It should be clear that this only should be done after making sure the beam is collimated. The third plane is not suitable for this application since there is no good measure for the height of the beam with respect to the optical table.

After the alignment of the laser beam, the DM is implemented. It is inclined at an angle of approximately 6.5 degrees; a smaller angle is not possible due to the size of the focusing lens. This angle is half the angle between the two optical axes. The distance between the collimating lens and the DM is 91 cm. The DAC and amplifier are placed on a separate desk.

The distance between the focusing lens and the DM is 27 cm. The distance between the focusing lens and the original optical axis (the line between the source and the DM) is approximately 6 cm. A lens with a focal length of 500 mm is used, to introduce less aberration.

The distance between the CMOS sensor and the focusing lens is the focal length of 500 mm. The CMOS is connected to the computer via a USB 2.0 cable. To align the CMOS sensor, first the DM is set to a flat surface. Then, using the uEye software, a live feed from the camera is depicted on the computer screen. The CMOS is then aligned by hand so that the PSF is in the centre. A simple filter is implemented between the CMOS and the focusing lens so that the total intensity is reduced, to prevent saturation

4.4. Control

The DAC that feeds the amplifier is controlled using MATLAB. Various MATLAB commands are presented in a *.mex* file. Below are the MATLAB commands used to interact with the DM. The used commands are listed in appendix A. For a more thorough explanation of the commands, see [11].

Before applying the voltages to the DAC, it is calibrated. This is done using a multi meter that is suitable for both lower voltages (the output by the DAC) and higher voltages (the direct output of the amplifier that is send to the DM). The DAC is calibrated in such a way that its output domain is from 0 to 3.7 volts. This amounts, with the amplifier of gain 81, for a output signal to the DM from 0 to 300 volts.

Because the DAC has a linear output, and the amplifier too, a control signal is very easily implemented. The control signal has a range from -1 to 1. Here -1 equals an output of 0 Volt, and 1 equals an output of 300 Volts, or the maximum displacement. Therefore, a control value of 0 corresponds to the zero state of the DM as described in the theory.

The parameters calculated using the fitting process of the impulse response functions to the phase delay function, as described in the previous chapter, are the relative weights of the functions. They are calculated directly as the control signal.

4.5. Upper limit on β

Although there is no theoretical upper limit on β, there is a practical one in using the DM as the phase shaping element. Since all β does is multiply the complete function by a constant, it only changes the scale of the phase delay function. Therefore, in theory, there is no upper limit on the value of β; it can be arbitrary large, and the phase delay function will just scale with it. However, since the phase delay translates directly into an displacement of the DM by a factor of one over the wave vector, there arises a problem: the DM's impulse response functions only have finite displacement capabilities. Because for the phase delay function of Gaussian to flat-top shaping the highest values are at the outer rim of the mirror, using a β that is too big will result in a mirror that is flattened at the edges. This results in a sub-perfect phase delay function and therefor an undesirable intensity distribution in the focal plane.

One could think that this problem is easily resolved by noting that the phase is periodic over 2π, and higher values than the maximum due to finite displacement can be replaced by a value between 0 and 2π. However, this results in a discontinuity in the surface of the DM, something which is not possible with this DM since its impulse response functions are modal.

The maximum displacement at the edge of the DM is 1.468 microns; this corresponds to a phase delay of 14.5 or 2.3 units of 2π. This limits the value of β. To minimize this problem, the beam size of the incoming beam can be altered. This has two influences on β, one direct and one indirect. The direct influence is that β scales linear with the radius of the incoming beam. However, this effect is not sufficient to allow for usable values of β. The indirect influence is that due to the fact that the incoming beam is smaller, there is no need to use the whole surface of the DM. Therefore, maximum values of the phase delay function will be smaller, which allows for a larger β. The combination of these two effects can be used to allow for a larger β.

To evaluate this, multiple simulations are run, for the following values of the incoming beam size, the used aperture on the DM, and the desired output beam size.

Table 1 - Simulations for various size of the aperture of the DM and output beam radius.

Simulation nr.	1	2	3	4	5	6	7	8	9	10	11
Incoming beam size (mm)	2	2	2	2	2	2	2	2	2	2	2
Aperture of DM (mm)	20	20	20	20	20	20	20	10	4	10	4
Output beam radius (mm)	2	1	0.5	0.25	0.1	0.05	0.01	2	2	0.5	0.5

The value of β will be calculated, and from the simulation results with the corresponding β the region of applicability for the DM can be determined. Also, the cross section of the output intensity distribution will be depicted and its RMS difference with a true uniform intensity distribution will be calculated. The intensity of the uniform distribution will be so that the total energy in the cross section stays the same.

Although desired output beam sizes can be bigger, this is not very useful for this project, since the used camera has a sensor of 6.66 mm by 5.32 mm [14]. Bigger beam sizes would therefore be bigger than the sensitive area.

4.6. Random walker optimization

Another approach to find the optimal shape of the DM, is to treat it as a black-box system. This means that no knowledge of the system and it's impulse response function is assumed to be known. The black-box has one or multiple control parameters, and one output signal. In this case the control parameters are the voltages applied to the DM, and the output signal is a cost function that is based on the image captured by the camera.

The goal of the algorithm is to minimize this cost function. This cost function may have many local minima, thus obtaining a global minimum is a non-trivial task.

The cost function in this case is the second order of the difference between an optimal image, which is calculated using the simulation, and the measured image, which is treated as a function of all of the parameters of the system.

$$(4.1) \quad \Delta I = \left(I_0 - I(\vec{B})\right)^2$$

Here ΔI is the cost function to be minimized, I_0 is the predefined optimal image obtained from the simulation or from some other source, the vector $\vec{B}$ the input signal vector of length 19 (the number of actuators on the DM), and I is the output signal of the system, being the image measured with the camera.

The concept of the random walker algorithm [8] is to make an initial guess of the signal with a corresponding initial value of the cost function. Then a small perturbation is put on the signal, and this is sent as a new signal to the system. The new value of the cost function is then compared to the initial value; if it is higher than the initial value the new signal is discarded. However, if the new value is lower, then the old signal and value of the cost function is replaced by the new ones. The process then repeats.

The perturbation is a random perturbation with a mean value of zero. One of the parameters that define a random walker algorithm is that which defines the domain of the perturbations. If they are too small, the algorithm will end up in some local minimum and will not be able to get out of it. If the perturbations are too big, there will be no fine tuning possible. It is then very unlikely that the algorithm will be able to find the global minimum and the algorithm becomes highly inefficient.

To make the algorithm terminate, the value of the cost function is also compared to a certain threshold value. If the value is lower than this threshold, the problem is regarded as optimized and the algorithm terminates. This threshold value is another parameter that defines the particular random walker algorithm. If it is too big, the algorithm will terminate before finding the optimal values. If it is too small, it may never terminate. However, given enough time, this may not be a problem since the algorithm can be stopped by hand. The best found values can then be easily shown.

4.7. Implementing random walker algorithm into MATLAB

The random walker algorithm is implemented in MATLAB. The camera outputs an image of 1280 by 1024 pixels. To minimize the noise, a selected area of interest of the captured image, based on the size of the desired beam, is used. The camera outputs the image as a matrix in MATLAB. Most entries in the matrix therefor are very close to zero. The maximum intensity corresponds to a value of 255.

First, the centroid of the acquired image is computed. This is done by calculating the first moment of every pixel and dividing it by the total intensity of the whole image. To cancel noise, every value in the matrix below a threshold of 5 is discarded. Then, the relevant portion of the image is taken out. The relevant portion is determined to be a square centred around the centroid with sides of three times the desired beam radius. This is to ensure that there will be no information lost during the calculation of the cost function. Since the pixel size is known as 5.2 microns, the size of the submatrix that is taken out of the image matrix is easily calculated. The amount of pixels is equal to three times the radius, divided by the pixel size. Since the centroid is a non-integer value, the four most nearby values are taken to be the vertices of the one by one square which encapsulates the actual centroid. Then, the four vertices of the submatrix which is to be taken out are easily calculated by subtracting or adding half of the calculated width from both the X and Y coordinate. To ensure that these values are integers, the *floor* and *ceil* commands from MATLAB are used on the lower and higher values of the X and Y coordinates of the vertices, respectively.

After the acquirement of the image, the result of the simulation is used to create I_0. However, this matrix is of different size than the image. Using the *interpol2* MATLAB command it is downscaled to the exact size of the image. The cost function can now be evaluated. First I is subtracted from I_0 in a pointwise manner. Then the result, which is still a matrix, is quadrated in a pointwise manner. Only thereafter the sum of all the entries is taken, to end up with a single valued output.

For a MATLAB implementation of the random walker algorithm, see appendix B. The following is an explanation of that code.

To make sure that the camera has enough time to output the image, there is a small pause implemented every time the camera takes an image. The pause is equal to 3 milliseconds. A *for* loop is used to re-evaluate the cost function for every perturbation. The then optimal signal is stored in the variable *image*, and it's corresponding signal in *voltages*. The corresponding value of the cost function is stored in the variable *I*_delta. Then a small perturbation is put on *voltages* and this is stored in the variable *newvoltages*. For *newvoltages*, an image is taken from the camera and stored in *image*. After cropping as described above, the cost function is evaluated and its value is stored in *I_delta_new*. First *I_delta_new* is compared with *I_delta*. If it is lower, the values in *voltages* and *I_delta* get replaced with the values from *I_delta_new* and *newvoltages*. If this is the case, a reference counter is incremented with one and stored in the variable *reference_counter*. Second, *I_delta_new* is compared with the threshold. If it is indeed below the threshold, the algorithm prints 'Optimized' and the vector *voltages*; it then breaks out of the loop. Third, the values of *I_delta*, *I_delta_new*, the reference counter and the amount of loops passed is printed. This is the end of the loop. The loop will run 10000000 times; this can be regarded as infinite for all intents and purposes.

5 Results

In this chapter the results of the various simulations and measurements are presented. They are divided into five subchapters:

1. Phase delay function
2. Simulation with perfect phase delay function
3. Simulation with fitted impulse response functions
4. Measurements when applying weights directly
5. Measurements with random walker optimization

Very little comment on the results is provided in this chapter, the discussion chapter is used for this purpose.

5.1. Phase delay function

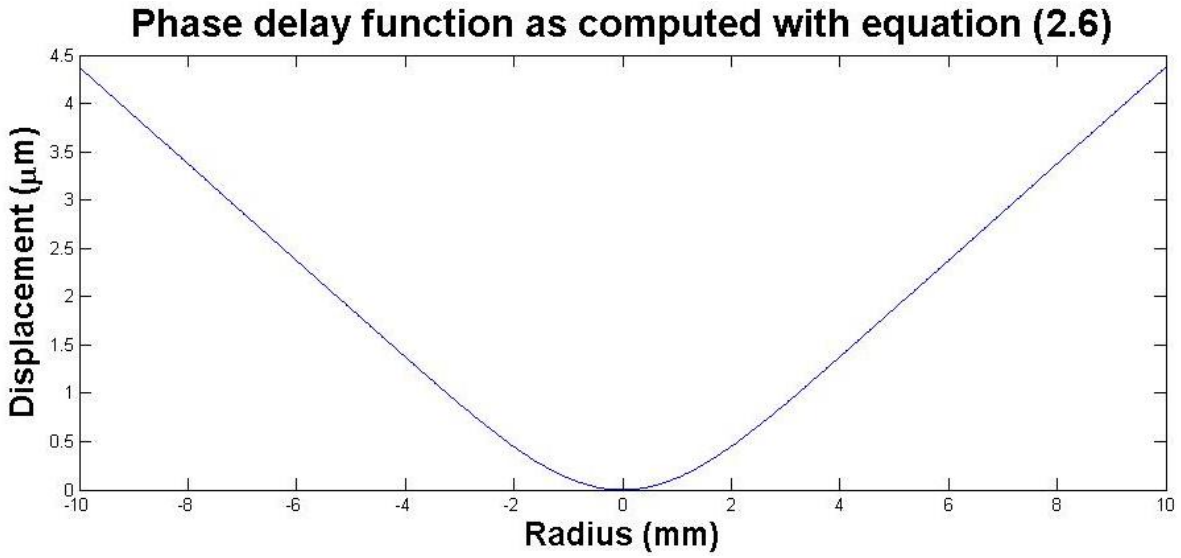


Figure 3 - Phase delay as computed with formula (2.6). Here is shown the phase delay for a value of β of 19.8 with a corresponding desired beam size of 0.5 mm. On the Y axis is micrometres, on the X-axis the radius in metres in the focal plane.

5.2. Simulating with perfect phase delay function at full and smaller DM aperture.

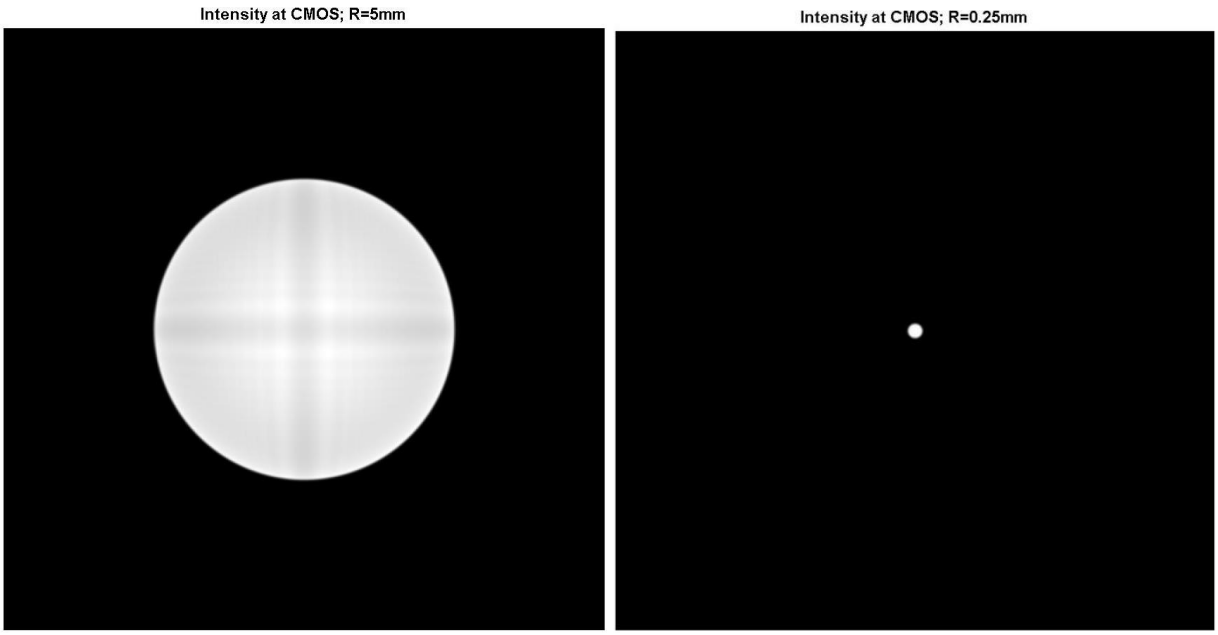


Figure 4 - Intensity at the focal plane for a simulation with a desired beam size of 5 mm(left) and 0.25 mm(right). The total width and height of the image is 20 mm. For this simulation, the phase delay is computed using equation (2.6), as plotted in figure (3)

Table 2- Simulations with different desired beam radiuses. B is directly computed with formula (2.7).

Simulation nr.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Incoming beam size. (mm)	2	2	2	2	2	2	2
Aperture of DM. (mm)	20	20	20	20	20	20	20
Output beam size. (mm)	2	1	0.5	0.25	0.1	0.05	0.01
Results							
Beta	79.16	39.6	19.79	9.89	3.96	1.98	0.40
RMS	0.006	0.069	0.069	0.067	0.058	0.045	0.11

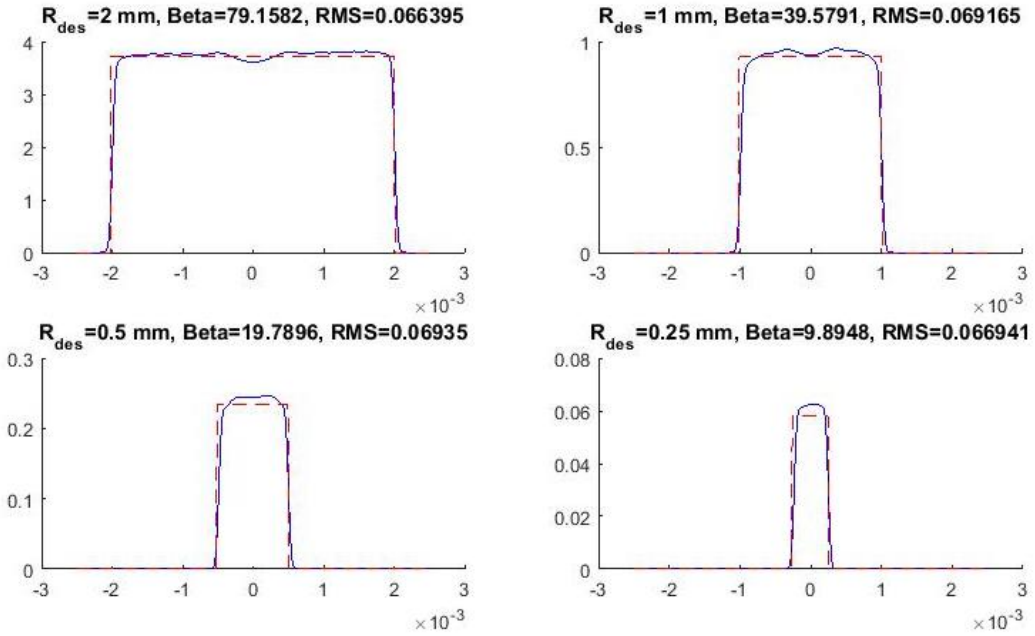


Figure 5 – Blue line: cross section of the intensity at the focal plane vs. the radius in the plane. Red dashed line: optimal top hat shape with the same total intensity as the incoming beam. On the X-axis the radius in the focal plane in meters. The desired beam radius is R_{des}. β is also given. The phase delay used is as depicted in figure (3); the optimal shape as described by equation (2.6). 'RMS' is the RMS value of the difference between the intensity and the optimal red dashed line.

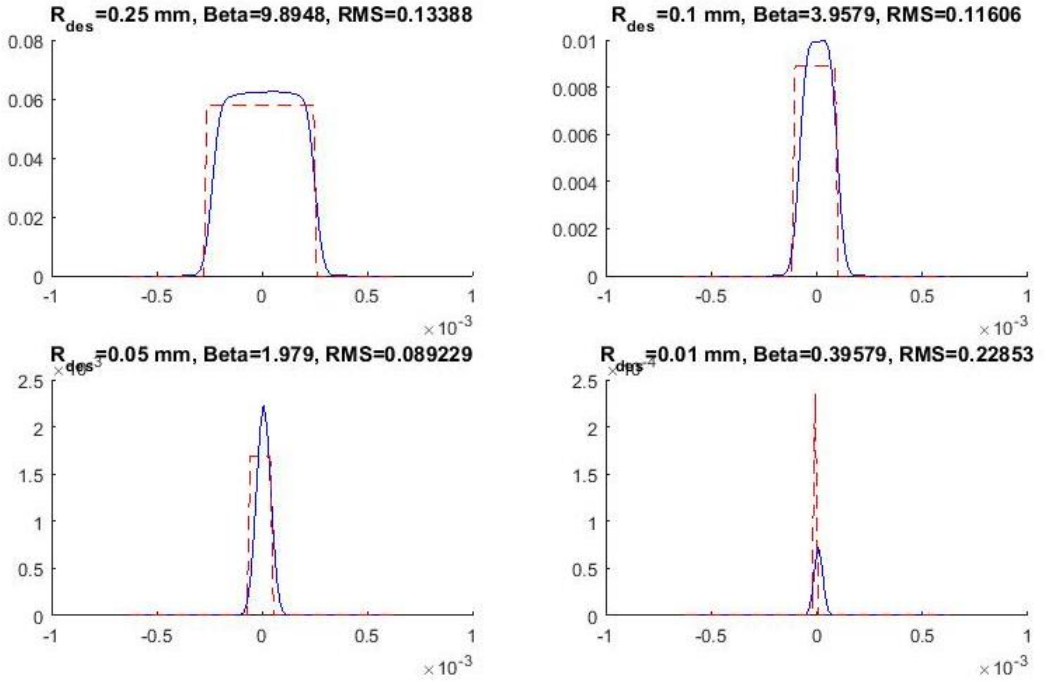


Figure 6 – Blue line: cross section of the intensity at the focal plane vs. the radius in the plane. Red dashed line: optimal top hat shape with the same total intensity as the incoming beam. The desired beam radius is R_{des}. β is also given. The phase delay used is as depicted in figure (3); the optimal result as described by formula (2.6). 'RMS' is the RMS value of the difference between the intensity and the optimal red dashed line. Note the different scale on the x-axis from figure (5).

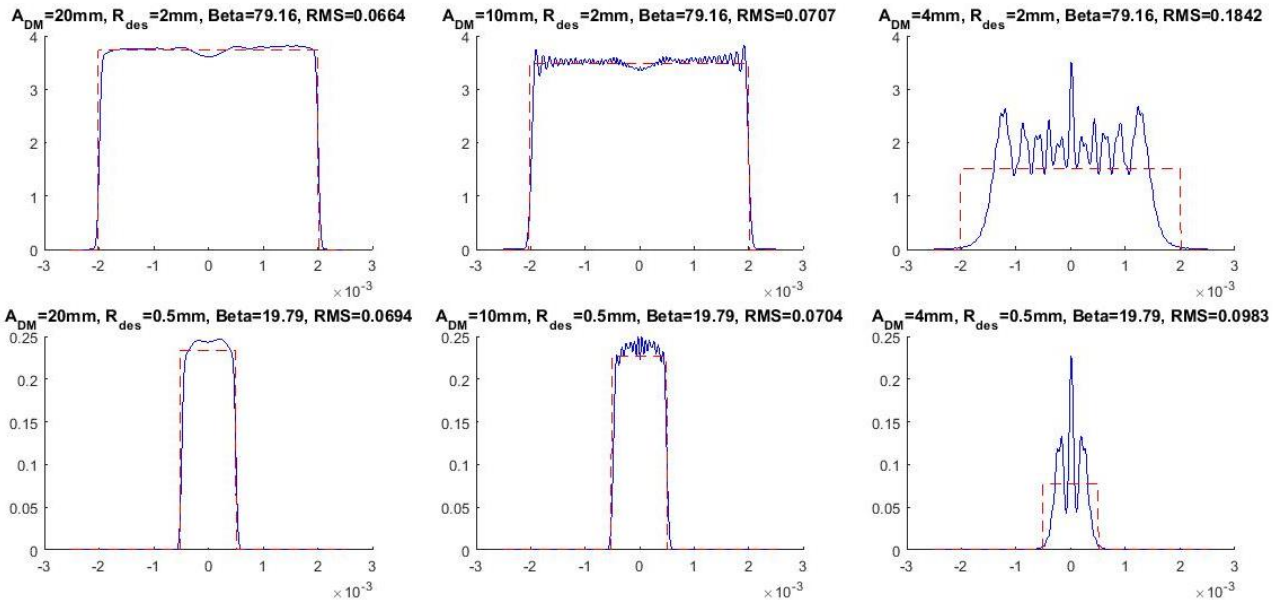


Figure 7 - Simulation using the same phase delay with different apertures of DM, for two sizes of desired beam radius. On the X-axis the radius in the focal plane in meters. A_{DM}, the aperture of the DM, is emulated by putting a circular aperture in front of the phase shaping element. The phase delay is once again as depicted in figure (3).

5.3. Simulating with fitted impulse response functions

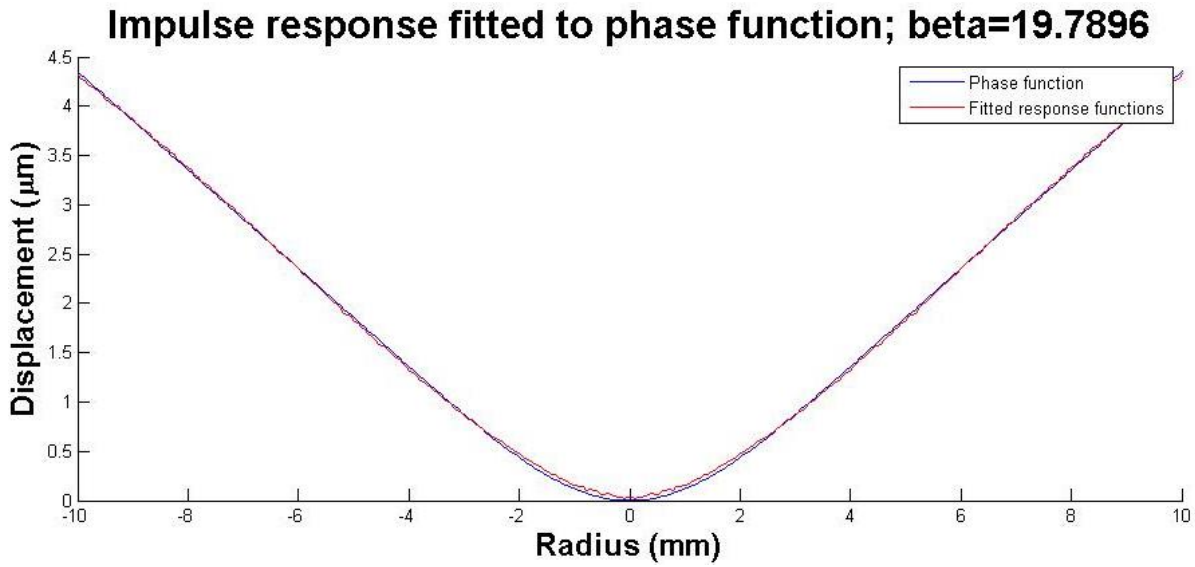


Figure 8 - Both the phase function as depicted in figure (3) (blue) and the response functions fitted to this phase function (red). The functions are depicted in micrometres. Since the fitting process is entirely linear, there is no need to include multiple values of β; in this case that would be literally only a scaling of the y-axis. Shown is the phase delay needed for a desired beam radius of 0.5 mm.

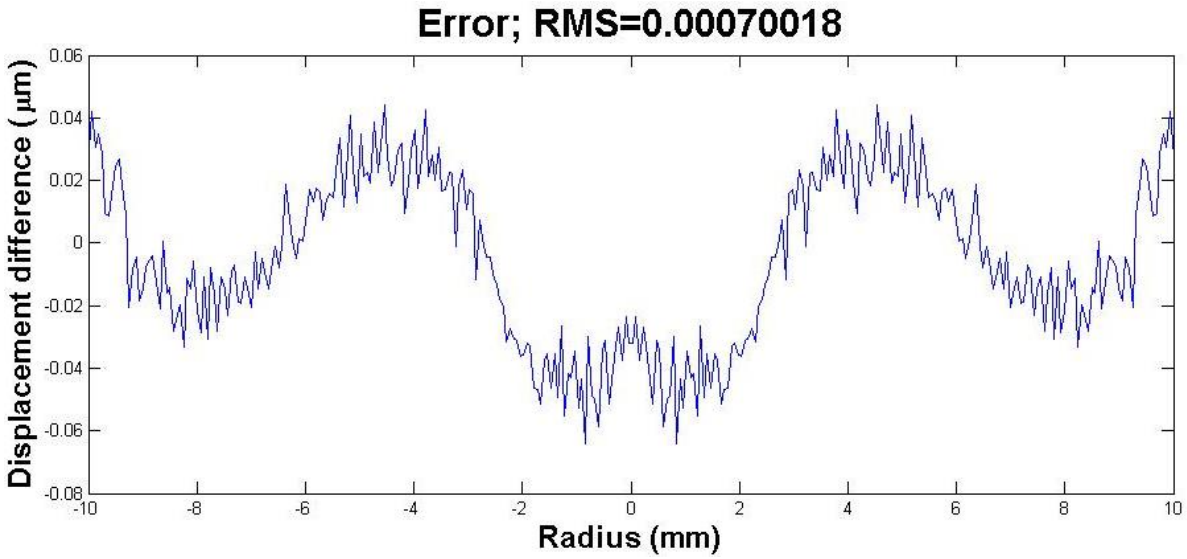


Figure 9 - The difference between the phase function and the impulse response functions fitted to it. Depicted here is the error for a radius of 0.5 mm. Again, multiple values of β's is not needed since the fitting is entirely linear.

Table 3 - Weights of impulse response functions for fitting to phase shape as calculated. Values in excess of maximum of 1 and thus not applicable due to the finite size of DM displacement are marked grey.

Simulation nr.	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Desired beam radius (mm)	2	1	0.5	0.25	0.1	0.05	0.01
β	79.16	39.6	19.79	9.89	3.96	1.98	0.396
Weights (relative)							
1	-7.30	-1.46	-0.73	-0.365	-0.146	-0.0730	-0.0146
2	9.23	1.85	0.92	0.461	0.185	0.0923	0.0185
3	9.46	1.89	0.95	0.473	0.189	0.0946	0.0189
4	9.29	1.86	0.93	0.465	0.186	0.0929	0.0186
5	9.38	1.88	0.94	0.469	0.188	0.0938	0.0188
6	9.38	1.88	0.94	0.469	0.188	0.0938	0.0188
7	9.29	1.86	0.93	0.465	0.186	0.0929	0.0186
8	9.46	1.89	0.95	0.473	0.189	0.0946	0.0189
9	9.23	1.85	0.92	0.461	0.185	0.0923	0.0185
10	9.50	1.90	0.95	0.475	0.190	0.0950	0.0190
11	8.60	1.72	0.86	0.430	0.172	0.0860	0.0172
12	8.36	1.67	0.84	0.418	0.167	0.0836	0.0167
13	8.54	1.70	0.85	0.423	0.171	0.0854	0.0171
14	8.44	1.69	0.84	0.422	0.169	0.0844	0.0169
15	8.44	1.69	0.84	0.422	0.169	0.0844	0.0169
16	8.54	1.70	0.85	0.423	0.171	0.0854	0.0171
17	8.36	1.67	0.84	0.418	0.167	0.0836	0.0167
18	8.60	1.72	0.86	0.430	0.172	0.0860	0.0172
19	8.31	1.66	0.83	0.415	0.166	0.0831	0.0166

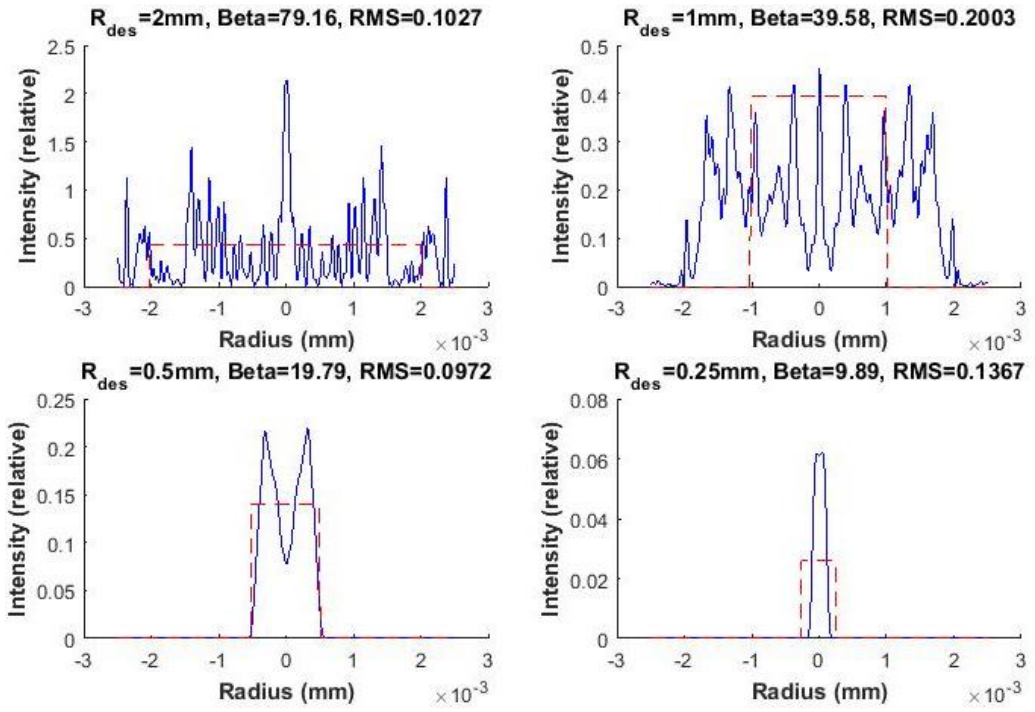


Figure 10 - Cross section of intensity distribution in the focal plane, when propagated using Lightpipes. Now, instead of the phase function from figure (3), the fitted response functions are used with weights from table (3).

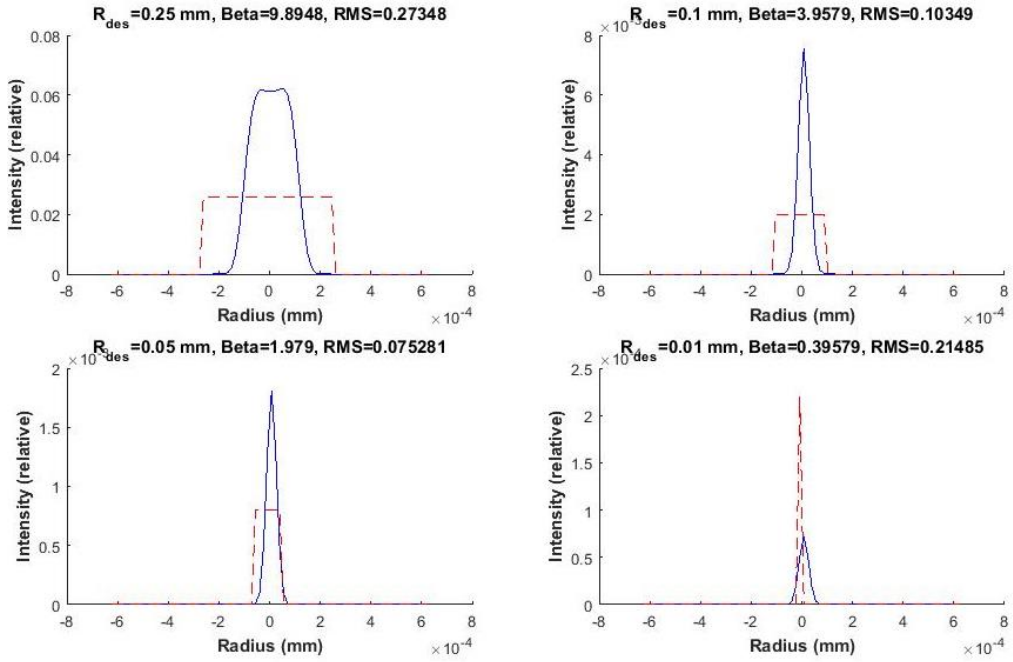


Figure 11 - Cross section of intensity distribution in the focal plane, when propagated using Lightpipes. Now, instead of the phase function from figure (3), the fitted response functions are used with weights from table (3). Note the different scale on the X axis than in figure (10).

5.4. Measurements when applying voltages directly

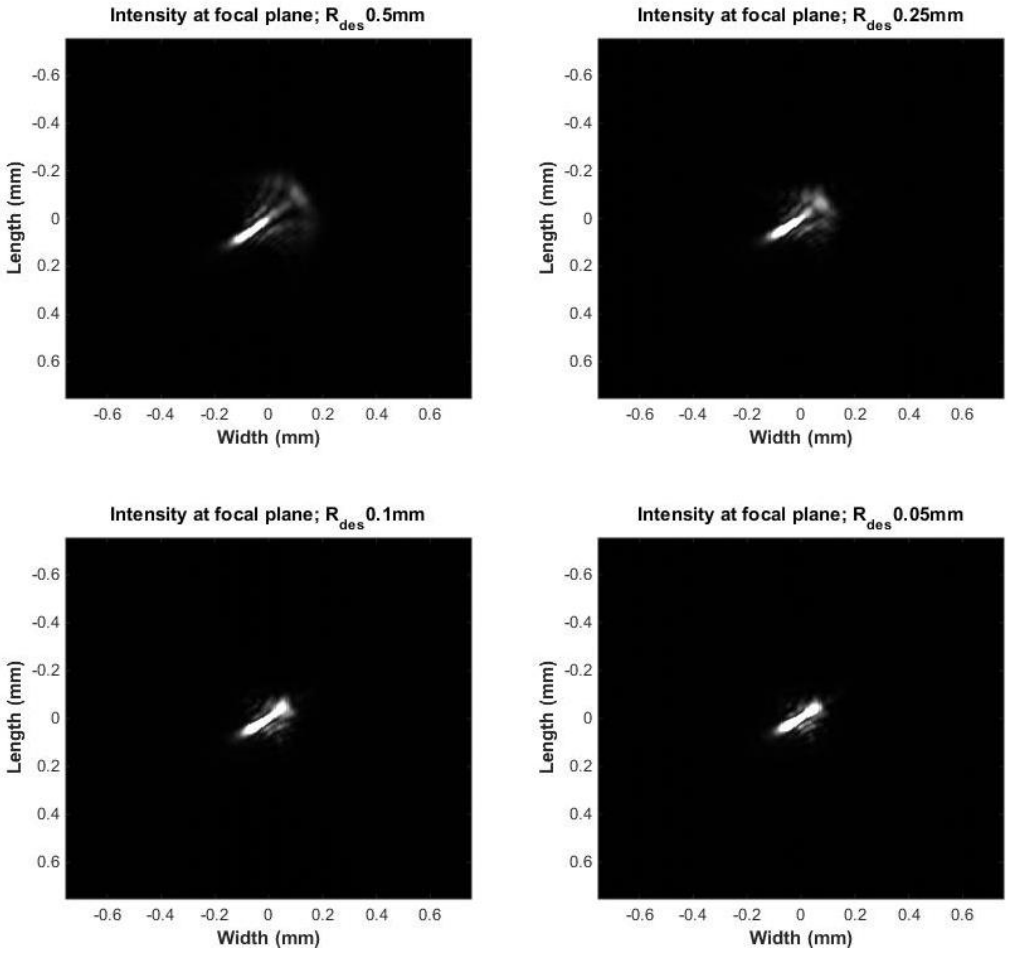


Figure 12 – Intensity at focal plane for Gaussian to flat top beam shaping. The voltages as computed with the weights from table (3) are directly applied to the DM. The image is taken from the camera and then cropped such that the width and length are 3 times the radius of the desired beam shape in the top left picture. The width and length of the crop are kept the same in all four images.

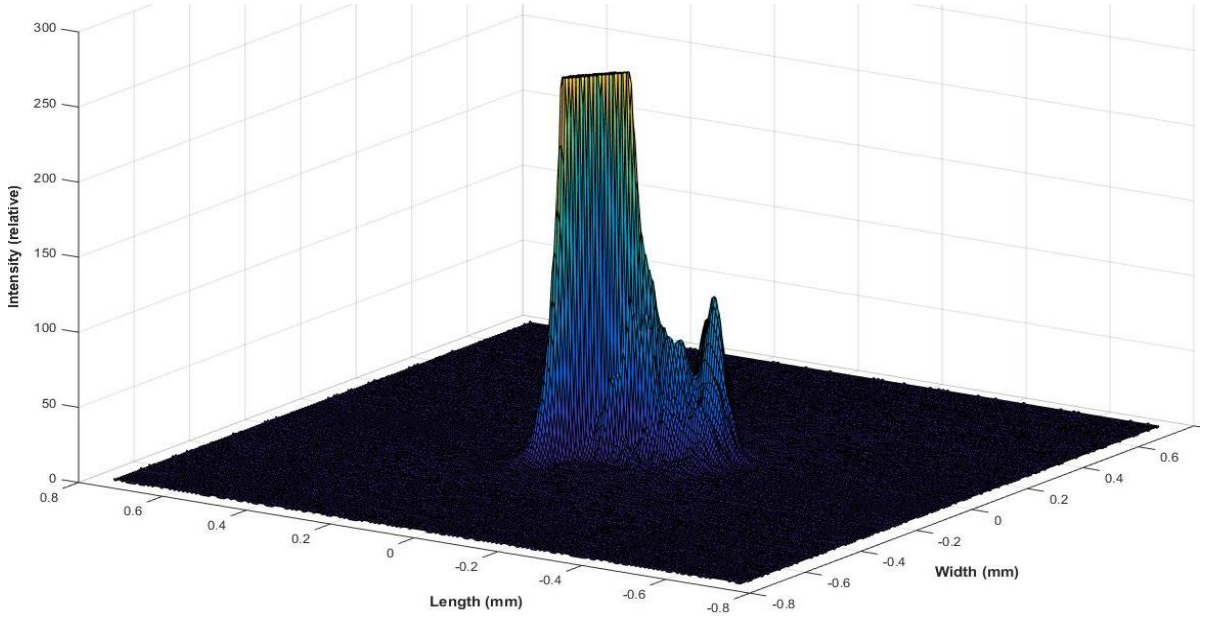


Figure 13 - 3D surfplot of intensity at focal plane for a desired beam radius of 0.5 mm. The image is taken form the camera in the focal plane and then cropped the same as in figure (12). The weights from table (3) are used to control the DM.

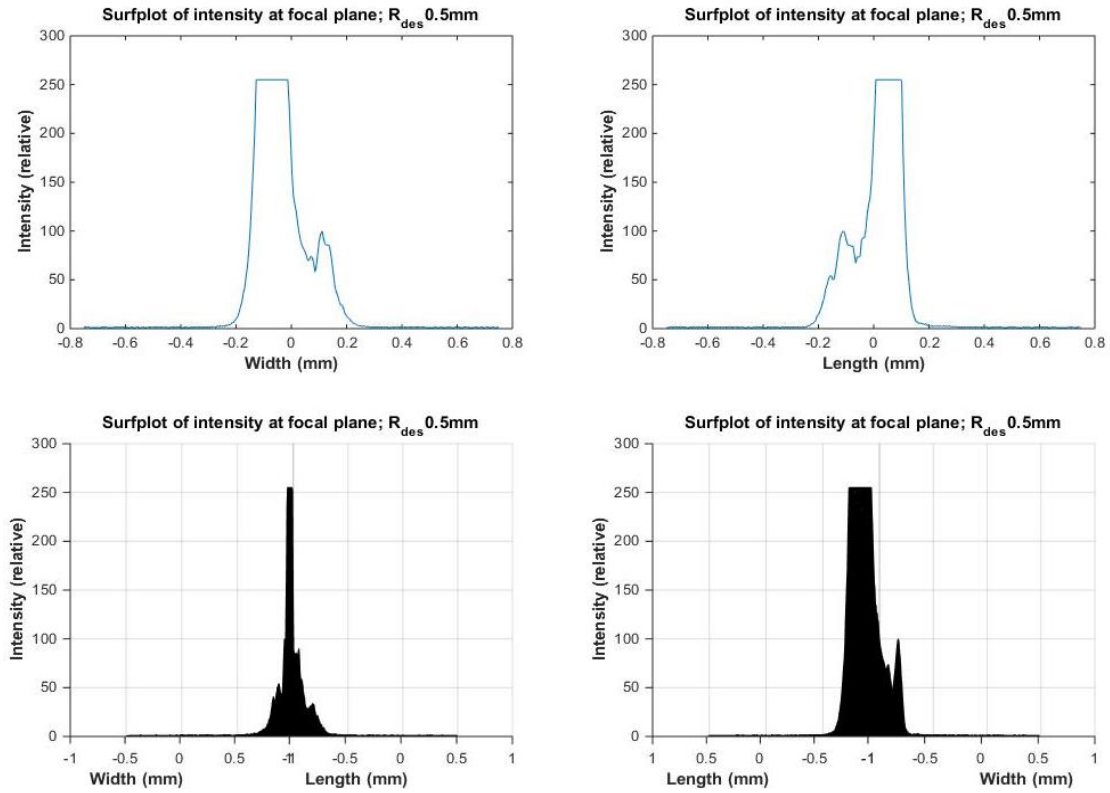


Figure 14 - Side view of intensity at focal plane with the weights from table (3) to control the DM. The desired beam radius is 0.5 mm. Top left: side view from side 'width' side in figure (13). Top right: side view from 'length' side in figure(13). Bottom left: side view from rightmost corner in figure (13). Bottom right: side view from bottom corner in figure(13).

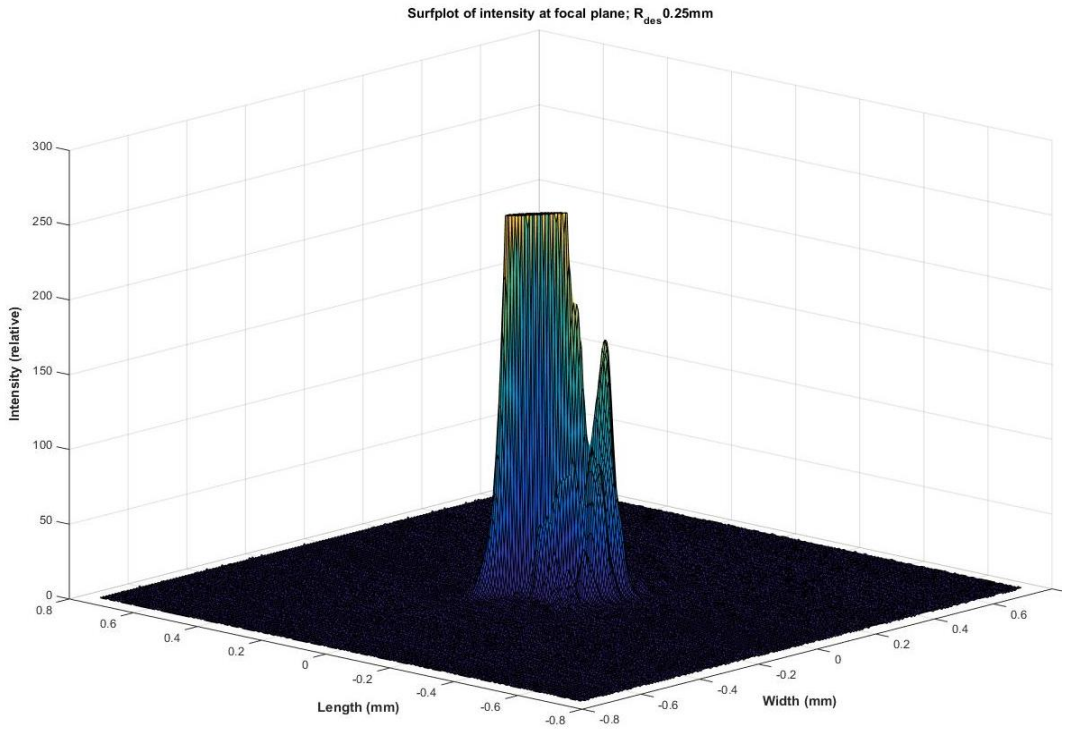


Figure 15 - 3D surfplot of intensity at focal plane for a desired beam radius of 0.25 mm. The image is taken form the camera in the focal plane and then cropped the same as in figure (12). The weights from table (3) are used to control the DM.

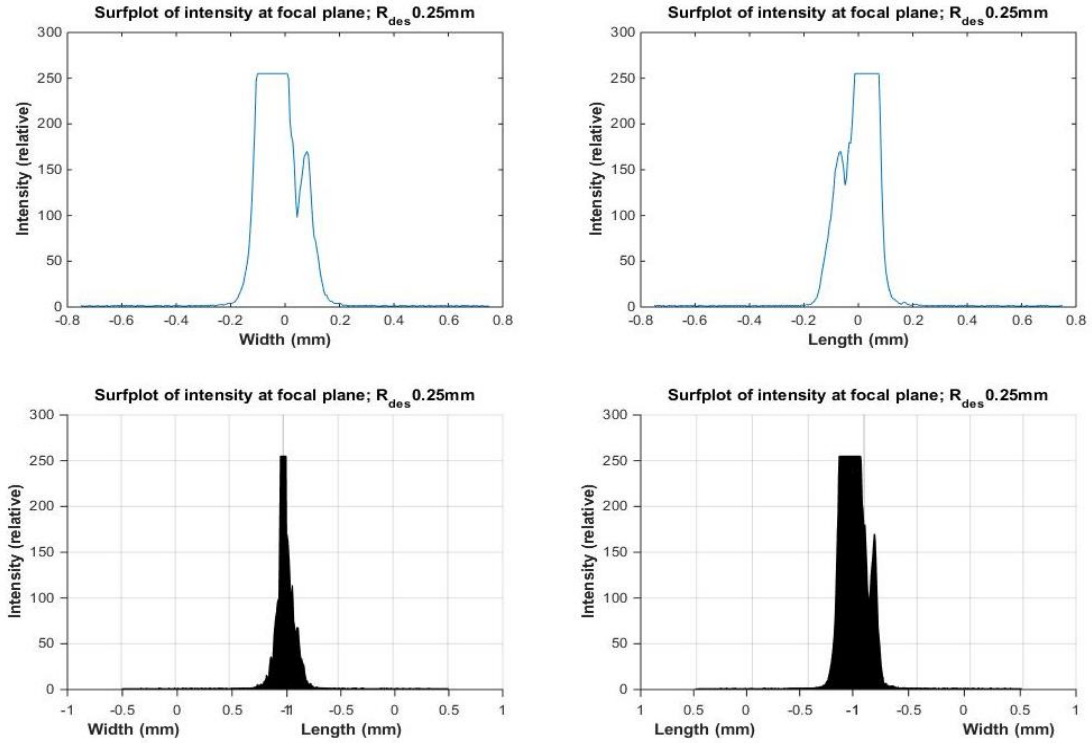


Figure 16 - Side view of intensity at focal plane with the weights from table (3) to control the DM. The desired beam radius is 0.25 mm. Top left: side view from side 'width' side in figure (15). Top right: side view from 'length' side in figure(15). Bottom left: side view from rightmost corner in figure (15). Bottom right: side view from bottom corner in figure(15).

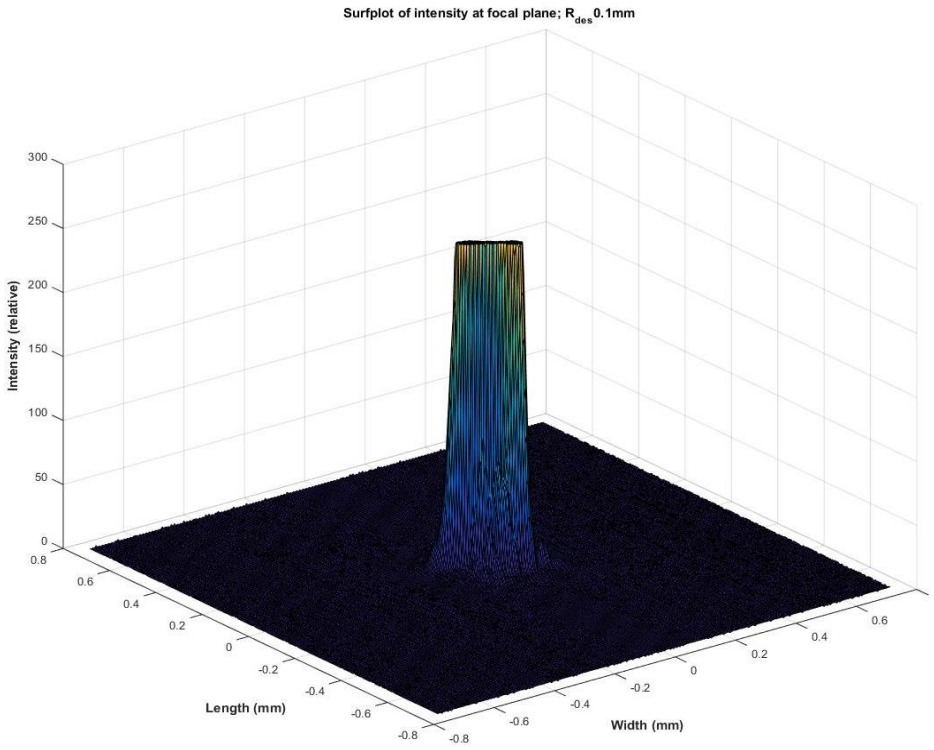


Figure 17 - 3D surfplot of intensity at focal plane for a desired beam radius of 0.1 mm. The image is taken form the camera in the focal plane and then cropped the same as in figure (12). The weights from table (3) are used to control the DM.

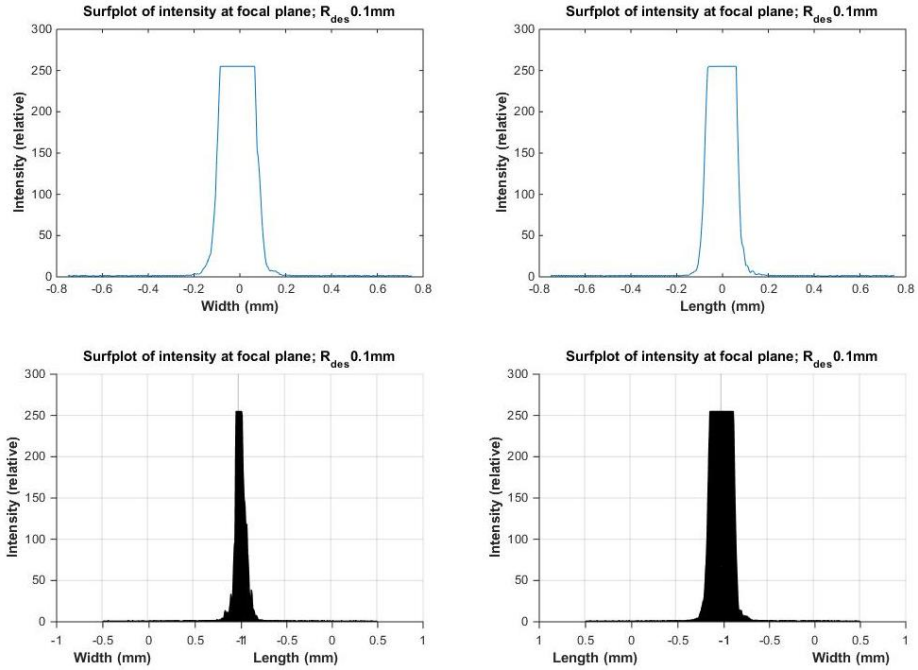


Figure 18 - Side view of intensity at focal plane with the weights from table (3) to control the DM. The desired beam radius is 0.1 mm. Top left: side view from side 'width' side in figure (17). Top right: side view from 'length' side in figure(17). Bottom left: side view from rightmost corner in figure (17). Bottom right: side view from bottom corner in figure(17).

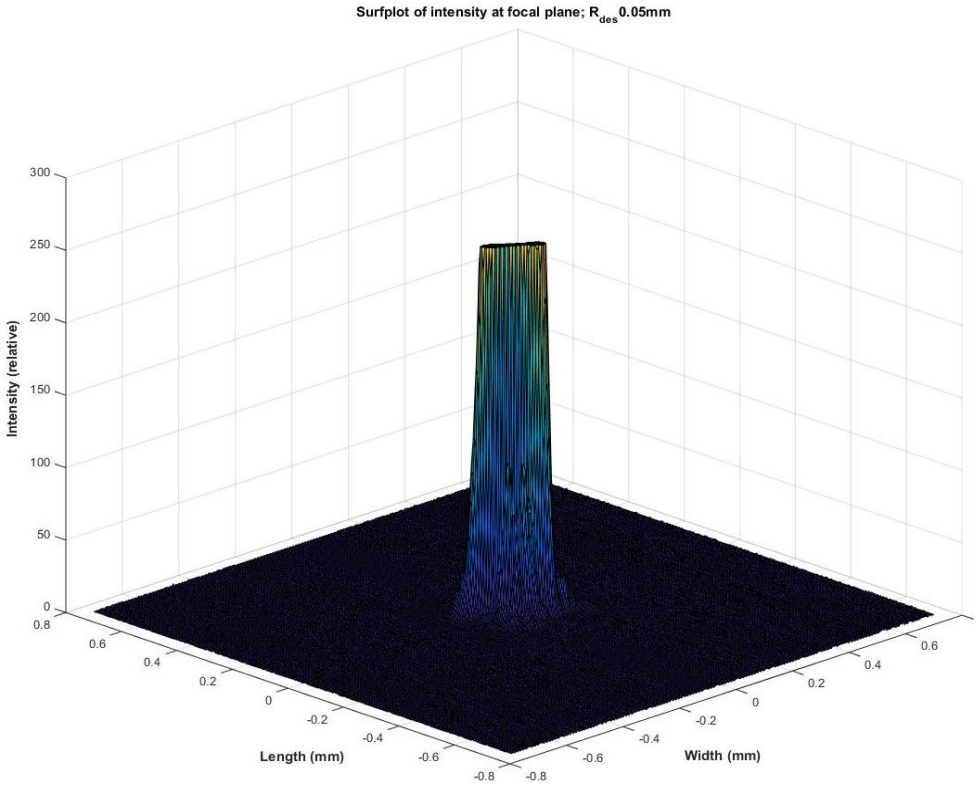


Figure 19 - 3D surfplot of intensity at focal plane for a desired beam radius of 0.05 mm. The image is taken form the camera in the focal plane and then cropped the same as in figure (12). The weights from table (3) are used to control the DM.

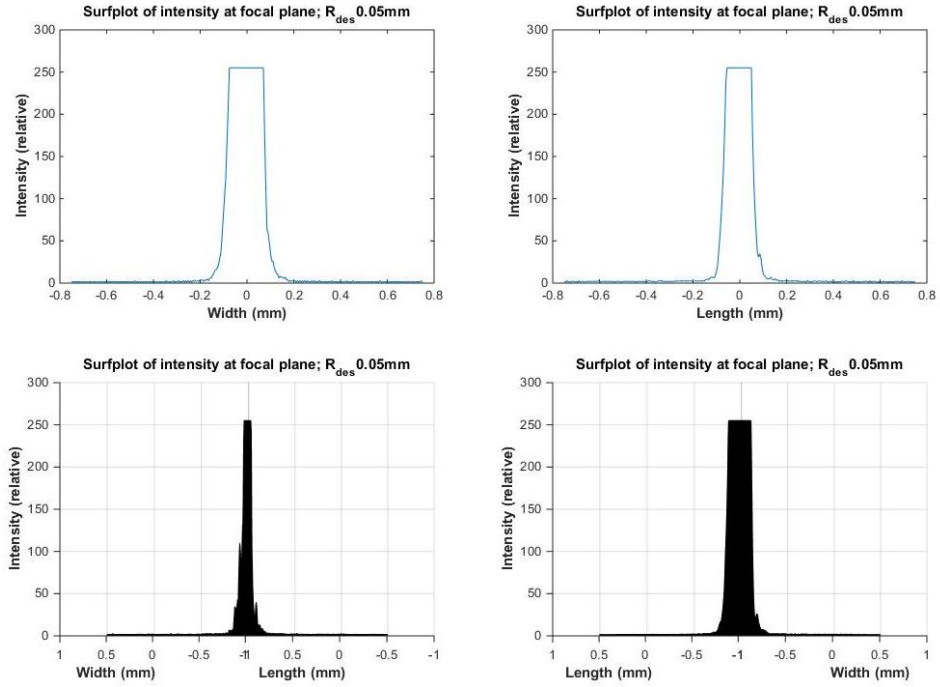


Figure 20 - Side view of intensity at focal plane with the weights from table (3) to control the DM. The desired beam radius is 0.05 mm. Top left: side view from side 'width' side in figure (19). Top right: side view from 'length' side in figure(19). Bottom left: side view from rightmost corner in figure (19). Bottom right: side view from bottom corner in figure(19).

5.5. Random walker measurements

Table 4 – Voltages from Randwom Walker algorithm for four different beam sizes.The corresponding weights of the impulse response functions are then calculated from the voltages.Note the multiple occurences of weights in excess of maximum or minimum of theoretical maximum.

Measurement nr.	**1**		**2**		**3**		**4**	
	Voltages	Weights	Voltages	Weights	Voltages	Weights	Voltages	Weights
Desired beam radius (mm)	0.1		0.5		0.25		0.1	
β (theoretical)	39.58		19.79		9.89		3.96	
Actuators								
1	-1,608	-1,8692	-1,6616	-1,8982	2,4394	0,3186	1,3524	-0,269
2	-1,8702	-2,0109	4,4927	1,42851	3,9521	1,13625	1,5838	-0,1439
3	-0,4085	-1,2208	1,4869	-0,1963	0,6821	-0,6313	2,1085	0,13973
4	-0,792	-1,4281	3,0383	0,64234	1,4706	-0,2051	1,228	-0,3362
5	-0,4178	-1,2258	4,0881	1,20977	3,5673	0,92828	3,1691	0,71305
6	-4,5051	-3,4352	1,9074	0,03104	2,6009	0,40588	3,1429	0,69889
7	3,7078	1,0042	4,5078	1,43662	3,2523	0,75797	1,8905	0,02191
8	0,3045	-0,8354	2,3219	0,25507	2,399	0,29678	2,3289	0,25887
9	2,7666	0,49548	3,7974	1,05263	3,4645	0,87268	1,555	-0,1595
10	-1,8373	-1,9931	1,2668	-0,3152	2,0503	0,10829	2,055	0,11082
11	-0,4666	-1,2522	4,8238	1,60748	4,11	1,22163	1,4798	-0,2001
12	-2,03	-2,0973	0,194	-0,8951	1,6862	-0,0885	0,9995	-0,4597
13	-1,7822	-1,9633	4,3898	1,37286	4,2654	1,30561	0,6408	-0,6536
14	-0,1783	-1,0964	-0,6385	-1,3451	2,5137	0,35873	1,9778	0,0691
15	3,1129	0,68264	3,9454	1,13262	2,6996	0,45923	1,3672	-0,2609
16	-0,2245	-1,1213	-0,1196	-1,0646	1,4676	-0,2067	2,352	0,27137
17	5,2614	1,84401	1,8007	-0,0266	2,0287	0,09661	1,2417	-0,3288
18	-0,1489	-1,0805	2,211	0,19512	0,7117	-0,6153	1,8988	0,02636
19	-1,0415	-1,563	5,0191	1,71301	2,223	0,2016	0,581	-0,6859

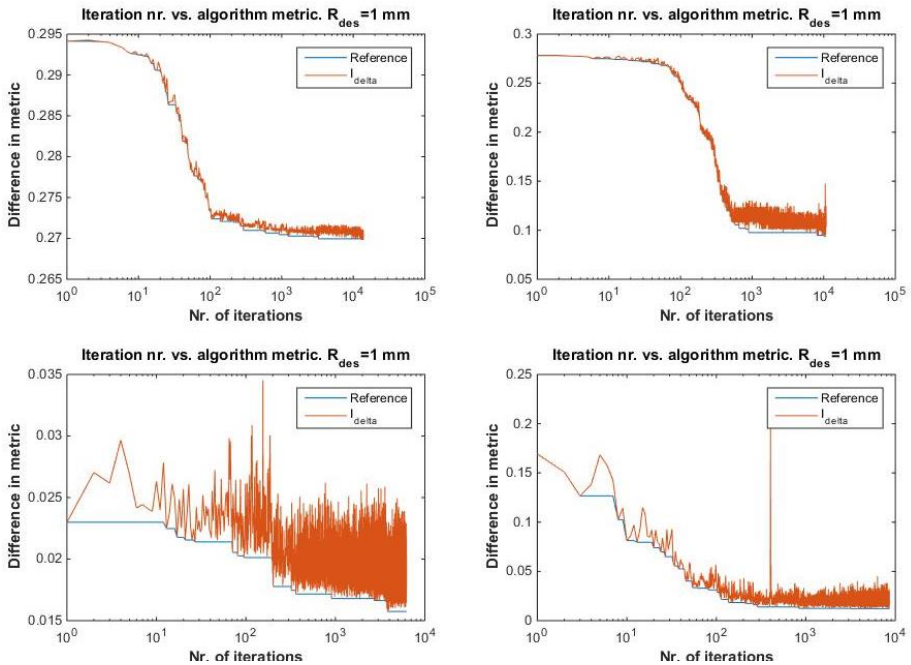


Figure 21 - Plot of the error metric used in the Random Walker algorithm vs. the iteration number.

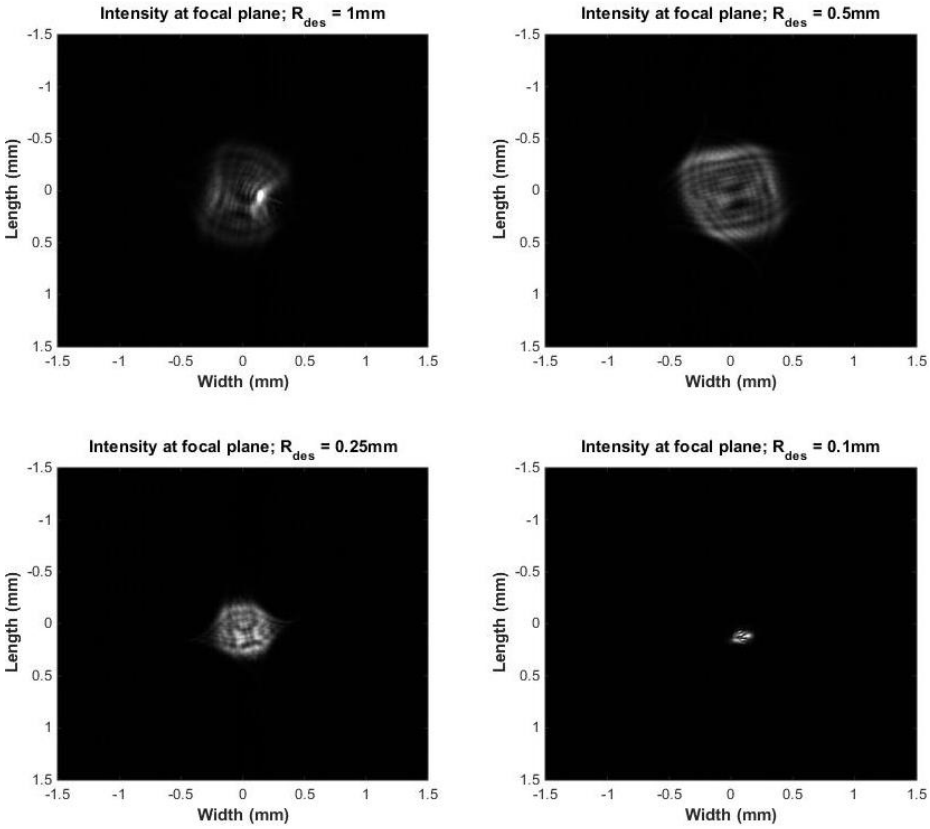


Figure 22 - Intensity at focal plane for Gaussian to flat top beam shaping. The voltages applied to the DM are those obtained with the random walker algorithm from table (4). The image is taken from the camera and then cropped such that the width and length are 3 times the radius of the desired beam shape in the top left picture. The width and length of the crop are kept the same in all four images.

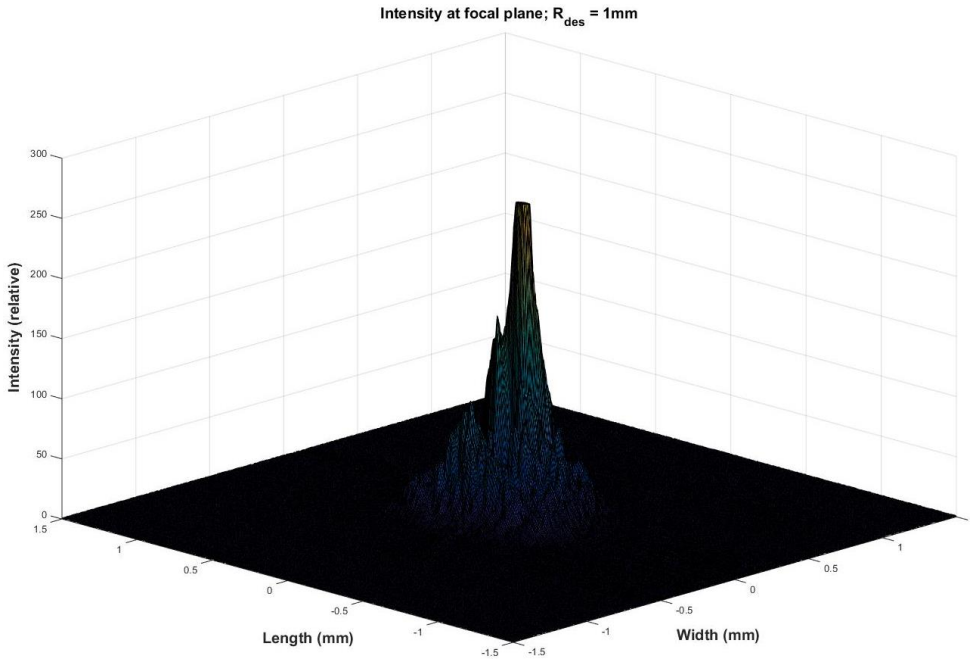


Figure 23 - Intensity distribution in the focal plane for a desired beam radius of 1 mm. The DM is controlled by the random walker algorithm, the voltages applied are given in table(4). The image is taken form the camera in the focal plane and then cropped the same as in figure (22).

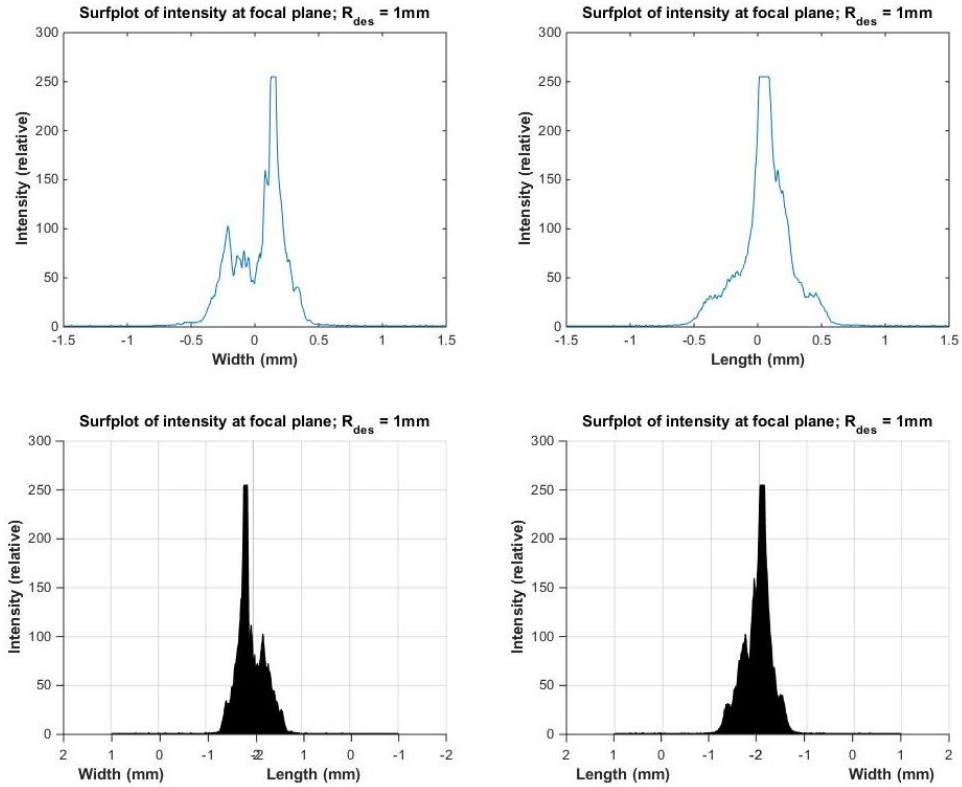


Figure 24 - Side view of intensity at focal plane with the voltages from table (4) to control the DM. The desired beam radius is 1 mm. Top left: side view from side 'width' side in figure (23). Top right: side view from ‘length’ side in figure(23). Bottom left: side view from rightmost corner in figure (23). Bottom right: side view from bottom corner in figure(23).

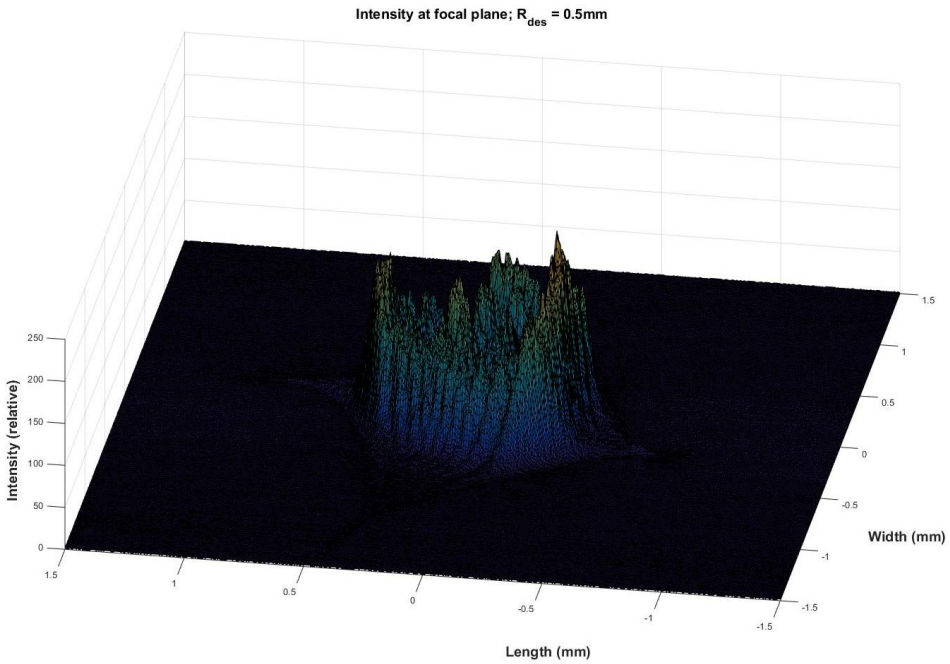


Figure 25 - Intensity distribution in the focal plane for a desired beam radius of 0.5 mm. The DM is controlled by the random walker algorithm, the voltages applied are given in table(4). The image is taken form the camera in the focal plane and then cropped the same as in figure (22).

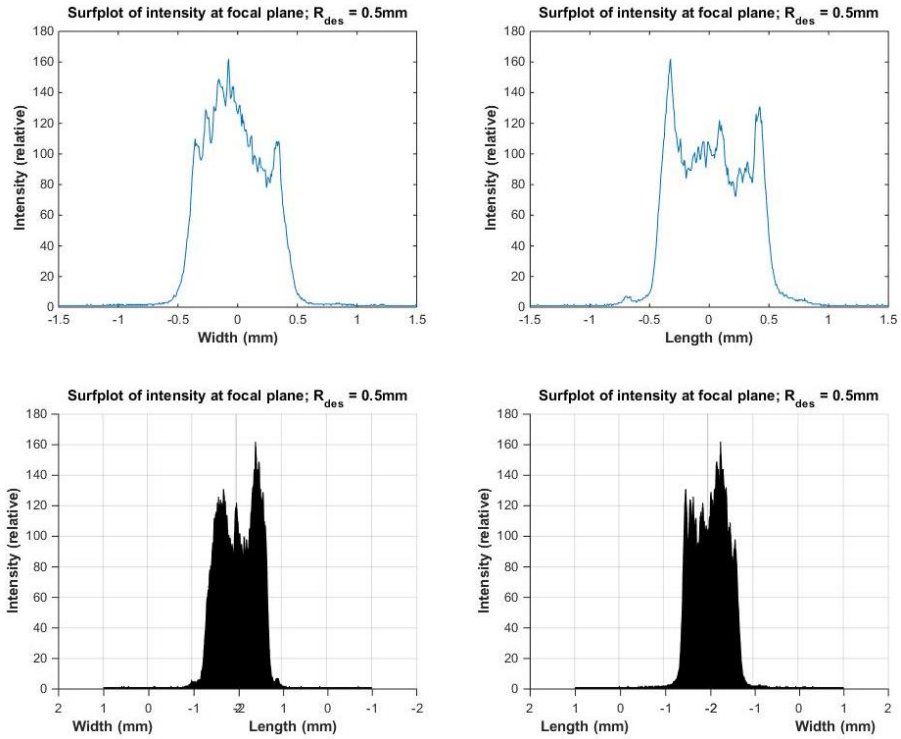


Figure 26 - Side view of intensity at focal plane with the voltages from table (4) to control the DM. The desired beam radius is 0.5 mm. Top left: side view from side 'width' side in figure (25). Top right: side view from ‘length’ side in figure(25). Bottom left: side view from rightmost corner in figure (25). Bottom right: side view from bottom corner in figure(25).

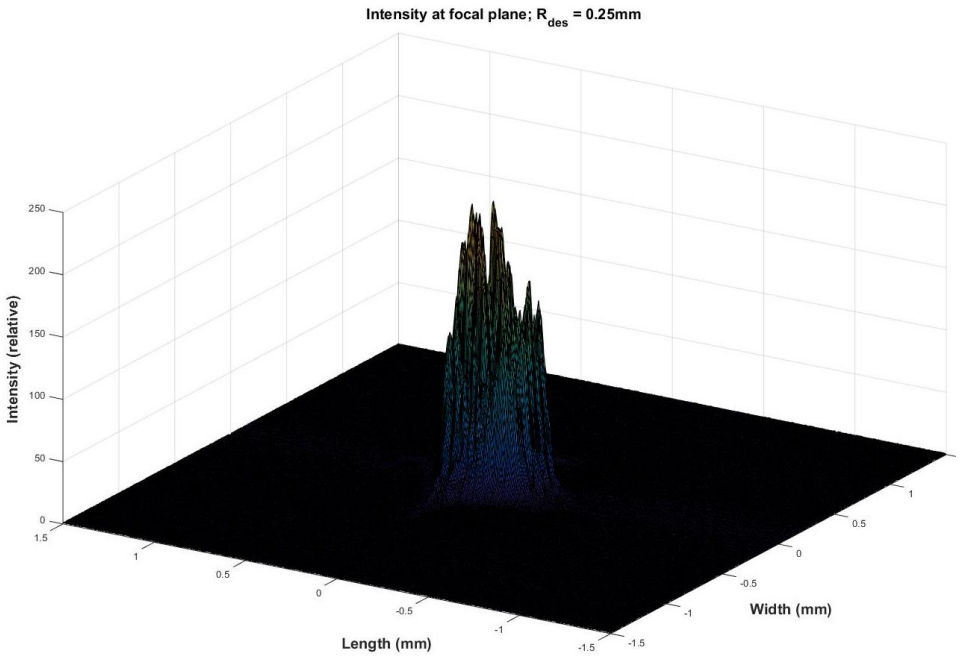


Figure 27 - Intensity distribution in the focal plane for a desired beam radius of 0.25 mm. The DM is controlled by the random walker algorithm, the voltages applied are given in table(4). The image is taken form the camera in the focal plane and then cropped the same as in figure (22).

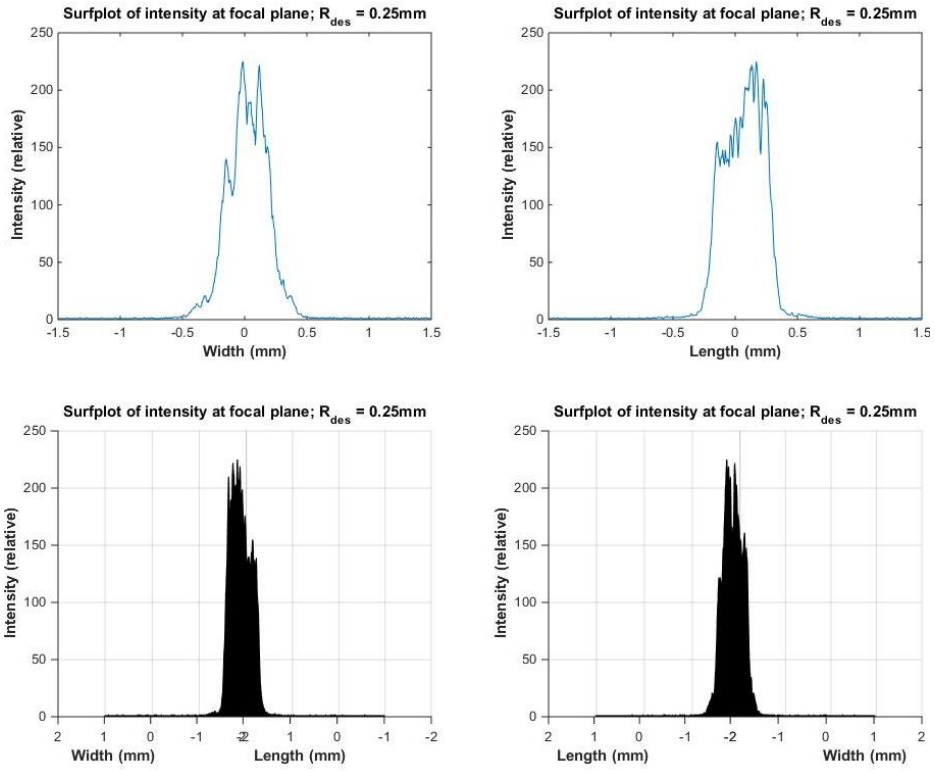


Figure 28 - Side view of intensity at focal plane with the voltages from table (4) to control the DM. The desired beam radius is 0.25 mm. Top left: side view from side 'width' side in figure (27). Top right: side view from 'length' side in figure(27). Bottom left: side view from rightmost corner in figure (27). Bottom right: side view from bottom corner in figure(27).

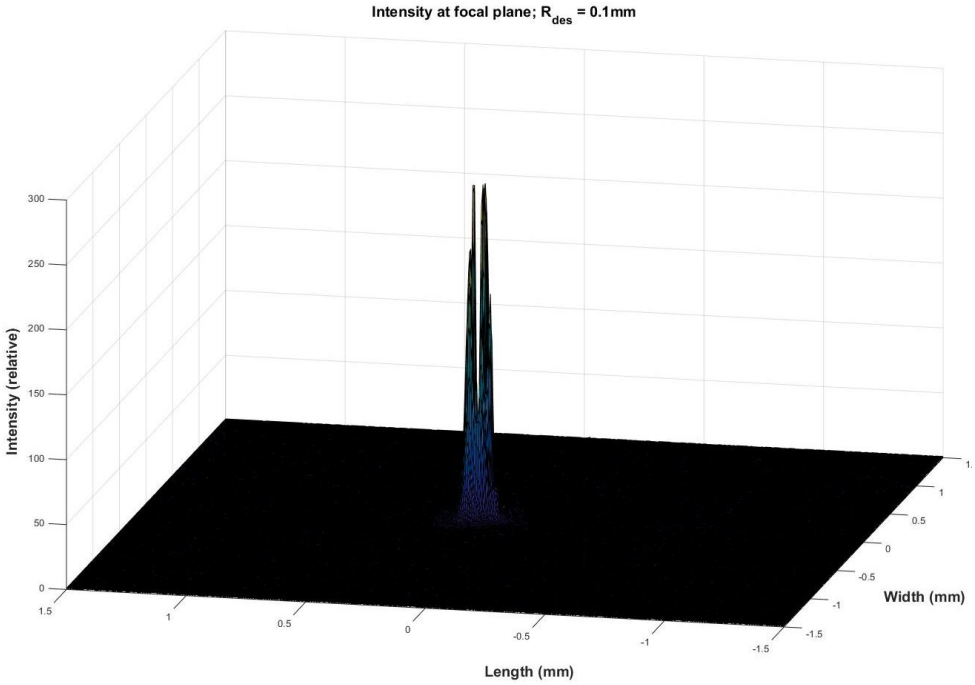


Figure 29 - Intensity distribution in the focal plane for a desired beam radius of 0.1 mm. The DM is controlled by the random walker algorithm, the voltages applied are given in table(4). The image is taken form the camera in the focal plane and then cropped the same as in figure (22).

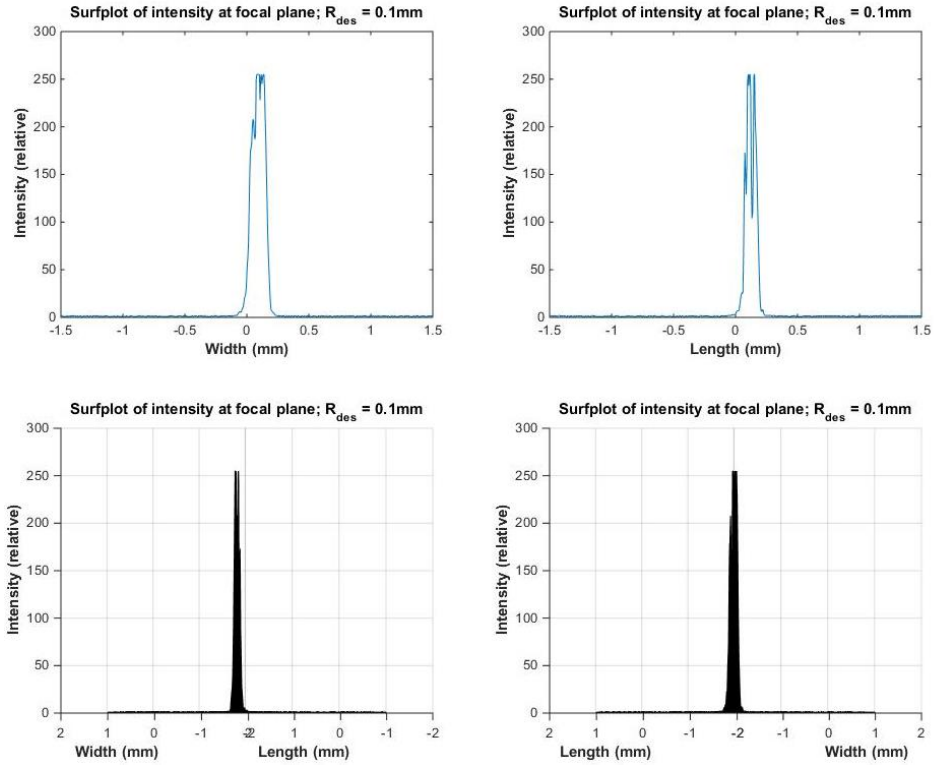


Figure 30 - Side view of intensity at focal plane with the voltages from table (4) to control the DM. The desired beam radius is 0.1 mm. Top left: side view from side 'width' side in figure (29). Top right: side view from 'length' side in figure(29). Bottom left: side view from rightmost corner in figure (29). Bottom right: side view from bottom corner in figure(29).

6 Discussion

In this chapter the results as presented in the previous chapter are discussed. Some preliminary conclusions are made. Whenever a RMS value is mentioned, what is meant is the RMS value of the difference between the result (e.g. simulation or measurement) and the optimal result, mostly being a top hat function.

6.1. Using the calculated phase delay

Function (2.6) is readily calculated using MATLAB. The gridsize used in all the simulations is 1280. This allows for a fine enough grid with enough detail. First, the whole setup is simulated with the parameters as presented in table (2). The aperture of the DM is actually larger than 20 mm, but it is effectively set to 20 mm by implementing a small aperture in front of the collimating lens.

When the phase delay function is used to propagate the beam, the results for larger beam sizes are shown in figure(5). As the optimal top hat function is shown as well, it is clearly visible that for larger β, upwards of a value of 10, the results are quite good. The RMS values are all smaller than 8%.

For smaller desired beam sizes, depicted in figure (6), the simulation gives less than satisfactory results. The RMS values increase, and it is also clearly visible that the intensity distribution and ideal top hat function diverge. For a desired beam size of 0.1 mm, corresponding to a β of 4, the difference is somewhat negligible, due to it being mostly a uniform shift in intensity, but for even lower values of β there arise more problems. For a β of 2, corresponding to a beam size radius of 0.05 mm, the intensity distribution becomes very pointy. Also the radius becomes less precise. For even lower values of β the results is completely dissatisfactory, and the RMS value becomes increasingly big.

6.2. Different apertures of the DM

In figure (7) there are simulations shown for different sizes of the DM aperture. This is done for both a desired beam radius of 2 mm and 0.5 mm. At a full aperture of 20 mm, there is little diffraction noticeable. When the aperture is decreased, some diffractive effects arise. However, the results are still quite uniform and the RMS value only increases very slightly. At an even smaller aperture of 4 mm, the results change drastically. The RMS value increases, but from figure (6) it is clearly visible that a lot of diffraction effects arise. Especially in the middle the intensity skyrockets. The desired beam radius has little effect on these results.

For all simulations and measurements, a DM aperture of 20 mm. is used for the best result.

6.3. Simulation with impulse response functions

The response functions are very closely fitted to the phase delay function, as is visible in figure (8). The error, depicted in figure(9), never is in excess of 1% of the total height of the phase function. The RMS value of the error is very small at 0.0007 microns. However, the error is quite jagged. Since the fitting process is entirely linear, to with for multiple values of β is not necessary. Both figure (8) and (9) would be scaled, without changing the relative error.

In table (3) the weights of the impulse response functions are given for different values of β. As explained in the chapter on the experimental setup, any value smaller than -1 or bigger than 1 corresponds to an displacement of the DM that is in excess of the maximum displacement of the DM's actuators. Therefore, these values of β are theoretically not possible to achieve with the used DM. All of the values outside the domain are marked grey; as is clearly visible a radius of 2 or 1 mm. is theoretically not obtainable. As is visible in the fourth column in table (3), a radius of 0.5 mm, or a β of 20 has weights that are almost 1. Based on these values it can be concluded that the usage of the DM is not advised for values of β larger than 20.

Figure (10) and (11) consist of images of the intensity at the focal plane. However, unlike figure (5) and (6), where the calculated phase delay is used, now the fitted impulse response functions are used. This is done for multiple values of β. Although all the results are less than satisfactory, it is clearly visible that there is much more distortion in the intensity profile for values of β above 20. Although at first hand this seems like an expected result, since the weights are outside the domain of the impulse response functions, this is not the case; the domain of the impulse response functions (and thus the finiteness of the displacement of the actuators) is not taken into account in this simulation. Although in theory the result should be better for larger values of β, which is also the case in the results using the phase delay function, this is not the case in these simulations. It is most likely due to the fact that the absolute error as depicted in figure (9) is bigger for larger values of β. Due to the linearity of both the fit and the impulse response functions, the error is scaled 1:1 with the value of β. Above a certain β the absolute error becomes too big and the imperfections in the phase delay caused by the fit acquires a dominant role.

For very small values of β the simulation breaks down as well. For values of β less than 10, the desired top hat intensity distribution becomes too small and the intensity distribution becomes a peak. The maximum intensity is greatly reduced as well by a factor of 10 when comparing a value of β of 10 to a value of 4. As mentioned in [1], smaller values of β are prone to these problems.

The only two simulations that give somewhat satisfactory results are that with β values of 20 and 10, as depicted in the bottom row of figure (10), and a value of 10 also depicted in the top left graph in figure (11), with a different scaling on the x axis. Although for β=10 the intensity distribution is relatively uniform, the desired beam radius is a factor 2 bigger than the acquired beam radius. For β=20, the radius is as desired, but the intensity distribution is not uniform, with values of the intensity in excess of 150% of the mean.

6.4. Measurements when applying voltages directly

Based on the results of the simulations, the weights for values of β=20, 10, 4 and 2 are used in the measurements. Other weights are discarded for reasons stated above.

The results of applying these weights directly to the DM are summarized in figure (12). For every desired beam radius the intensity profile is also 3D plotted and the side view from various angles are plotted as well. These results can be seen in figure 13-20.

It is clearly visible that the results are less than satisfactory. In all the results there is a narrow, long distortion in the intensity profile. This slit-like distortion has a much higher intensity than the rest of the profile, and as such make the results unfavourable.

For a β value of 20, corresponding to a beam radius of 0.5 mm, the profile is not big enough. This is best seen in the side view plots depicted in figure (14). The intensity profile is only nonzero up to a radius of 0.2 mm, over a factor of two less than the desired result. There is also little uniformity in the profile.

The next measurement, a β value of 10 or a radius of 0.25 mm, gives slightly better results. Although there is still a very high peak, and there is still little uniformity in the intensity profile, the radius of the beam resembles the desired beam radius somewhat more closely. Although not entirely radially symmetric, as seen in figure (16), the radii are in the 0.2 mm-0.25 mm range, which is a smaller error than in the previous result.

For β values of 4 and 2, a radius of 0.1 and 0.05 mm respectively, any intensity profile apart from the large slit-like distortion is not perceivable. Although from the bottom row in figure (10) it seems like there is a small round profile beneath the distortion, but the side view plots from figure (18) and (20) show that there are most likely part of the larger distortion.

The large distortion can be explained as astigmatism. Since the distortion occurs in all different measurements and is somewhat the same in all of them, it can be viewed as a static aberration. Because the response functions of the DM are linear, in theory this astigmatism could be cancelled by adding an extra set of weights to those presented in table (3).

6.5. Optimization measurements

The random walker approach is done for four values of the desired beam radius: 1, 0.5, 0.25 and 0.1 mm. The final voltages are displayed in table (4). Although not used, for reference the corresponding control signal value or weight from the response functions are calculated and shown in the table as well.

What is clearly visible in the table is that there are a lot of voltages that correspond to a control signal value that is outside the domain of the impulse response function (e.g. a voltage anywhere outside the domain of 0 to 3.7 Volt). Especially for larger values of β there are a lot of voltages outside the domain. These have no impact on the DM whatsoever, because the DAC outputs them as 0 or 3.7 volts anyway. Therefore, 13 of the 19 actuators are not retracted at all for the 1st measurement. Since for 6 consecutive actuators the voltages are all negative, it means that an considerable part of the DM's surface is flat.

This first measurement, for a radius of 1 mm, does not yield very good result. The intensity profile resembles a PSF more closely than a uniform distribution. This can be easily seen in figure (23). The top view, depicted in the top left picture in figure (22), shows that the distribution is also not radially symmetrical. Also, as is clearly visible in figure (24), the radius of non-zero intensity (the radius beyond there is no noticeable intensity) is about 0.5 mm., half the desired beam radius.

The 2nd and 3th measurement give more favourable results. From the top view images depicted in figure (22), it is clearly visible that they have a more radially symmetric intensity profile. Figure (25) & (26) and (27) & (28) for a radius of 0.5 mm. and 0.25 mm., respectively, show that they are still not completely uniform. However, for both the radius of non-zero intensity closely resemble the desired beam size. Apart from some local peaks, the intensities are relatively uniform.

For an even smaller radius, of 0.1 mm., the results is again unsatisfactory. As is visible in figure (29), the beam is seemingly split in half. Also, as can be seen from the side view in figure (30), the beam is shifted away from the centre. However, taking the simulation in figure (11) (top right) into account, the results is relatively good. Still, it is not in the absolute sense a favourable result.

6.6. Random walker algorithm for optimization

In figure (21), the error in between the image and the optimal intensity, which is the metric for the random walker algorithm, is plotted versus the iteration number. The algorithm converges relatively fast, but as is evident from the plots, the error does not converge to zero. There is always a considerable error that has to be taken into account.

The time that the algorithm takes heavily relies on the polling rate of the camera. This polling rate will be the most evident factor in the speed of the algorithm, since at every loop the camera has to be addressed and an image has to be taken. The calculations done in the algorithm are relatively simple and as such do not form an important factor in the speed of the algorithm.

Another advantage in using an optimization algorithm is that the static aberrations in the system, that are evident in the measurements with the directly applied voltages, are cancelled out awhile.

The algorithm is quite simple and easy to implement. However, the algorithm also does have some disadvantages [15].

One major disadvantage of the random walker algorithm is the fact that it does not allow for constraints on the output parameters. Therefore, the output signals can go off to arbitrary large values, both positive and negative. They can thus reach values beyond the domain of the output signal, which is very much so the case in the first and also the second measurement in this report. It may then be very unlikely for the algorithm to escape the local minimum that was found with these parameters.

7 Conclusions

7.1. Conclusions

Laser beam shaping with a low order deformable mirror is only doable for satisfactory results in a small region of applicability. For β values outside the region of 10 to 20, corresponding to a beam radius of 0.25 to 0.5 mm., the results suffer from either the limitations of the DM or the fundamental theory of laser beam shaping.

Directly applying calculated weights to the DM gives heavy aberrations and distortions. The acquired results show that this is not a favourable method of laser beam shaping. Further research may be able to improve these results.

The usage of the random walker algorithm allows for better results, but it cannot compensate for the limitations described above. Therefore, only shaping for values of β between 10 and 20 gives favourable results using this DM. The random walker algorithm has some disadvantages, but does yield good results, however in a small range of applicability.

7.2. Further research

Further research can be conducted to improve the beam shaping process. For both methods of achieving the result there can be made improvements on the result.

When using the calculated voltages from the fitting process, the most notorious improvements can be made on the static aberrations of the DM. Due to the linearity of the DM, these aberrations could be accounted for by adding an extra phase delay function to the DM. To acquire this phase delay function, an optimization can be done to achieve an perfect PSF (an airy pattern), for instance using the Beamtuner software [16].

Further research in the optimization method can be done by using another algorithm. The random walker algorithm is relatively simple, but has some disadvantages for the use presented in this report. More advanced algorithms can take the constraints on the output signals into account and can therefore, in theory, either better the result or reduce the algorithm time.

Further research options not specific to a certain method are for instance trying to achieve different intensity profiles. It is not recommended to try for different input intensity profiles, since most lasers emit a Gaussian profile, but the intensity profile in the focal plane can be changed very much so. Uniform intensities with another top-view shape, such as a square, a triangle or a rectangle can be tried to achieved. Non-uniform intensities are of smaller interest [1-3], but can be tried nonetheless.

List of references

1. Dickey, F.M. and S.C. Holswade, *Laser beam shaping : theory and techniques*. Optical engineering. 2000, New York: Marcel Dekker. xi, 428 p.
2. Henderson, B.G. and J.D. Mansell. *Laser beam shaping with membrane deformable mirrors*. in *Proc. SPIE*. 2008.
3. Ping, Y., et al. *Adaptive Laser Beam Shaping Using a Genetic Algorithm*. in *Mechatronics and Automation, 2007. ICMA 2007. International Conference on*. 2007.
4. Sharma, K.K., *Optics: principles and applications*. 2006: Academic Press.
5. *Lightpipes for Mathcad and MATLAB*. 2016 [cited 2016 6-2]; Available from: http://www.okotech.com/lightpipes-for-mathcad-and-matlab.
6. Goor, F.v. *Lightpipes for MATLAB manual*. 2016 [cited 2016 6-2]; Available from: http://www.okotech.com/images/LightPipes/manuals/lightpipes-for-matlab-manual.pdf.
7. *Handheld laser source HLS635*. 2016; Available from: http://www.thorlabs.de/newgrouppage9.cfm?objectgroup_id=5929&pn=HLS635References%20(simulation).
8. *Mrfit: mirror simulation software*. 2016 [cited 2016 6-02]; Available from: http://www.okotech.com/mrfit.
9. *Thorlabs AC254-200-A-ML*. 2015; Available from: http://www.thorlabs.de/thorproduct.cfm?partnumber=AC254-200-A-ML.
10. *Piezoelectric DM Wavefront correctors*. 2016; Available from: http://www.okotech.com/pdm.
11. *EDAC40 User manual*. 2013; Available from: http://www.okotech.com/images/pdfs/edac40_user_guide.pdf.
12. *High voltage amplfier units*. 2016; Available from: http://www.okotech.com/high-voltage-amplifier-units.
13. *Thorlabs AC508-500-A-ML*. 2015; Available from: https://www.thorlabs.com/thorproduct.cfm?partnumber=AC508-500-A-ML.
14. *Thorlabs DCC1545 High resulution USB 2.0 CMOS Camera*. 2015; Available from: http://www.thorlabs.com/thorProduct.cfm?partNumber=DCC1545M.
15. Cura, T., *A random search approach to finding the global minimum.* Int. J. Contemp. Math. Sciences, 2010. **5**(4): p. 179-190.
16. *BeamTuner*. 2016; Available from: http://www.okotech.com/beamtuner.

Appendix A

A.1. Lightpipes commands

1. LPBegin(gridsize , lambda , dimension)

Every Matlab script with Lightpipes code should start with this command. It initiates a reactangular wavefield with sides '*dimension*' of light with wavelength *'lambda*', divided over a matrix with '*gridsize*' dimensions. The light is collimated, thus it has constant phase. *'gridsize*' should be an even number.

2. LPFresnel(distance , field)

LPFresnel propagates the lightfield '*field*' '*distance*' units along the optical axis. It uses a convolution to calculate the diffractive integral.

3. LPLens(f , x_shift , y_shift , field)

This commands simulates a lens of focal length '*f*' at position (*x_shift*,*y_shift*). Since a lens is actually a phase retarder, it multiplies the *'field*' with a phase function of a lens. Please note that this is all the command does. It *does not* propagate the field to the focal plane.

4. LPGaussAperture(R , x_shift , y_shift , T , field)

This command puts a Gaussian aperture over the specified '*field*' with the centre at (*x_shift* , *y_shift*). It has radius *R* and the transmission (between 0 and 1) at the centre is *T*.

5. LPMultPhase(phase , field)

Use this command to multiply the *field* with an arbitrary phase function specified in *phase*. *phase* should be a square matrix of the same size as the *field*. It then multiplies *field* with exp(*i*phase*) in a pointwise manner.

6. LPIntensity(field)

This command simulates an intensity sensor, it calculates the intensity at every entered value in the *field* matrix. It outputs the intensity as a matrix of the same dimensions as *field*.

7. LPPhase(field)

To calculate the phase of the *field*, use this command. The output is a square matrix of the same dimensions as *field*. It calculates the phase as-is, and does not 'normalize' the phase to a value between 0 and 2π. This is the desired result for this application.

A.2. DAC and Ueye commands

1. *edac40*(*'open',1,'IP'*). This command opens the DAC for writing in MATLAB. It looks for a DAC at local IP-address XXX.XXX.XXX.X entered as a string. If no DAC is available at this address, it does not return a warning. The IP address of the DAC can be easily found using the ipconfig command in the command prompt from windows.
2. *edac40* (*'write',1,'signal'*). This command sends the vector 'signal' to the DAC. The vector can be up to 40 values long. Each channel outputs the respective value in *'signal'* as a voltage. However, it is not a 1:1 conversion; a signal of 0 corresponds to the minimum voltage of -12 volts.. Any signal below 0 or above the maximum is cut off and corresponds to either -12 or +12 Volts.
3. *edac40* (*'close',1*). Closes the DAC in MATLAB so it can be put to other use.

1. ueye('open',1,1)
   Opens the camera if present.
2. ueye('configure',1,0,(['*[Bottom X Bottom Y]*';'*[Width Height]*']),'*Max Intensity'*)

Configures the camera. Two things are specified: the active pixels and the maximum intensity. '*[Bottom X Bottom Y]'* are the lower bound on the pixel selection for both X and Y direction. Standard is [0 0]. '*[Width Height]*' is the width and the height of the cropped image. For full range use [1280 1024]. *'Max Intensity'* is an number anywhere between 1 and 255 that can be used as a cut off of high intensity values. For maximum intensity use 255.

3. ueye('settiming',1,*'frame time'*,*'frames per second'*,*'polling rate'*)

Sets the frame time, frames per second and polling rate of the camera. If values outside the boundaries of the camera are given, it is cut off to its maximum or minimum.

4. ueye('close',1)

Closes the camera for further use.

Appendix B

B.1. Main simulation code

```
clc,clf
m=1; nm=1e-9*m; um=1e-6*m; mm=1e-3*m; cm=1e-2*m; % Define lengths for
robustness

%% Define parameters

multipl=4;                                  % Multiplier for girdsize
lambda = 635*nm; k = (2*pi)./lambda;        % Wavelength
real_size = 20*mm;                          % Size of the grid which lightpipes
simulates
gridsize = multipl*320;                     % Amount of gridpoints

R_apt = 2*mm;                               % Radius of Gaussian beam
R1=3*mm;                                    % Desired beam radius
DM_apt=20*mm;                               % Aperture of the DM
f1=50*cm;                                   % Focal length of focusing lens

%% Initiate the field
Field = LPBegin(real_size,lambda,gridsize); % Begins the field with above
defined parameters.

%% Gaussian aperture
Field1 = LPGaussAperture(R_apt,0,0,1,Field); % Turn uniform intensity into
gaussian

%% Circ Aperture
Field1 = LPCircAperture(DM_apt,0,0,Field1); % A circular aperture to simulate
the aperture of the DM

%% Phase delay element
beta = R_apt*R1*2*pi/(lambda*f1); % Calculate beta parameter
Field2=LPMultPhase(beta.*chi,Field1); % Indtroduce the phase delay of the
beam shaping element

%% Lens
Field3 = LPLens(f1,0,0,Field2); % Indtroduce the phase delay of the lens

%% To CMOS
z1=f1;                                  % Define the propagation distance
Field4 = LPFresnel(z1,Field3);          % Propagate the field a distance z1

%% CMOS
Intensity = LPIntensity(1,Field4);  % Simulate CMOS

%% Plot the intensity at focal plane
figure(1); imshow(Intensity);
title(strcat('Intensity at CMOS; R=',num2str(R1),'mm'))
```

B.2. Calculate the phase delay function

```
%% Phase element
ksi=zeros(gridsize,gridsize);
size_correction = (real_size/(gridsize-1))/R_apt;
% This is the physical length that corresponds to one step in the grid,
divided by the incoming beam radius.


for i=1:gridsize/2
    for j=1:gridsize/2
        ksi(gridsize/2+i,gridsize/2+j)...
            =size_correction*sqrt(i^2+j^2);
% Ksi is a grid with a radial coordinate, normalized to equal one at the
incoming beam radius.

        [ksi(gridsize/2+i,gridsize/2+1-j),...
            ksi(gridsize/2+1-i,gridsize/2+1-j),...
            ksi(gridsize/2+1-i,gridsize/2+j)]...
            =deal(ksi(gridsize/2+i,gridsize/2+j));
% Copy the bottom right quadrant to the other quadrants

    end
end

alpha = @(x) (1-exp(1).^(-1.*(x.^2))).^(1/2);
% Define equation (2.6)

chi=zeros(gridsize,gridsize);
for i=1:gridsize/2
    for j=1:gridsize/2
        chi(gridsize/2+i,gridsize/2+j)...
            =integral(alpha,0,ksi(gridsize/2+i,gridsize/2+j));
% Calculate at every point on the grid the value of equation (2.6)

        if sqrt(i^2+j^2)*size_correction*R_apt > DM_apt/1.99;
            chi(gridsize/2+i,gridsize/2+j) = 0;
% Simulate finite aperture of DM
        end

        [chi(gridsize/2+i,gridsize/2+1-j),...
            chi(gridsize/2+1-i,gridsize/2+1-j),...
            chi(gridsize/2+1-i,gridsize/2+j)]...
            =deal(chi(gridsize/2+i,gridsize/2+j));
% Copy the bottom right quadrant to the other quadrants
    end
end
```

B.3. Fitting the response functions to the phase delay function

```
N_act = 19; % Number of actuators
%% Displacement of actuators corresponding to 300 V
PV_0 = 3.3777;
PV_1tm8 = 2.132;
PV_9 = 2.129;
PV_10tm17 = 1.476;
PV_18 = 1.468;

Exc(1)=PV_0;
for i=2:9
    Exc(i)=PV_1tm8;
end
Exc(10)=PV_9;
for i=11:18
    Exc(i)=PV_10tm17;
end
Exc(19)=PV_18;                      % Make one vector with displacement

%% Import response functions
for i=1:19
    filename = strcat(num2str(i-1), '_mirror.bmp');
    Resp_func{i}=imread(filename);
    Resp_func{i}=double(Resp_func{i})-
ones(320,320).*double(Resp_func{i}(1,1));
    Resp_func{i}=Exc(i)*Resp_func{i}./max(max(Resp_func{i}));
end


%% Backwards propagation (DO NOT EXECUTE FOR SIMULATION, ONLY CALCULATING
VOLTAGES TO DM)
z5 = 40*cm;
% Distance DM->Lens
Field_backwards = LPForvard(-z5,Field2);
% Propagate the field from the main simulation before the lens backwards.
chi=LPPhase(Field_backwards);
% Calculate the phase profile

%% Import simulated shape of DM
clear Phase

for k = 1:4
    for i = 1:160
        for j = 1:160
            Phase_downscale(160+i,160+j)...
                =(1/2).*(lambda./um./(2*pi)).*betas{k}...
                .*chi(gridsize/(2)+multipl*i-(multipl-
1),gridsize/(2)+multipl*j-(multipl-1));
% Downscale the phase delay function by subsampling. Every other 'multiplier
value'th entry is taken

            [Phase_downscale(160+i,161-j),Phase_downscale(161-i,161-
j),Phase_downscale(161-i,160+j)]...
                =deal(Phase_downscale(160+i,160+j));
% Copy bottom right quadrant to the others
```

```
        end
    end

%% Find parameters B;
    for i = 1:19;
        for j = 1:19
            RR(i,j) = sum(sum(Resp_func{i}.*Resp_func{j}));
% Calculate matrix A from equation (3.4)
        end
        YY(i)=sum(sum(Phase_downscale.*Resp_func{i}));
% Calculate vector c from equation (3.4)
    end

    B{k}=inv(RR)*YY';
% Solve for vector b from equation (3.4)

%% Build up fitted response function phase delay
    chi_lsqfit{k}=zeros(320,320);
    for i = 1:19;
        chi_lsqfit{k} = chi_lsqfit{k} + B{k}(i).*Resp_func{i};
% Build up the fitted phase delay function from the impulse response
functions
    end

    [Xmesh,Ymesh]=meshgrid(-real_size/2:real_size/319:real_size/2);
    [Xmeshq,Ymeshq] = meshgrid(-real_size/2:real_size/(320*multipl-
1):real_size/2);
    chi_lsqfit_intp{k}=interp2(Xmesh,Ymesh,chi_lsqfit{k},Xmeshq,Ymeshq);
% Interpolate the 320*320 fitted response function to gridsize*gridsize
end

%% Calculate RMS
error = Phase_downscale(160,:)-chi_lsqfit(160,:);
RMS = sum(1./(320).*error.^2);
```

B.4. Random walker implementation

```
clc
ueye('close',1)
edac40('close',1);
pixel_size=5.2.*um;
%% Opening camera and DAC
ueye('open',1,1)
ueye('configure',1,0,([[0 0];[1280 1024]]),255)
ueye('settiming',1,0.064,14.54,25e6)
edac40('open',1,'169.254.159.198')

%% Specify crop dimension
image = ueye('capture',1);
[Xc,Yc] = centroid(image);
% Use predefined centroid function that calculates the center of an matrix
using first moment.

if R1 > 1024.*pixel_size./2;
    fprintf('Warning: beam size larger than cmos sensor size')
% Break if sensor is too small
    break
end
length=ceil(3*R1./pixel_size);
% Define length of cropped image

Xb = floor(Xc-1.5*R1./pixel_size);
% Define bottom for X and Y coordinate
Yb = floor(Yc-1.5*R1./pixel_size);
crop_dim = [Xb Yb length length];

if R1 > 1024.*pixel_size./3;
% Use full image if R1 > 66% of sensor
    crop_dim = [1 1 1024 1024];
end
image2 = imcrop(image,crop_dim);
% Take initial image for reference in scaling

%% Scale I_0
Intensity_unscaled = Intensity;
% Take precalculated intensity profile (must be centered)
b = (1.5).*(R1.*gridsize./real_size);
% Define half of width of cropped image
Ixb = floor(gridsize/2 - b);
Ixt = ceil(gridsize/2 + b);
Iyb = floor(gridsize/2 - b);
Iyt = ceil(gridsize/2 + b);

Intensity_crop = Intensity_unscaled(Ixb:Ixt,Iyb:Iyt);
[Xdim,Ydim] = size(Intensity_crop);
% Crop intensity such that physical size is same as captured image

[Xdimq,Ydimq] = size(image2);
[Xmesh_int, Ymesh_int] = meshgrid(1:Xdim,1:Ydim);
[Xmesh_intq,Ymesh_intq] = meshgrid(1:(Xdim-1)/(Xdimq-1):Xdim,1:(Ydim-
1)/(Ydimq-1):Ydim);
```

```
I_0=interp2(Xmesh_int,Ymesh_int,Intensity_crop,Xmesh_intq,Ymesh_intq);
% Interpolate such that I_0 and captured image have the same size

%% Specify initial guess
voltages = ctrlsign(B);
% Specify initial guess with calculated voltages, Ctrlsign is function that
converts weights into voltages
I_meas = imgrab(voltages,crop_dim);
% imgrab is a predefined funtion that sets the DAC to 'voltages' and then
grabs an image, crops it for 'crop_dim' and outputs it as a matrix with type
double entries.
I_delta = Idiff(I_0,I_meas);
% Compute I_delta using predefined Idiff function

%% Specify random walker parameter and treshold
maxvolt=3.7;
% 0 - 3.7 is the region of voltages that is valid
a=0.2*maxvolt;                              % Define random walker parameter
treshold=0.01;                                          % Define threshold

%% For loop
figure(5)
ref_count = 0;
for i = 0:10000000;

%% This can be used to optimize the image intensity by altering the exposure
time. Code by T. Agbana, DCSC TU Delft.
%     for j = 1:3;
%          ueye('settiming',1,timeval,14.54,25e6);
%          image = ueye('capture',1);
%         % imcrop(image,[])
%            psf= imcrop(image,crop_dim); %Small region of rest
%          test_saturation = max(max(psf));
%         if test_saturation > 237;
%             timeval = 0.95 * timeval;
%         elseif test_saturation < 227;
%             timeval = 1.1 * timeval;
%         else
%             break;
%         end
%     end
%
    newvoltages = voltages - a./2.*ones(20,1) + a.*rand(20,1);
% Introduce perturbation on input signal
    I_meas_n = imgrab(newvoltages,crop_dim);
% Grab the new image using predefined imgrab function
    I_delta_n = Idiff(I_0,I_meas_n);
% Calculate the difference between new and old
    imshow(I_meas_n./max(I_meas_n(:)))
% Show new image
    if I_delta_n < I_delta;
% Compare new and old
        I_delta = I_delta_n;
% Set new metric as old
        voltages = newvoltages;
```

```
% Set new voltages as old
        ref_count = ref_count+1;
% Update new reference counter
    end
    if I_delta <= treshold;
% Check for threshold
        fprintf('Optimized, \n The voltages are:\n')
        disp(voltages)
            break
    end
    fprintf('Iteration number = %d, I_delta_old = %f2, I_delta_n = %f2, new
reference counter = %d\n',i,I_delta,I_delta_n,ref_count)
% Display info during runtime
end

%%
ueye('close',1)
edac40('close',1);
```