BABEŞ–BOLYAI UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF PHYSICS
SPECIALIZATION: COMPUTATIONAL PHYSICS

Master's Thesis

Determination of laser beam properties

ADVISOR:

BORBÉLY SÁNDOR, PHD.
UNIVERSITY LECTURER

AUTHOR:

NAGY MÓNIKA

2025

UNIVERSITATEA BABEŞ–BOLYAI, CLUJ-NAPOCA
FACULTATEA DE FIZICĂ
SPECIALIZAREA FIZICA COMPUTAȚIONALĂ

Lucrare de disertație

Determinarea proprietăților fasciculului laser

CONDUCĂTOR ŞTIINŢIFIC:
LECTOR DR. BORBÉLY SÁNDOR

ABSOLVENT:
NAGY MÓNIKA

2025

BABEŞ–BOLYAI UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF PHYSICS
SPECIALIZATION: COMPUTATIONAL PHYSICS

Master's Thesis

Determination of laser beam properties

Abstract

In this project a laser mounted on a 3D printer was used to burn small holes into a surface made of black tape or gold-coated foil, which then were analyzed to determine some of the key properties of the laser beam. The laser moved along the Z-axis, and at each predefined height, it fired a short pulse to burn a single hole into the material. After this, microscope images were taken of each burn hole. A custom Python-based image processing program was then used to analyze the holes, measuring their area as well as the lengths of the minor and major axes, assuming the shape to be elliptical. These measurements were then used to calculate key characteristics of the laser beam, such as the beam waist, the location of the focal point, and the M^2 beam quality factor.

Overall, the results generally agreed with the predictions of our simple model, and for longer laser pulses, it was possible to consistently determine the pulse parameters, which were not significantly influenced by the pulse duration and by the surface material. The measured beam width was around $90\ \mu m$ and the beam quality factor estimated between 40 - 50, which is close to the values provided by the manufacturer of the laser head.

However, when analyzing the shape of the hole, found out that the behavior along the minor axis did not always match theoretical expectations. This suggests that the current theoretical model may require further development.

It was also found that very short laser pulses produced more scattered and less reliable data, in contrast, longer laser pulses led to more stable and clearly defined results, which made the beam characteristics easier to evaluate.

One limitation of the setup was that the sample was fixed on a frame, provided good stability during the experiment, but it also restricted the number of sample points along the Z-axis. As a result, in some cases the measurement range was too narrow to fully capture the expected double-peak structure. This limitation only became obvious during analysis. For future work, it will be important to redesign the sample holder to allow a wider Z-range, both upward and downward.

2025 NAGY MÓNIKA

ADVISOR:
BORBÉLY SÁNDOR, PHD.
UNIVERSITY LECTURER

Contents

Introduction

Lasers are now an essential tool in a wide range of fields, including industrial manufacturing, medical technology, scientific research, and even everyday consumer electronics.
However, in order to use a laser system effectively, it is crucial to have a solid understanding of the laser beam's characteristics. These include properties such as the beam shape, diameter, intensity distribution, focus, divergence, and stability over time.

There are several methods to characterize a laser beam. One of the simplest and most accessible methods is to direct the laser beam onto a piece of material and examine the burn mark or pattern it leaves behind [1] . While this doesn't provide as much quantitative detail as professional instruments, it can offer useful visual information about the beam's focus and symmetry.

In more advanced applications, dedicated beam profiling equipment is used. These instruments often rely on two-dimensional imaging technologies, like CCD or CMOS camera sensors, which can display the intensity distribution of the beam in real time [1]. Other setups include mechanical scanning systems, such as slit, knife-edge, or pinhole methods, which measure the beam's profile by scanning across one axis at a time [1]. Additional techniques involve thermal sensors, pyroelectric detectors (especially useful for infrared beams) [2], and integrating sphere or encircled energy measurements that assess how the energy is spatially distributed within the beam [1].

In this particular project, we used a laser head mounted on a 3D printer to burn small holes into a surface at different laser-surface distances. The process began by sending a programmed set of coordinates and laser control commands to the printer in the form of a *G-code* file. At each burn-cycle, the laser was turned on for a very short time-period, producing a tiny, visible burn mark. Once a full pattern of holes was created, the sample was placed under a digital microscope, which captured high-resolution images of each mark. This allowed for a visual inspection and comparison of the laser's effect at different points corresponding to different laser-surface distances.

To extract meaningful data from the images, we developed a Python-based image proces-

sing program. The script was able to automatically detect the outlines of the burned holes and measure key it's relevant parameters such as diameter, area, and higher order moments. Fitting this data with simple models the parameters of the laser beam (beam focal point location, beam waist, and the M^2 beam quality factor) were extracted.

This updated "burn hole" method for the characterization of laser beams offers results comparable to precision of laser profiling tools. It also had the advantage of being low-cost, easily repeatable.

1. chapter

Theory

1.1. Derivation of the Paraxial Wave Equation

In free space, the propagation of electromagnetic waves is generally described by the scalar Helmholtz equation:

$$\nabla^2 \widetilde{\varepsilon}(x,y,z) + k^2 \widetilde{\varepsilon}(x,y,z) = 0, \tag{1.1}$$

where $\widetilde{\epsilon}(x,y,z)$ denotes the complex phasor representation of the electric field, and $k = \frac{2\pi}{\lambda}$ is the wave number, or propagation constant, in the medium.
Assuming that the wave propagates along the Oz-axis in an isotropic and homogeneous medium, the field can be written as

$$\widetilde{\epsilon}(x,y,z) = \widetilde{h}(x,y,z)\, e^{-jkz}, \tag{1.2}$$

where $\widetilde{h}(x,y,z)$ is a slowly varying complex amplitude that describes the transverse beam profile.
Substituting this expression into Eq. (1.1) the paraxial wave equation

$$\frac{\partial^2 \widetilde{h}}{\partial x^2} + \frac{\partial^2 \widetilde{h}}{\partial y^2} - 2jk\frac{\partial \widetilde{h}}{\partial z} = 0. \tag{1.3}$$

This equation can be written more compactly using the transverse Laplacian operator $\nabla_\perp^2$, defined over the transverse coordinates

$$\nabla_\perp^2 \widetilde{h}(\boldsymbol{r}_\perp, z) - 2jk\frac{\partial \widetilde{h}(\boldsymbol{r}_\perp, z)}{\partial z} = 0, \tag{1.4}$$

where $\boldsymbol{r}_\perp = (x, y)$ in Cartesian coordinates, or alternatively $\boldsymbol{r}_\perp = (r, \theta)$ in cylindrical coordinates. The operator $\nabla_\perp^2$ acts only in the transverse plane.

1.2. Huygens' Integral

A general solution to the scalar wave equation that physically represents a uniformly diverging spherical wave from a point source located at position $\boldsymbol{r}_l$ can be expressed as

$$\widetilde{\epsilon}(\boldsymbol{r}; \boldsymbol{r}_l) = \frac{e^{ik\,\rho(\boldsymbol{r}, \boldsymbol{r}_l)}}{\rho(\boldsymbol{r}, \boldsymbol{r}_l)}, \tag{1.5}$$

where $\widetilde{\epsilon}(\boldsymbol{r}; \boldsymbol{r}_l)$ denotes the field at observation point $\boldsymbol{r}$ due to a point source located at $\boldsymbol{r}_l$, and

$$\rho(\boldsymbol{r}, \boldsymbol{r}_l) \equiv \sqrt{(x - x_l)^2 + (y - y_l)^2 + (z - z_l)^2} \tag{1.6}$$

is the distance between the source and observation points.

In the Fresnel (or paraxial) approximation the spherical wave in Eq. (1.5) simplifies to a paraxial spherical wave

$$\widetilde{\epsilon}(x, y, z) \approx \frac{1}{z - z_l} \exp\left[-jk(z - z_l) - ik\frac{(x - x_l)^2 + (y - y_l)^2}{2(z - z_l)}\right]. \tag{1.7}$$

Neglecting the overall phase factor $\exp[-ik(z - z_l)]$ $\widetilde{h}(x, y, z)$ is given by

$$\widetilde{h}(x, y, z) = \frac{1}{z - z_l} \exp\left[-ik\frac{(x - x_l)^2 + (y - y_l)^2}{2(z - z_l)}\right]. \tag{1.8}$$

1.3. Gaussian spherical beams

The field distribution of a spherical wave emanating from a complex source point can, under the paraxial approximation, be expressed as

$$\widetilde{h}(x, y, z) = \frac{1}{\widetilde{g}(z)} e^{-ik\frac{x^2 + y^2}{2\widetilde{g}(z)}}, \tag{1.9}$$

where $\widetilde{g}(z)$ is the complex beam parameter, defined as

$$\widetilde{g}(z) = \widetilde{g}_0 + (z - z_0), \tag{1.10}$$

with $\widetilde{g}_0$ being the complex beam parameter at reference position z_0.

Equation (1.9) can be rewritten in the standard Gaussian beam form commonly used in laser optics

$$\widetilde{h}(x, y, z) = \frac{1}{\widetilde{g}(z)} e^{-ik\frac{x^2+y^2}{2R(z)} - \frac{k(x^2+y^2)}{w^2(z)}}, \tag{1.11}$$

where:

- $R(z)$ is the radius of curvature of the wavefront,
- $w(z)$ is the beam (spot) radius or waist at position z.

The beam width $w(z)$ describes how the radius of the beam evolves as it propagates along the z-axis. It is given by

$$w(z) = w_0 \sqrt{1 + \left(\frac{z}{z_0}\right)^2}, \tag{1.12}$$

where w_0 is the beam waist (minimum spot size) at the focus and z_0 is the Rayleigh range, which characterizes the distance over which the beam remains approximately collimated [3].

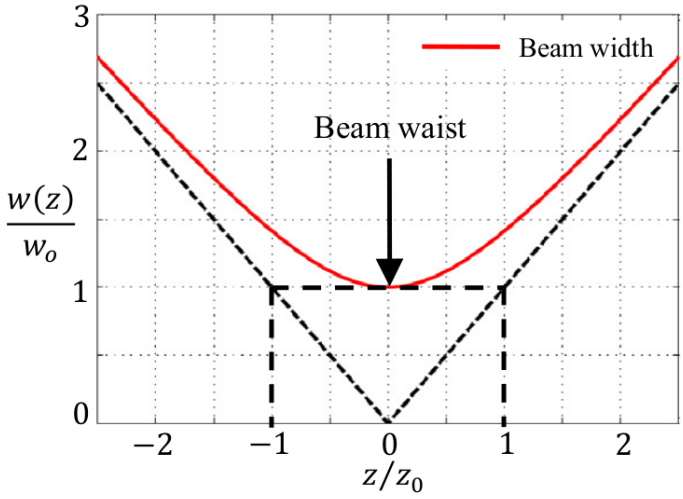
At $z = 0$, there is a minimum, which is designated as beam waist, w_0, as shown in Figure 1.1 .

The radius of curvature of the wavefront of a Gaussian beam profile is represented by

$$R\left(z\right) = z\left[1 + \left(\frac{z}{z_0}\right)^2\right]. \tag{1.13}$$

The optical intensity of the Gaussian beam is expressed by

$$I\left(r, z\right) = \frac{2P_0}{\pi w^2} exp\left[-\frac{2r^2}{w^2}\right] \tag{1.14}$$



1.1. Figure. Width $w(z)/w_0$ of the Gaussian beam depending on the axial distance z/z_0.

or

$$I\left(r,z\right)=\frac{2P_0}{\pi w_0^2\left[1+\left(\frac{z}{z_0}\right)^2\right]}exp\left[-\frac{2r^2}{w_0^2\left[1+\left(\frac{z}{z_0}\right)^2\right]}\right], \tag{1.15}$$

and at the $z=0$, $r=0$ points is given by

$$I_0=\frac{2P_0}{\pi w_0^2}. \tag{1.16}$$

We assumed, that the laser beam burns (ablates) our test sheet if the laser beam that exceeds critical intensity, which is parametrized as

$$I_{cr}=\alpha I_0, \tag{1.17}$$

substituting this into (15),

$$\alpha I_0=I_0\frac{1}{\left[1+\left(\frac{z}{z_0}\right)^2\right]}exp\left[-\frac{2r_{cr}^2}{w_0^2\left[1+\left(\frac{z}{z_0}\right)^2\right]}\right],$$

which can be rearranged as

$$\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]=exp\left[-\frac{2r_{cr}^2}{w_0^2\left[1+\left(\frac{z}{z_0}\right)^2\right]}\right].$$

By calculating the logarithm of the above expression we obtained

$$ln\left[\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]\right]=-\frac{2r_{cr}^2}{w_0^2\left[1+\left(\frac{z}{z_0}\right)^2\right]},$$

from which r_{cr}^2 can be expressed as

$$r_{cr}^2=-\frac{w_0^2}{2}\left[1+\left(\frac{z}{z_0}\right)^2\right]ln\left[\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]\right]. \tag{1.18}$$

The zero points of the derivative of r^2 according to z give the maximums and the minimum of the function (1.18),

$$\frac{d(r_{cr}^2)}{dz}=-w_0^2\frac{z}{z_0^2}\left[1+ln\left[\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]\right]\right].$$

So if $\frac{d(r_{cr}^2)}{dz}=0$, two cases occur, namely the first case when $z=0$, then

$$r_{cr}^2=-\frac{w_0^2}{2}ln(\alpha). \tag{1.19}$$

In the second case $1+ln\left[\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]\right]=0$. Moving the 1 to the other side gives the following equation

$$ln\left[\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]\right]=-1,$$

then taking the exponential of both sides gives

$$\alpha\left[1+\left(\frac{z}{z_0}\right)^2\right]=\frac{1}{e},$$

after rewriting this leads to

$$1+\left(\frac{z}{z_0}\right)^2=\frac{1}{\alpha e}.$$

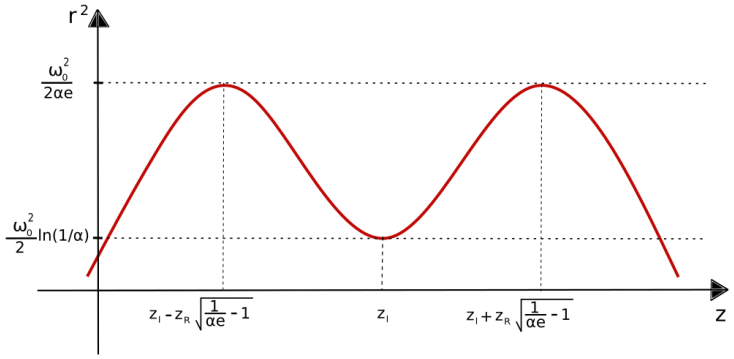
There are real solutions only if $\alpha<\frac{1}{e}$, in this case

$$z=\pm z_0\sqrt{\frac{1}{\alpha e}-1}, \tag{1.20}$$

and the r_{cr}^2 is

$$r_{cr}^2=\frac{w_0^2}{2\alpha e}. \tag{1.21}$$

Considering that $\alpha<1/e$, the first extreme point from 1.19 at $z=0$ gives a minimum, and the other two extreme ponits from 1.20, 1.21 give two maximum (Figure 1.2).



1.2. Figure. The maximum and minimum points of r^2 as a function of z if $\alpha<1/e$.

An important practical consideration in Gaussian beam optics is how quickly an ideal Gaussian beam expands due to diffraction as it propagates away from the waist. More specifically, we are interested in determining the propagation distance over which a collimated Gaussian beam maintains its narrow width before significant spreading occurs.

This characteristic distance is defined as the point along the propagation axis at which the beam radius increases by a factor of $\sqrt{2}$, meaning the beam's cross-sectional area has doubled compared to its minimum value at the waist. This distance is known as the **Rayleigh range**, and is denoted by z_0. It is given by

$$z_0 = \frac{\pi w_0^2}{\lambda}, \tag{1.22}$$

where w_0 is the beam waist radius, and λ is the wavelength of the beam.
Within a distance approximately equal to z_0 from the beam waist, the beam remains relatively well-collimated. Beyond this range, diffraction-induced divergence becomes more pronounced, and the beam begins to expand significantly.

$$z = z_0 \equiv z_R \equiv \frac{\pi w_0^2}{\lambda}, \tag{1.23}$$

which is called the *"Rayleigh range"* .

To develop a more general and practical framework for characterizing beam propagation, we first recall that the Gaussian beam spot size $w(z)$ evolves with axial distance according to

$$w^2(z) = w_0^2 + \left(\frac{\lambda}{\pi w_0}\right)^2 (z - z_l)^2 = w_0^2 \left[1 + \left(\frac{z - z_l}{z_R}\right)^2\right], \tag{1.24}$$

where w_0 is the minimum spot size (beam waist) located at $z = z_l$, and $z_R = \pi w_0^2/\lambda$ is the Rayleigh range. The far-field angular divergence of the beam is then given by $\theta = \lambda/(\pi w_0)$.

To extend this description to arbitrary, possibly non-Gaussian laser beams, we begin by noting that the spot size of a fundamental Gaussian beam can be related to its transverse intensity variance. Specifically, for a TEM_{00} mode

$$w_x = 2\sigma_x, \quad w_0 = 2\sigma_{0x}, \tag{1.25}$$

where σ_x is the standard deviation of the intensity distribution in the x-direction. A similar relation holds in the y-direction.

Inspired by this, we define the generalized real-beam spot size for an arbitrary beam as:

$$W_x(z) \equiv 2\sigma_x(z), \quad W_y(z) \equiv 2\sigma_y(z), \tag{1.26}$$

where $\sigma_x(z)$ and $\sigma_y(z)$ are the root-mean-square widths of the intensity distribution $I(x, y, z)$ at a fixed propagation distance z. Based on this definition, it can be shown that the generalized spot sizes evolve according to

$$W_x^2(z) = W_{0x}^2 + M_x^4 \left(\frac{\lambda}{\pi W_{0x}}\right)^2 (z - z_{lx})^2 = W_{0x}^2 \left[1 + \left(\frac{z - z_{lx}}{z_{Rx}}\right)^2\right], \tag{1.27}$$

with a similar expression for $W_y^2(z)$, involving the parameters W_{0y}, z_{ly}, and M_y^2.

The crucial additional parameter in this generalized formalism is the beam quality factor M^2. This factor quantifies how closely a real beam resembles an ideal Gaussian for a perfect TEM_{00} beam, $M^2 = 1$ and for a real beam, $M^2 > 1$ indicates increased divergence and/or distortion. The beam quality factor reflects how much a real beam deviates from the ideal case in terms of the product of its near-field waist and far-field divergence.

In summary, a general real laser beam can be fully characterized by the waist asymmetry ($W_{0x} \neq W_{0y}$), astigmatism ($z_{lx} \neq z_{ly}$), and divergence asymmetry ($\frac{M_x^2}{W_{0x}} \neq \frac{M_y^2}{W_{0y}}$). This formalism provides a powerful and practical toolset for analyzing arbitrary laser beams in both research and applied optics.

In this case the optical intensity is expressed by

$$I\left(x, y, z\right) = \frac{2P_0}{\pi W_x(z) W_y(z)} e^{-\frac{2x^2}{W_x^2}} e^{-\frac{2y^2}{W_y^2}}, \tag{1.28}$$

and for $x = 0$, $y = 0$, $z = 0$

$$I_0 = \frac{2P_0}{\pi W_{0x} W_{0y}}. \tag{1.29}$$

We assumed again, that the laser beam burns (ablates) out test sheet if the laser beam that

exceeds critical intensity, which is parametrized as

$$I\left(x,y,z\right)=\alpha I_0. \tag{1.30}$$

Substituting the explicit form of the intensity we obtain

$$\frac{2P_0}{\pi W_x(z)W_y(z)}exp\left[-\frac{2x_{cr}^2}{W_x^2(z)}-\frac{2y_{cr}^2}{W_y^2(z)}\right]=\alpha\frac{2P_0}{\pi W_{0x}W_{0y}}, \tag{1.31}$$

divided on both sides by $\frac{2P_0}{\pi}$ the equation simplifies to

$$\frac{1}{W_x(z)W_y(z)}exp\left[-\frac{2x_{cr}^2}{W_x^2(z)}-\frac{2y_{cr}^2}{W_y^2(z)}\right]=\alpha\frac{1}{W_{0x}W_{0y}}, \tag{1.32}$$

after rearrangement and logarithmic calculation we get

$$\frac{2x_{cr}^2}{W_x^2(z)}+\frac{2y_{cr}^2}{W_y^2(z)}=-\ln\left(\alpha\frac{W_x(z)W_y(z)}{W_{0x}W_{0y}}\right). \tag{1.33}$$

Substituting the explicit form of $W_x(z)$ and $W_y(z)$ the equation will look like this:

$$\frac{2x_{cr}^2}{W_{0x}^2\left[1+\left(\frac{z-z_{lx}}{z_{Rx}}\right)^2\right]}+\frac{2y_{cr}^2}{W_{0y}^2\left[1+\left(\frac{z-z_{ly}}{z_{Ry}}\right)^2\right]}=-ln\left[\alpha\sqrt{1+\left(\frac{z-z_{lx}}{z_{Rx}}\right)^2}\sqrt{1+\left(\frac{z-z_{ly}}{z_{Ry}}\right)^2}\right]. \tag{1.34}$$

From the above equation it can be observed, that the shape of the burn-hole is an ellipse. The half-axes of the ellipse can be determined by applying Eq. 1.34 for $x_{cr}=0$ and $y_{cr}=0$:

$$x_{cr}^2=-\frac{W_{0x}^2}{2}\left[1+\left(\frac{z-z_{lx}}{z_{Rx}}\right)^2\right]ln\left[\alpha\left[1+\left(\frac{z-z_{lx}}{z_{Rx}}\right)^2\right]\right] \tag{1.35}$$

$$y_{cr}^2=-\frac{W_{0y}^2}{2}\left[1+\left(\frac{z-z_{lx}}{z_{Rx}}\right)^2\right]ln\left[\alpha\left[1+\left(\frac{z-z_{lx}}{z_{Rx}}\right)^2\right]\right]$$

, and the zero points of the derivative give the maximum and the minimums, namely at $z=z_{0x}$ and $z=z_{0y}$

$$x_{cr}^2 = -\frac{W_{0x}^2}{2} ln(\alpha) \tag{1.36}$$

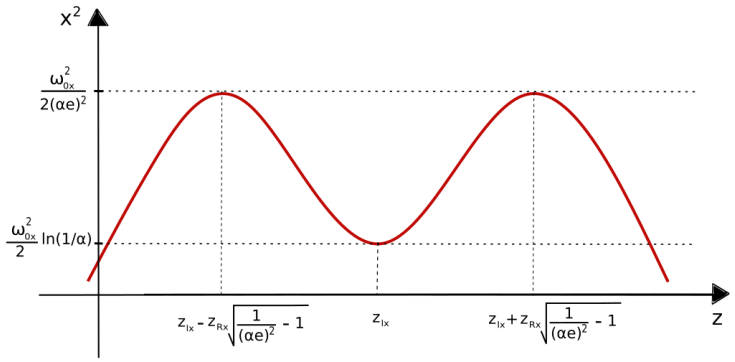
$$y_{cr}^2 = -\frac{W_{0y}^2}{2} ln(\alpha),$$

which is a minimum for $\alpha < \frac{1}{e}$, and at $z = z_{lx} \pm z_{Rx}\sqrt{\frac{1}{(\alpha e)^2} - 1}$ and $z = z_{ly} \pm z_{Ry}\sqrt{\frac{1}{(\alpha e)^2} - 1}$ points

$$x_{cr}^2 = \frac{W_{0x}^2}{2(\alpha e)^2} \tag{1.37}$$

$$y_{cr}^2 = \frac{W_{0y}^2}{2(\alpha e)^2}, \tag{1.38}$$

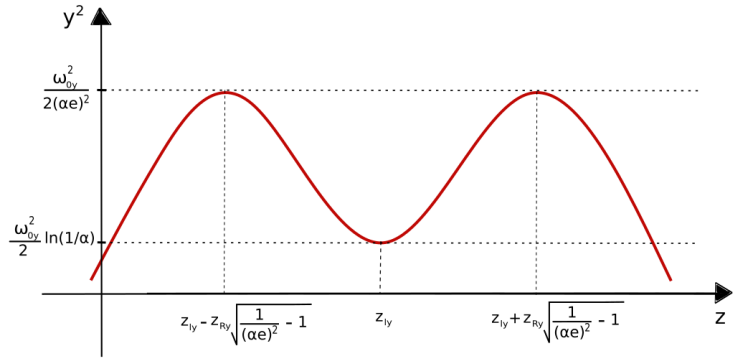
which is two maximum for $\alpha < 1/e$. These maximums and the minimum are shown on the Figure 1.3, 1.4.



1.3. Figure. The maximum and minimum points of x_{cr}^2 as a function of z if $\alpha < 1/e$.

1.4. Image moments

Image moments are used to describe the distribution and intensity of pixels in an image. They help determine an object's size, center, and orientation, making them especially useful in image



1.4. Figure. The maximum and minimum points of y_{cr}^2 as a function of z if $\alpha < 1/e$.

processing.

For a two dimensional continuous function $f(x, y)$ the moment is defined as:

$$M_{ij} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^i y^j f(x, y)\, dxdy. \tag{1.39}$$

In image processing we work with discrete pixels rather than continuous functions. Therefore, instead of integrals, we use a double summation to calculate the moments of an image. For a grayscale image the image moments calculated as:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y), \tag{1.40}$$

where x, y are the row and column indexes and $I(x, y)$ is the intensity at the location (x, y). When $i = 0$ and $j = 0$ the equation simplifies to the zero-order moment

$$M_{00} = \sum_x \sum_y I(x, y). \tag{1.41}$$

In a grayscale image this corresponds to the sum of pixel intensity values. In a binary image (as it is the situation of the present work), this moment gives the total number of non-zero pixels in the object, which corresponds to the object's area.

The average position of all the points is given by the centroid, which can be calculated as

$$(\overline{x}, \overline{y}) = (\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}}), \tag{1.42}$$

which in our case gives the center of the burn-hole on the image.

The central moments describe properties of the shape in an image. They are calculated by

$$\mu_{pq} = \sum_x \sum_y (x - \overline{x})^p (y - \overline{y})^q I(x, y). \tag{1.43}$$

We can use the second order central moments to build a covariance matrix, which shows how the shape is spread out and oriented (i.e. to determine the geometric properties of the binary object). The covariance matrix of the binary object is

$$cov[I(x, y)] = \frac{1}{\mu_{00}} \begin{pmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02}, \end{pmatrix} \tag{1.44}$$

and the eigenvectors of this covariance matrix correspond to the major and minor axes of the equivalent ellipse. If the main axes lie along the x and y axes, then $\mu_{11} = 0$, so

$$cov[I(x, y)] = \frac{1}{\mu_{00}} \begin{pmatrix} \mu_{20} & 0 \\ 0 & \mu_{02} \end{pmatrix} => \lambda_1 = \frac{\mu_{20}}{\mu_{00}}, \lambda_2 = \frac{\mu_{02}}{\mu_{00}}. \tag{1.45}$$

The discrete moments reflect the geometry of the shape. The corresponding continuous analogues of these discrete moments are represented as integrals. For an ellipse, the second-order central moments can be analytically calculated, and from the results the semi-axes of the ellipse can be expressed as function of centroid moments.
Starting from the ellipse equation

$$\frac{x^2}{a} + \frac{y^2}{b} = 1, \tag{1.46}$$

we write the second-order moments for a continuous distribution as

$$\mu_{00} = \int \int dxdy = \pi ab \tag{1.47}$$

$$\mu_{20} = \int \int x^2 dxdy \quad \text{and} \quad \mu_{02} = \int \int y^2 dxdy, \tag{1.48}$$

where μ_{00} gives the area of the ellipse.

Since the ellipse boundary is known from the Eq. 1.46, we write

$$\mu_{20} = \int_{-b}^{b} dy \int_{-a\sqrt{1-\frac{y^2}{b^2}}}^{a\sqrt{1-\frac{y^2}{b^2}}} x^2 dx. \tag{1.49}$$

After integrating, μ_{20} is

$$\mu_{20} = \frac{\pi}{4} a^3 b. \tag{1.50}$$

Similarly, it can be deduced that

$$\mu_{02} = \int\int y^2 dx dy = \frac{\pi}{4} a b^3. \tag{1.51}$$

Knowing this, the length of the minor and major half-axes can now be determined using Eq. 1.45 and 1.47

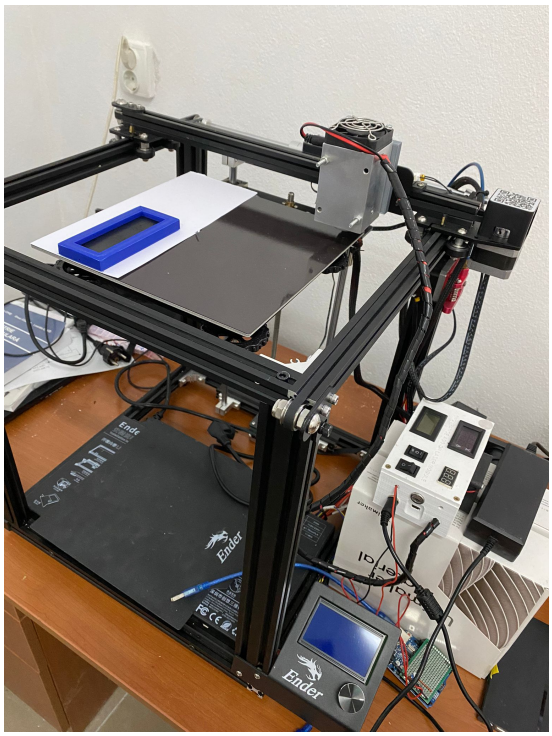
$$a = \sqrt{4\frac{\mu_{20}}{\mu_{00}}} \quad b = \sqrt{4\frac{\mu_{02}}{\mu_{00}}}, \tag{1.52}$$

where a and b are the lengths of the minor and major semi-axes.

2. chapter

The experimental setup

The experimental setup has two main parts: the laser ablation system and the microscope system. These two parts work together.



(a) The laser equipment.
On the top right, the laser head connected to the 3D printer is visible, while on the left side, the black tape fixed to the blue frame can be seen, where the holes are being burned.

(b) The microscope is connected to a laptop, and the sample is digitized through the microscope's camera. Under the microscope objective, the black tape fixed to the blue frame is now marked with holes burned by the laser.

The laser ablation system is shown in Figure 2.1a. It includes a 10-watt diode laser head (manifactured by Endurance [6]), which is connected to a 3D printer (Ender 5-Pro manufactured by Creality [7]). In this setup, the 3D printer it serves to move the laser head across the

sample surface. This movement is controlled by a *G-code* file, which contains the exact positions (X, Y, Z), laser on/off commands, and power settings (see Figure 2.2 for a *G-code* snippet).

```
 1 G1 F2000 X220 Y220 Z0
 2 G1 F2000 X220 Y220 Z47.0
 3 G1 F2000 X48 Y30 Z47.0
 4 G4 P5000
 5 M106 S255
 6 G4 P100
 7 M106 S0
 8 G4 P5000
 9 G1 F2000 X50 Y30 Z47.0
10 G4 P5000
11 M106 S255
12 G4 P100
13 M106 S0
14 G4 P5000
15 G1 F2000 X52 Y30 Z47.0
```

(a)

```
926 G4 P5000
927 G1 F2000 X56 Y90 Z53.0
928 G4 P5000
929 M106 S255
930 G4 P100
931 M106 S0
932 G4 P5000
933 G1 F2000 X58 Y90 Z53.0
934 G4 P5000
935 M106 S255
936 G4 P100
937 M106 S0
938 G4 P100
939 G1 F2000 X0 Y0 Z53.0
940 G1 F2000 X0 Y0 Z0
```

(b)

2.2. Figure. *G-code file for controlling the laser head*
The starting point of the laser is from the corners - this is the instruction in the first two lines - and then the laser head moves horizontally at the same height initially. A pause follows the instruction of the fourth line, then the laser is switched on at maximum power (line 5), and after another pause of 0.1 seconds the laser is switched off (line 7 - M106 S0). This is repeated at different positions until the end of the experiment, when the laser head is switched off and moves to the position $(0, 0, 0)$. Between neighboring positions of the laser head the distance between the laser head and the burn surface is gradually increased.

Writing such a *G-code* file manually would take a lot of time, especially because it requires hundreds of points. So instead, a Python script is used to generate the *G-code* automatically. This script creates the full movement and laser pattern based on a few parameters. This script is included in the Appendix A.

Since the frequency of the PWM signals generated by our 3D printer board is low, it did not allowed for a precise control of the turn-on period of the laser pulse. In order to make the laser pulses more consistent and accurate, an *Arduino UNO* is also connected to the system between the printer board and the laser driver unit. Instead of letting the printer directly control the laser's on-time, the *G-code* commands are used to send a trigger signal to the *Arduino*. When the *Arduino* receives this signal, it switches on the laser for exactly a well-defined period of time, then turns it off again. This gives much better control over the duration of each laser pulse. As a result, the laser burns precise holes at each marked spot, one by one, while the printer moves the head across the sample.

The second major part of the setup is the microscope, shown in Figure 2.1b, which is connected to a laptop. The microscope's built-in camera allows us to see the holes created by the laser and also to save images for further analysis. The microscope was calibrated beforehand,

so it was easy to measure the size and shape of the holes accurately afterward.

Another important part of the system is the blue frame, which shown in Figures 2.1a and 2.1b. This frame was designed using *FreeCAD*, a 3D design software, and it holds the sample firmly in place during the laser process. The design ensures that the sample stays flat and doesn't move, which is very important for accurate laser marking.

3. chapter

The measurement protocol

The experiment had two main stages:

- burning holes with a laser and
- microscope analysis and digitization.

As a first step, I learned how the laser head works and how it moves based on the instructions specified in the *G-code* file. The lines in the *G-code* tell the machine where the laser should go and when it should turn on or off. After understanding this system, I tested how the laser burns holes into different materials, first on a simple piece of paper.

Next, I designed a frame to hold the samples. The purpose of this frame was to provide a flat and stable surface for the burning process, so the holes would be more consistently and reliably produced.

We tested several materials during the experiment: black tape, gold-coated foil, and silver foil. The black tape and the gold-coated foil worked well because the laser made clear and visible marks on their surface. The silver foil caused problems: if the laser was on for a short time, it didn't burn a hole, if it was on too long, the surface below the foil (the 3D printer bed) got too hot and sometimes changed color or the paper underneath even caught fire.

To control the laser precisely, we used an *Arduino UNO*. It turned the laser on for exactly a well-defined period of time when it detected a *G-code* command. The 3D printer's controller can also send these commands, but it's not accurate enough for short pulses.

During the measurements, we made five laser burns at each Z-height (i.e. distance between the laser head and the print bed). Then we changed the Z height and repeated the process five more times. This gave us a structured data set that we could analyze later (Figure 3.1).

After making all the samples, I took photos of each burn hole with a microscope. Before

3.1. Figure. Holes burnt into the black tape fixed to the frame at different Z heights.

taking the images, I calibrated the microscope using a microscope calibration slide. This helped me figure out how many pixels equal one millimeter.

The images were saved in separate folders, organized by Z-height. This made it easy to find and analyze them later. Then I wrote a Python script to process the images. The program loaded the pictures, adjusted brightness, converted them to grayscale, and removed noise. After that, it detected the outlines of the holes.

The program calculated the following properties for each hole: the area (in mm^2), the lengths of the major and minor axes (which show the shape), and the angle of rotation (orientation). It also calculated the average and standard deviation of these values.

4. chapter

Data processing and modeling

The data processing was done in two main steps: first, taking and organizing the microscope images, and second, writing a Python program for image processing. This program helped to measure the shape and size of the holes made by the laser. This chapter explains the full process of data processing in detail.

During the microscope work, I saved the images in a clear folder structure. For each Z-height, I made a separate folder, and saved the pictures from that height inside it. This helped keep the work organized and made it possible to process the images automatically. All pictures were saved in *JPEG* format, using the same naming style, which showed the Z-position of the sample.

Before processing the pictures, I calibrated the microscope. The goal of this step was to find out how big one pixel is *SI* units. To do this, I placed a standard microscope calibration slide under the microscope, with lines at known distances. I measured how many pixels fit into one millimeter. This gave the pixel to millimeter ratio. In the code, I used this value: $pixel\_mm = 1/889.3583$, which means that 889.3583 pixels equal one millimeter.

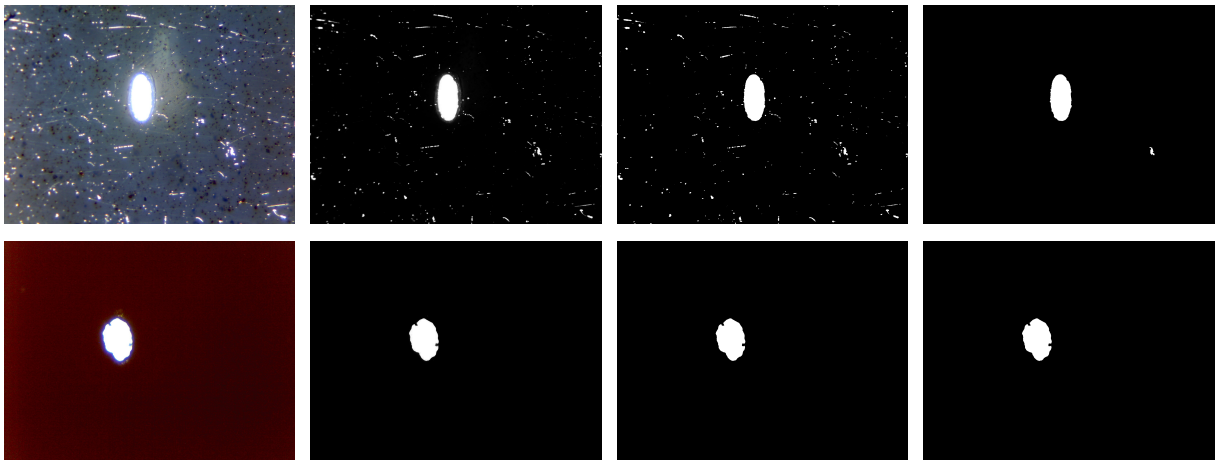
For the image processing, I wrote a Python script using standard libraries (like *OpenCV*, *Scikit-Image*, *NumPy*, and *Matplotlib*).

The program processed the images in the following steps (Figure 4.1):

1. First, the program loaded the images from the correct folders. Then, it applied **gamma correction** to the images. This made the dark edges of the burned holes easier to see.

2. After that, the images were converted to **grayscale**. To make the background smooth and even, I used a flood fill algorithm. This filled the background with one color.

3. In the next step, I used the **otsu threshold method** to make the image binarized.



4.1. Figure. Filterd images from left to right. Top row shows the gold-coated foil and bottom row the black tape.

4. Then, I removed unwanted objects near the edges of the image. I also deleted very small and unimportant objects, like dust or noise. Small holes inside the burned areas were filled. This made sure that the program only found the main shape made by the laser beam.

5. After the image was clean, the program goes through each object and picks the one with the biggest area (meaning the one made up of the most pixels) – this is the hole burned by the laser. To do this, it uses functions like `measure.label()` and `regionprops()` from the `skimage` library [8]. To extract the geometrical properties of this main object, the `regionprops()` function is applied. This function calculates several key descriptors for each labeled region: **area** (the size of the object in pixels), **major** and **minor axis length** (the length of the long and short axes from an ellipse fitted to the object). The program checks also its **orientation angle** (ϕ) (which shows how much the shape is rotated). If the angle is steep (around or above 45°), it's adjusted slightly. Based on the angle, the program decides which side is longer (major axis) and which is shorter (minor axis), and assigns them consistently to the x and y directions. Finally, the measurements (which are originally in pixels) are converted to millimeters, and the final values are stored in lists. The code snippet in Appendix B describes this process.

The program automatically calculated these values for each image. At each Z-height, five measurements were taken. From these results, the program calculated the average and the stan-

dard deviation for each shape property at every Z-position. It saved these results in separate files.

After saving all the data at the given Z-position, these data are loaded into another Python program for further analysis and plotting (see Appendix C).

First, the program fits a quadratic equation to the locations of the maximum and minimum points. Then, using the coefficients of this fit, it calculates the values and Z-axis positions of these maxima and minima. From these, the program calculates other related parameters. The quadratic equation used for the fit is:

$$f(z) = az^2 + bz + c, \tag{4.1}$$

where a, b, and c are the fitted coefficients.
The point where the curve reaches its minimum (or maximum) it corresponds to the beam's focus. This location can be determined by taking the derivative of the fitted curve and setting it to zero:

$$\frac{df}{dz} = 2az + b = 0, \tag{4.2}$$

which gives the focus position z_l as:

$$z_l = -\frac{b}{2a}. \tag{4.3}$$

Next, can be determined the value of the α parameter by numerically solving the following equation:

$$\frac{\frac{w_0^2}{2} ln(\frac{1}{\alpha})}{\frac{w_0^2}{2\alpha e}} = \alpha e ln(\frac{1}{\alpha}), \tag{4.4}$$

where the left side of the equation is the ratio of the minimum and maximum values of the beam width squared (see Figure 1.2).

Using the value of α, and knowing the distance between the two maxima Δz, the Rayleigh

distance (z_R) is calculated by the formula based on Figure 1.2:

$$z_R = \frac{\Delta z}{2\sqrt{\frac{1}{\alpha e} - 1}}. \tag{4.5}$$

The beam waist w_0 is then calculated using the maximum and α by the following formula:

$$w_0 = \sqrt{2\alpha e h_{max}}, \tag{4.6}$$

where h_{max} means the average of the height of two maxima.

Finally, the M^2 parameter, which describes the beam quality, is calculated from z_R and w_0 as:

$$M^2 = \frac{\pi w_0^2}{\lambda z_R}, \tag{4.7}$$

where λ is the wavelength of the laser beam.

However, the shape of a real beam is often not perfectly circular but rather elliptical, meaning that the beam widths in the x and y directions may differ. Therefore, the program must treat the data separately in each direction. For both x and y, the same type of quadratic fit is applied, and the focus position is calculated individually:

$$z_{lx} = -\frac{b_x}{2a_x}, \quad z_{ly} = -\frac{b_y}{2a_y} \tag{4.8}$$

These positions can differ if the beam is astigmatic.

Based on the Figure 1.3 and 1.4, the Rayleigh distances (z_R) in the x and y directions are given by:

$$z_{R,x} = \frac{\Delta z_x}{2\sqrt{\frac{1}{(\alpha e)^2} - 1}}, \quad z_{R,y} = \frac{\Delta z_y}{2\sqrt{\frac{1}{(\alpha e})^2 - 1}}. \tag{4.9}$$

The beam waists in the x and y directions can then be calculated from the respective maxima:

$$W_{0x} = \alpha e\sqrt{2h_{\max_x}}, \quad W_{0y} = \alpha e\sqrt{2h_{\max_y}}, \tag{4.10}$$

where $h_{\max_x}$ and $h_{\max_y}$ are the averages of the two maxima in each direction.

Lastly, the beam quality parameters M_x^2 and M_y^2 for the two directions are:

$$M_x^2 = \frac{\pi W_{0x}^2}{\lambda z_{Rx}}, \quad M_y^2 = \frac{\pi W_{0y}^2}{\lambda z_{Ry}}. \tag{4.11}$$

These values help describe the beam's shape and quality in both directions.

5. chapter

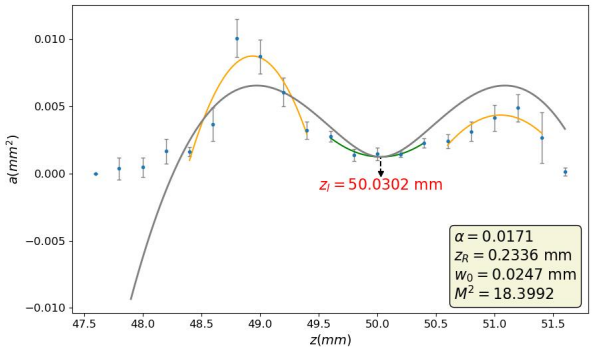
Results

The figures of the present chapter show the results of processing the measurement data recorded during the experiments. They present the area of the laser beam's burned hole, as well as the minor and major axes squared, all plotted as functions of the Z-height. Curves fitted from quadratic equations represent the maximum and minimum of both the burned hole area and the ellipse axes, with local maxima highlighted in orange and local minimum marked in green. From these data, key parameters such as the focus position (z_l), Rayleigh range (z_R), beam waist (w_0), and beam quality factor (M^2) have been calculated based on the procedure outline in chapter 4, and are also indicated on the graphs. The measurements were taken using different laser pulse durations, so the results also clearly demonstrate how the pulse length affects the shape and propagation of the laser beam.

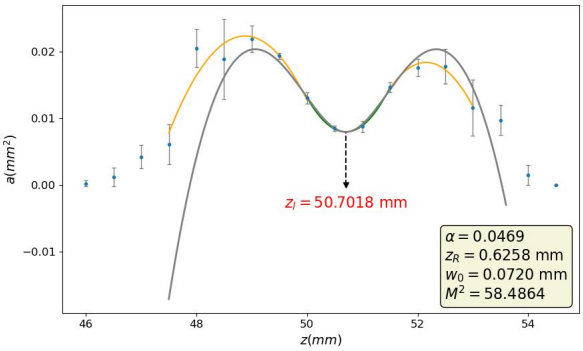
Measurements were carried out on two different materials: black tape and gold-coated foil. The measurements were repeated using different Z-distances and laser-on times to produce comparable results. Overall, both materials showed similar behavior, although there were some small differences.

First (Figures 5.1 to 5.4), the area of the holes burned in the black tape as a function of the Z-distance is shown for increasing laser pulse durations. It can be observed that while the focus position (z_l) consistently remained at $50\ mm$ in all cases, the values of the other parameters fluctuated quite a bit.

The results for the two shortest laser pulse durations of $5\ ms$ and $10\ ms$ (Figure 5.1), showed significantly different outcomes compared to the longer pulse durations. This fluctuation in the values of the determined parameters can be attributed to the fact, that for the shorter laser pulse durations the size of the burn-holes is small, comparable with the size of the error bars.



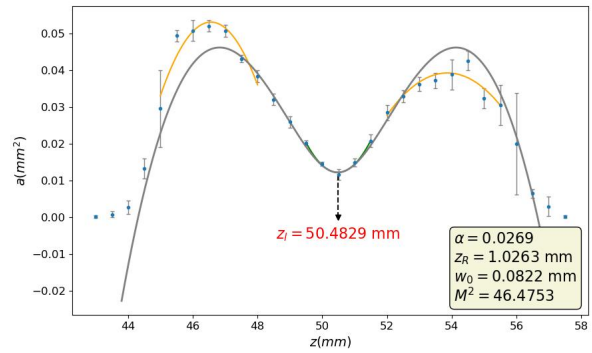
(a) $5ms$



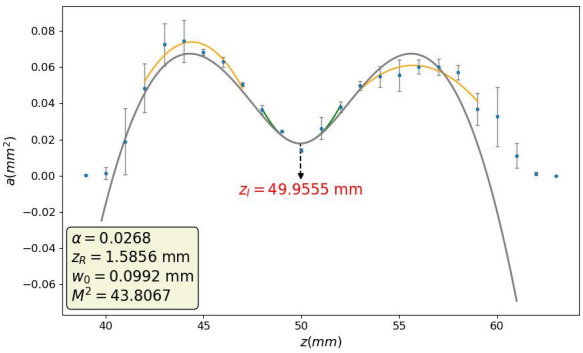
(b) $10ms$

5.1. Figure. Black tape, laser on time: $5\ ms$ and $10\ ms$.

With longer laser pulse durations (Figure 5.2), the results became more consistent; in both cases, the beam width was roughly around $90\ \mu m$, while the M^2 parameter was about 45.



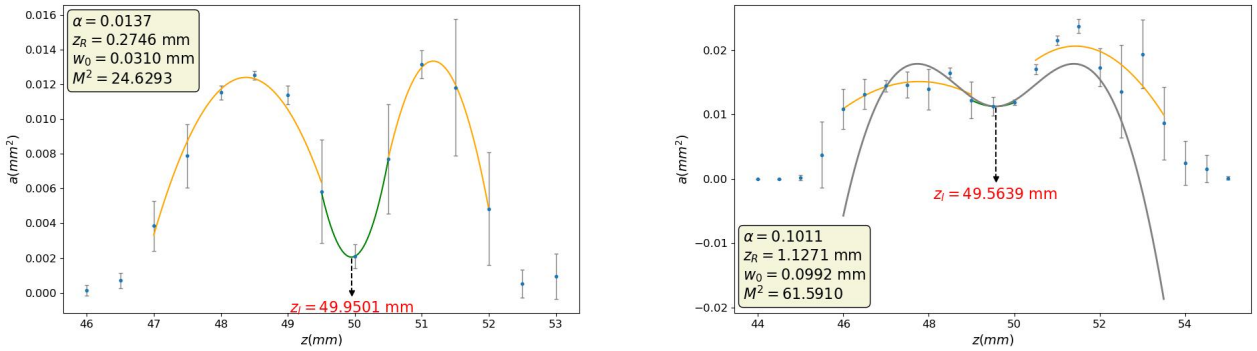
(a) $20ms$



(b) $40ms$

5.2. Figure. Black tape, laser on time: $20\ ms$ and $40\ ms$.

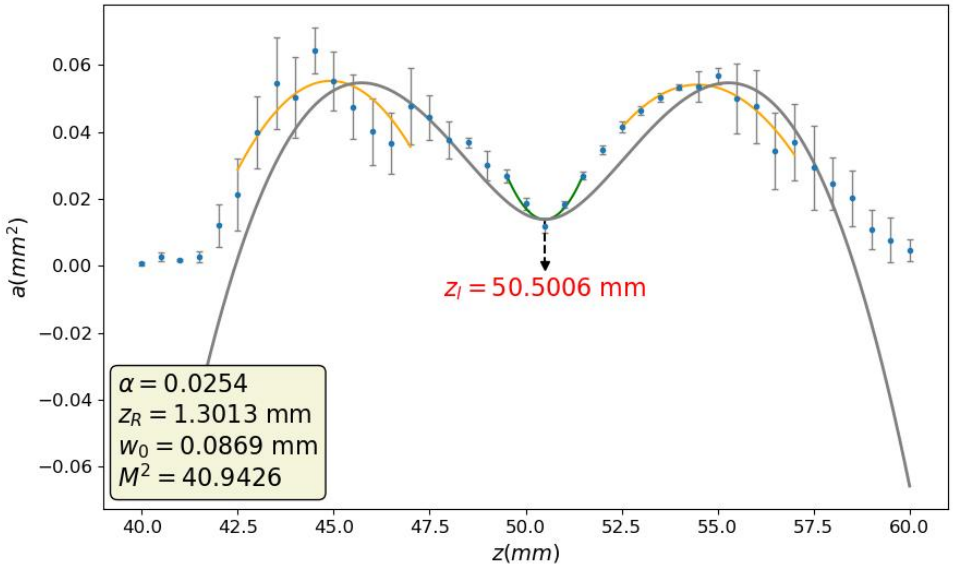
In the case of the gold-coated foil, shorter laser pulse durations again produced quite different parameter values, with the difference in beam widths being nearly $70\ \mu m$.



(a) 300 μs (b) 500 μs

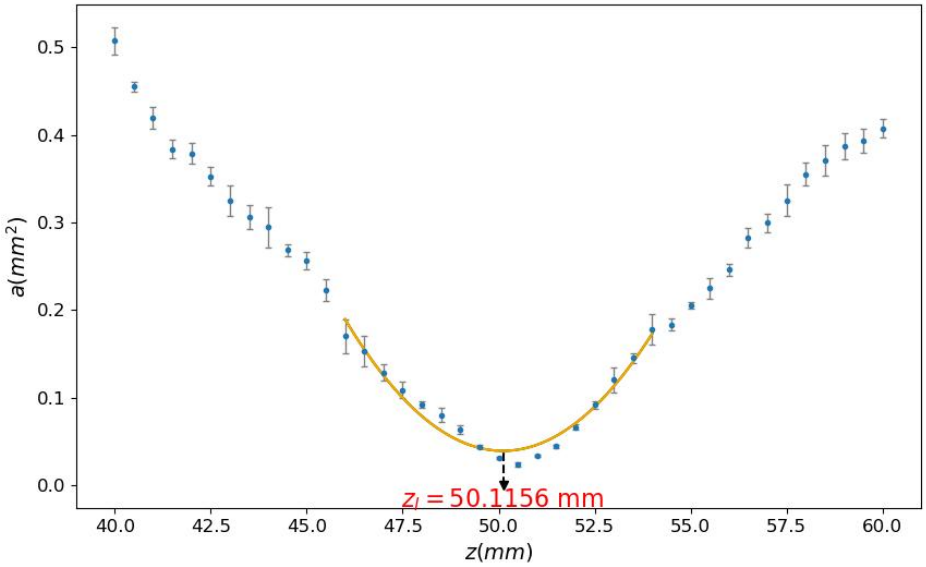
5.3. Figure. Gold-coated foil, laser on time: $300\mu s$ and $500\mu s$.

However, the result for the $1\ ms$ laser pulse duration on the gold-coated foil matched the values obtained for the black tape at longer laser pulse durations.



5.4. Figure. Gold-coated foil, laser on time: $1\ ms$.

In Figure 5.5, the laser pulse duration was even longer, but the selected Z-distance range was too narrow for a proper measurement. As a result, the characteristic double-peak shape of the beam profile did not fully appear. Instead of the expected peaks, only a single minimum was observed.

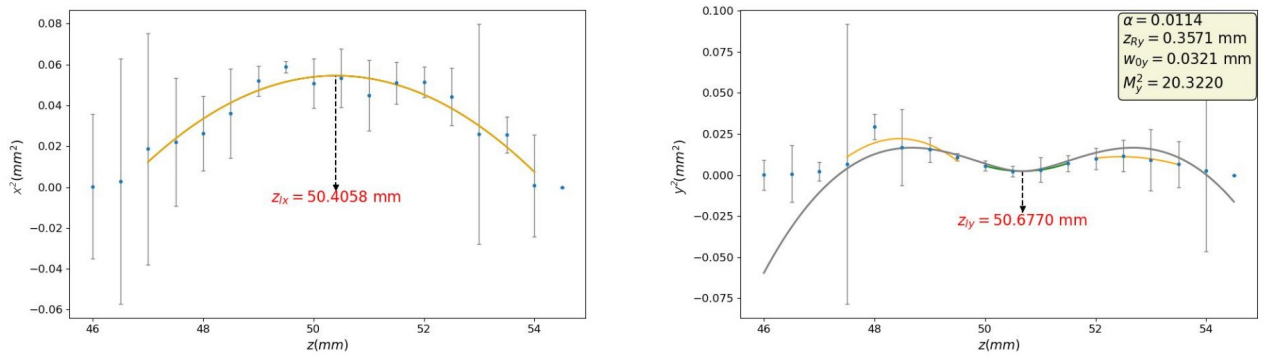


5.5. Figure. Gold-coated foil, laser on time: $10\ ms$.

As part of the further analysis, since the shape of the burned holes can be well approximated by an ellipse, I examined the fitted ellipse's minor and major axes lengths squared as a function of the Z-height. The results of this analysis are illustrated in the following seven figures (from Figure 5.6 to Figure 5.12).

Similar to the previous figures, the important parameters characterizing the laser beam—such as the focus position (z_l), Rayleigh range (z_R), beam waist (w_0), and beam quality factor (M^2)—are indicated on the graphs. Quadratic curves were fitted to the data points, separately for the maximum and minimum values of the minor and major axes. On the curves, the maxima are marked in orange, while the minima are highlighted in green.

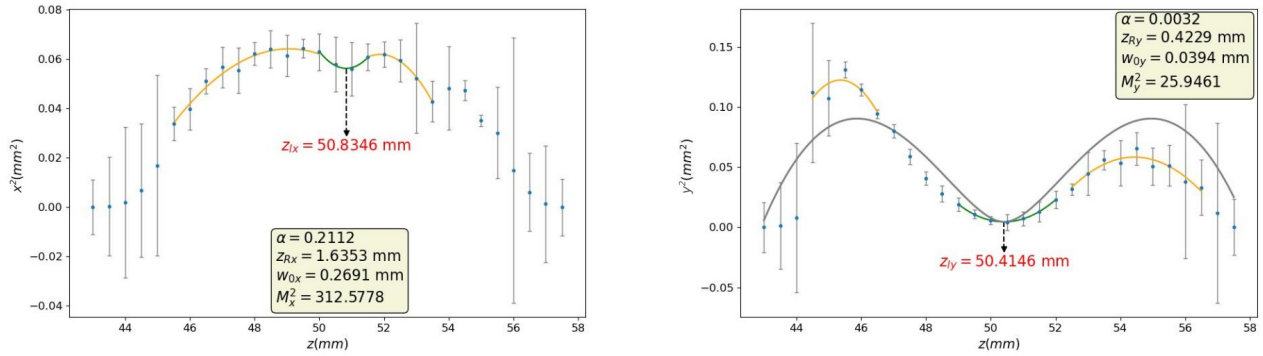
For the black tape, the first figure (with a $10\ ms$ laser-on time, see Figure 5.6) clearly shows that this pulse duration was not enough to create actual holes consistently in the material. Only a single maximum appeared along the minor axis, and the major axis also did not show the typical two maxima with a minimum. The values remained almost constant throughout the entire measurement.



(a) $x^2 - z$ (b) $y^2 - z$

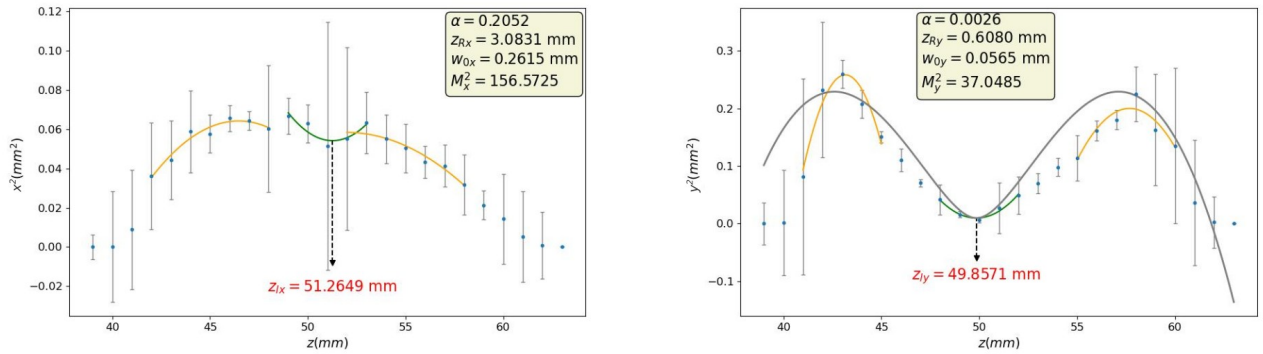
5.6. Figure. Black tape, laser on time: $10\ ms$.

However, when using a longer laser-on time (as seen in Figures 5.7 and 5.8), the characteristic pattern of the laser focus became much more visible. On the minor axis, the two maxima and one minimum barely appeared, however along the major axis the double-peak shape is clearly observable. For the longer pulse duration, the laser pulse parameters along the major axis were consistently determined. However, this not the case along the minor axis, where the large relative error in the minor axis length determination prevented a consistent determination of the pulse parameters.



(a) $x^2 - z$ (b) $y^2 - z$

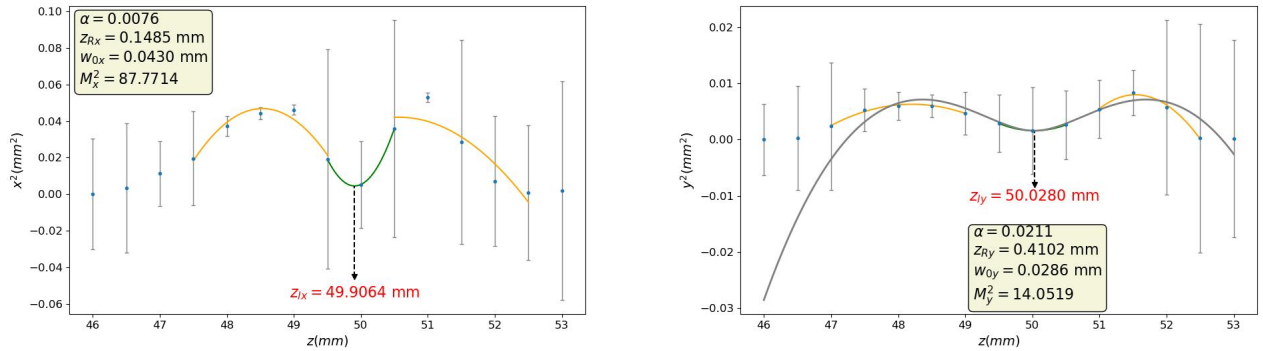
5.7. Figure. Black tape, laser on time: $20\ ms$.



(a) $x^2 - z$ (b) $y^2 - z$

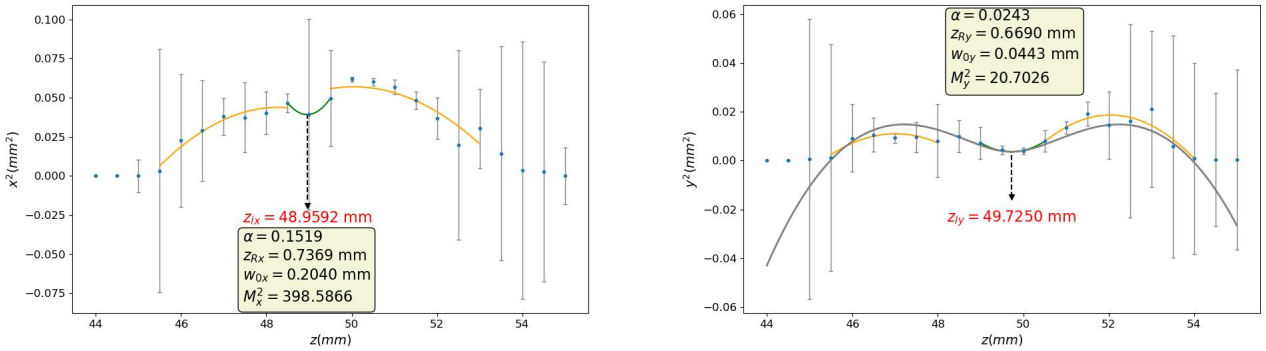
5.8. Figure. Black tape, laser on time: $40\ ms$.

In the case of the gold-coated foil, we observed similar trends. With shorter laser times, the pattern was less visible, while with longer laser-on times, the two maxima and one minimum became much clearer along the major axis. Interestingly, the minor axis sometimes showed unusual behavior, with patterns that differed from the typical shape. Here, along the major axis the determination of the pulse parameters was also consistent for the longer pulse durations. The obtained values for the gold foil are consistent with the ones obtained on the black tape.



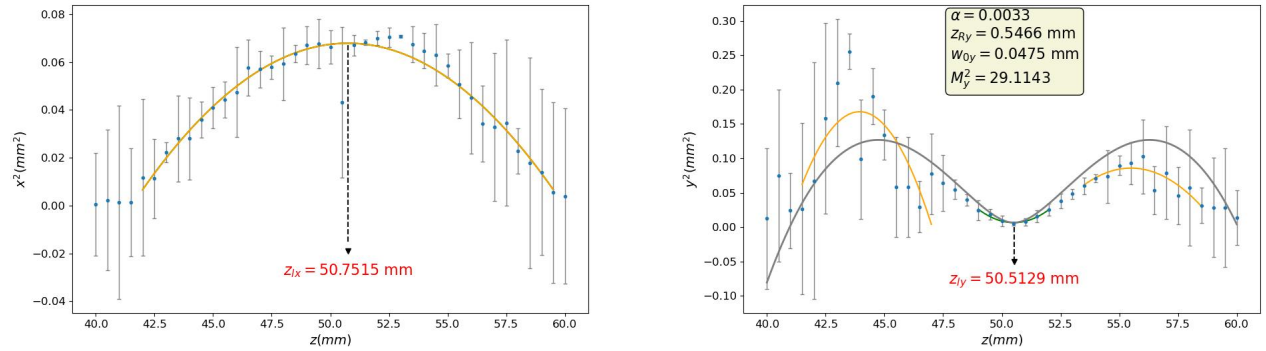
(a) $x^2 - z$ (b) $y^2 - z$

5.9. Figure. Gold-coated foil, laser on time: $300\ \mu s$.



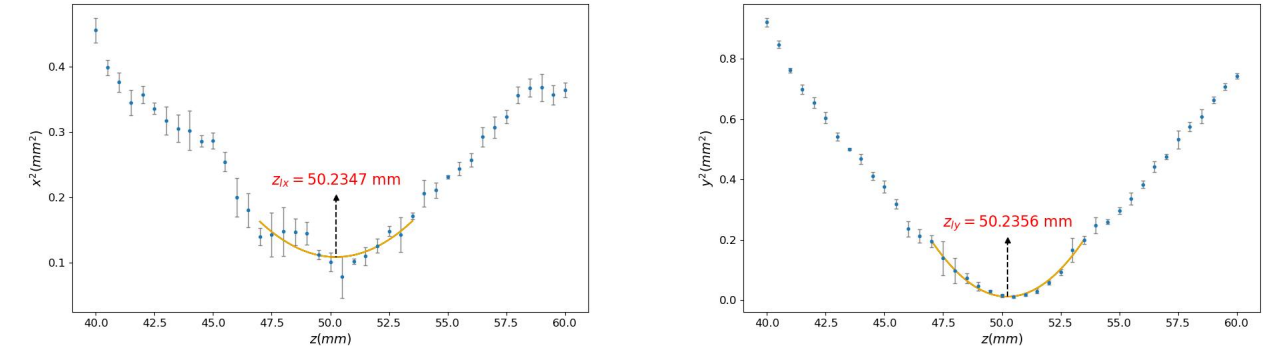
(a) $x^2 - z$ (b) $y^2 - z$

5.10. Figure. Gold-coated foil, laser on time: $500\ \mu s$.



(a) $x^2 - z$ (b) $y^2 - z$

5.11. Figure. Gold-coated foil, laser on time: $1\ ms$.



(a) $x^2 - z$ (b) $y^2 - z$

5.12. Figure. Gold-coated foil, laser on time: $10\ ms$.

Conclusions

The results showed a good agreements with the predictions of our simple model which assumed cylindrically symmetric pulse shape. With longer laser pulse durations, the expected beam width (w_0) of around $90\ \mu m$ and the estimated beam quality factor(M^2) between 40 - 50 could be obtained consistently for both materials.

For asymmetric laser pulse shape, when the shape of the burnt holes was assumed to be elliptical, by separately examining how the minor and major axes of the ellipse vary as a function of Z-height, the theoretical pattern was generally well drawn for the major axis, i.e. one minimum and two maxima, while for the minor axis often only one maximum was visible, the expected minimum did not appear. This suggest that the theoretical does not capture all the features of the experiment and needs to be improved.

It was also observed that with a short laser pulse durations the data became much more scattered and the beam profile was harder to see. This can be attributed to the shot-to-shot instability of the pulses caused by the transient phenomena occurring during the laser switch-on. In contrast, longer laser pulse durations produced much more stable and interpretable results.

Furthermore, I could only change the position along the Z-axis within a relatively narrow range during the measurement. This was due to the fact that the black tape and the gold foil were attached to a fixed frame for stability, but this limited the number of sample points along the Z-axis. At the time, I had not considered that this range might not be sufficient. As a result, the measurements could not cover the variations in the Z direction. This should definitely be improved in the future: a wider measurement range is needed, both downwards and upwards.

As a result of this work it was shown that the simple burn-hole method with some improvements can be a reliable method for determining the parameters laser beams. The method is robust, the measured parameters are not influenced significantly by laser pulse parameters and by the materials in which the holes are burnt.

A. Appendices

G-code generating script

```
from pylab import *

# Generate X coordinates
X = vstack([[arange(50, 60, 2)] * 31]).reshape(155)

# Generate Y coordinates
Y = repeat(arange(30, 92, 2), 5)

# Generate Z coordinates
Z = repeat(arange(47, 53.1, 0.2), 5)

# Open a file to write the G-code
myfile = open("lezer.gcode", "w")

# Move to initial position at speed F2000
myfile.write("G1 F2000 X220 Y220 Z0\n")

# Move to initial laser height (Z[0])
myfile.write("G1 F2000 X220 Y220 Z" + str(Z[0]))

# Move to the first hole burning position
myfile.write("\nG1 F2000 X48 Y30 Z" + str(Z[0]))

# Loop through all points to burn
for i in range(len(Z)):
    # Wait 5 seconds, turn laser on briefly, then off, then wait again
    myfile.write("\nG4 P5000\nM106 S255\nG4 P100\nM106 S0\nG4 P5000\n")
    myfile.write("G1 F2000 ")
    # Move to next position with current X, Y, Z coordinates
    myfile.write("X" + str(X[i]))
    myfile.write(" Y" + str(Y[i]))
    myfile.write(" Z"  %.1f  % Z[i])

# Final laser pulse
myfile.write("\nG4 P5000\nM106 S255\nG4 P100\nM106 S0\nG4 P100\n")

# Move back to origin at current Z height
myfile.write("G1 F2000 X0 Y0 Z" + str(Z[-1]))

# Reset Z axis to 0
myfile.write("\nG1 F2000 X0 Y0 Z0")
```

B. Appendices

Image analysis program

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@author: monika
"""
import os
import math
import glob2
import cv2
import numpy as np
import matplotlib.pyplot as plt
from numpy.linalg import eig
from statistics import stdev
from scipy import optimize
from scipy.optimize import fsolve
from skimage import (
    io,
    exposure,
    segmentation,
    morphology,
    measure,
    img_as_ubyte,
    color,
    filters
)

# PixelToMM
pixel_mm = 1/889.3583

# Read image folders
folders = glob2.glob( /home/monika/Documents/Egyetem_mesteri/
   diplomamunka/mikroszkop/fekete4X_20ms_Z43_Z60/* )
imagenames_list = []


# Initialize lists for results
Z = []
x_avrg = []
y_avrg = []
area_avrg = []
phi_avrg = []

x_dev = []
```

```
y_dev = []
area_dev = []
phi_dev = []

log_a = []
log_y = []
log_x = []
log_phi = []

# Loop through each folder
for folder in sorted(folders):
    isdir = os.path.isdir(folder)
    if isdir == True:
        # Extract Z values from folder name
        z_str = folder.split( / )[-1]
        z = float(z_str.replace("Z", ""))
        Z.append(z)

        # Lists for data
        areas = []
        x_a = []
        y_b = []
        phis = []

        # Loop through each image in folder
        for image in glob2.glob(folder+ /*.jpg ):
            img = cv2.imread(image)
            h, w = img.shape[:2]

            if img is not None:
                imagenames_list.append(img)

            # Apply gamma adjustment
            img0 = exposure.adjust_gamma(img, 6)
            # Convert to grayscale
            gray_img = cv2.cvtColor(img0, cv2.COLOR_BGR2GRAY)
            # Pad borders to enable flood fill
            pad = cv2.copyMakeBorder(gray_img, 1,1,1,1, cv2.
                BORDER_CONSTANT, value=255)
            h, w = pad.shape
            mask = np.zeros([h + 2, w + 2], np.uint8)
            # Perform flood fill to fill background
            img1 = cv2.floodFill(pad, mask, (0,0), 0, (5), (0), flags=8)
                [1]
            img1 = img1[1:h-1, 1:w-1]
            # Threshold to create a binary image
            threshold = threshold_otsu(img1)
            img2 = img1 > threshold
            # Remove borders touching the edges
            img3 = segmentation.clear_border(img2)
            # Remove small objects
            img4 = morphology.remove_small_objects(img3, 500)
            # Fill small holes
            img5 = morphology.remove_small_holes(img4, 500)
            # Determine the principal axes (x, y) and the angle of
                rotation (phi) of the ellipse
            labels = measure.label(img5)
            prop = measure.regionprops(labels, img5)
```

```
            x = 0
            y = 0
            phi = 0
            max_area = 0
            # Loop through components to find the correct minor- and
               major axis
            for i in range(0,labels.max()):
                if max_area < prop[i].area:
                    max_area = prop[i].area
                    phi = prop[i].orientation
                    if phi > 0.785:
                        phi = phi - 0.628
                    if (-0.785 < phi < 0.785):
                        y = prop[i].major_axis_length
                        x = prop[i].minor_axis_length
                    if (phi < -0.785 or phi > 0.785):
                        x = prop[i].major_axis_length
                        y = prop[i].minor_axis_length

            # Store results
            areas.append(max_area*pixel_mm*pixel_mm)
            x_a.append(x*pixel_mm)
            y_b.append(y*pixel_mm)
            phis.append(phi)
            # Log to files
            log_a.append(str(z) +"\t" + str(max_area)+"\n")
            log_y.append(str(z) +"\t" + str(y)+"\n")
            log_x.append(str(z) +"\t" + str(x)+"\n")
            log_phi.append(str(z) +"\t" + str(phi)+"\n")

        # Calculate average and standard deviation
        x_avrg.append(np.mean(x_a))
        x_dev.append(stdev(x_a))
        y_avrg.append(np.mean(y_b))
        y_dev.append(stdev(y_b))
        area_avrg.append(np.mean(areas))
        area_dev.append(stdev(areas))
        phi_avrg.append(np.mean(phis))
        phi_dev.append(stdev(phis))


# Save results to files
np.savetxt("areas.txt", log_a, fmt="%s")
np.savetxt("y.txt", log_y, fmt="%s")
np.savetxt("x.txt", log_x, fmt="%s")
np.savetxt("phi.txt", log_phi, fmt="%s")

np.savetxt("area_avrg.txt", np.transpose([Z, area_avrg,area_dev]), fmt="
    %s", delimiter= \t )
np.savetxt("y_avrg.txt", np.transpose([Z, y_avrg, y_dev]), fmt="%s",
    delimiter= \t )
np.savetxt("x_avrg.txt", np.transpose([Z, x_avrg, x_dev]), fmt="%s",
    delimiter= \t )
np.savetxt("phi_avrg.txt", np.transpose([Z, phi_avrg, phi_dev]), fmt="%s
    ", delimiter= \t )
```

C. Appendices

Plot results and calculate parameters

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@author: monika
"""

# Import required libraries
import numpy as np
import matplotlib.pyplot as plt
from numpy.polynomial.polynomial import Polynomial
from scipy.optimize import fsolve
import math

# Wavelength in mm (445 nm)
lam = 445e-6

# Load measurement data: z-position, average area, and standard deviation
z, avrg, dev = np.loadtxt( /home/monika/Documents/Egyetem_mesteri/diplomamunka/program/results/folia_10ms_Z40_Z60/area_avrg.txt ).T

# Create a figure and plot the data with error bars (standard deviation)
plt.figure(figsize=(10, 6))
plt.errorbar(z, avrg, yerr=dev, fmt= . , ecolor= gray , elinewidth=1, capsize=2)
plt.xlabel("$z(mm)$", fontsize=14)
plt.ylabel("$x^2 (mm^2)$", fontsize=14)
plt.tick_params(axis= both , labelsize=12)

# Define index intervals where maxima and minima are located
intervals = [
    (4, 10),   # first maximum
    (14, 19),  # minimum
    (21, 28)   # second maximum
]

# First maximum
# Fit a 2nd-degree polynomial to the first interval
a, b, c = Polynomial.fit(z[intervals[0][0]:intervals[0][1]], avrg[intervals[0][0]:intervals[0][1]], 2).convert().coef
max1_z = -b / (2 * c)  # Z-position of the first maximum
max1_avrg = a + b * max1_z + c * max1_z**2  # Value of the first maximum
```

```
z_fit = np.linspace(z[intervals[0][0]:intervals[0][1]].min(), z[
    intervals[0][0]:intervals[0][1]].max(), 100)
avrg_fit = a + b * z_fit + c * z_fit**2  # Fit curve
plt.plot(z_fit, avrg_fit, color= orange )  # Plot the first maximum
    curve

# Minimum
# Fit a 2nd-degree polynomial to the second interval
a, b, c = Polynomial.fit(z[intervals[1][0]:intervals[1][1]], avrg[
    intervals[1][0]:intervals[1][1]], 2).convert().coef
min_z = -b / (2 * c)  # Z-position of the minimum
min_avrg = a + b * min_z + c * min_z**2  # Minimum value
z_fit = np.linspace(z[intervals[1][0]:intervals[1][1]].min(), z[
    intervals[1][0]:intervals[1][1]].max(), 100)
avrg_fit = a + b * z_fit + c * z_fit**2  # Fit curve
plt.plot(z_fit, avrg_fit, color= green )  # Plot the minimum curve

# Second maximum
# Fit a 2nd-degree polynomial to the third interval
a, b, c = Polynomial.fit(z[intervals[2][0]:intervals[2][1]], avrg[
    intervals[2][0]:intervals[2][1]], 2).convert().coef
max2_z = -b / (2 * c)  # Z-position of the second maximum
max2_avrg = a + b * max2_z + c * max2_z**2  # Value of the second
    maximum
z_fit = np.linspace(z[intervals[2][0]:intervals[2][1]].min(), z[
    intervals[2][0]:intervals[2][1]].max(), 100)
avrg_fit = a + b * z_fit + c * z_fit**2
plt.plot(z_fit, avrg_fit, color= orange )  # Plot the second maximum
    curve

# Print maximum and minimum values
print(f"max1_avrg = {max1_avrg:.6f}")
print(f"max2_avrg = {max2_avrg:.6f}")
max_avrg = (max1_avrg + max2_avrg) / 2  # Average of the two maxima
print(f"max_avrg = {max_avrg:.6f}")
print(f"min_avrg = {min_avrg:.6f}")
print(f"min_z = {min_z:.6f}")

# Define the equation used to solve for the alpha parameter
def equation(alpha):
    if alpha <= 0:
        return 1e6  # Avoid invalid log or division
    return alpha * math.e * math.log(1 / alpha) - (min_avrg / max_avrg)

# Solve the equation numerically to find alpha
alpha = fsolve(equation, 0.01)[0]
print(f"alpha = {alpha:.6f}")

# Calculate the Rayleigh range (zR)
val = 1 / (alpha * math.e) - 1
if val > 0:
    zR = (max2_z - max1_z) / (2 * math.sqrt(val))
else:
    print(f"{alpha:.6f}     sqrt argument negative ({val:.6f})")
    zR = float( nan )
print(f"zR = {zR:.6f}")

# Calculate the beam waist (w0)
```

```
w0 = math.sqrt(2 * alpha * math.e * max_avrg)
print(f"w0 = {w0:.6f}")

# Calculate the beam quality factor (M^2)
M2 = math.pi * w0**2 / (lam * zR)
print(f"M^2 = {M2:.6f}")

# Mark the focus point (minimum) on the graph
plt.vlines(min_z, ymin=+0.2, ymax=min_avrg, color= black , linestyle= --
    , linewidth=1.5)
plt.plot(min_z, +0.2, marker= ^ , color= black )
plt.text(min_z, +2.2*min_avrg, f"$z_{{l}} = {min_z:.4f}$ mm", color= red
    ,
         fontsize=16, ha= center , va= top )

# Add a text box with calculated parameters
text =  \n .join((
    f"$α = {alpha:.4f}$",
    f"$z_{{R}} = {zR:.4f}$ mm",
    f"$w_{{0}} = {w0:.4f}$ mm",
    f"$M_^2 = {M2:.4f}$"))
props = dict(boxstyle= round , facecolor= beige )
plt.text(0.4, 0.99, text, transform=plt.gca().transAxes, fontsize=16,
         verticalalignment= top , bbox=props)

# Define the theoretical beam shape equation
def theoretical(z, w0, z0, zR, alpha):
    return - (w0**2 / 2) * (1 + ((z - z0) / zR)**2) * np.log(alpha * (1
       + ((z - z0) / zR)**2))

# Plot the theoretical curve
z_theoretical = np.linspace(z.min(), z.max(), 500)
avrg_theoretical = theoretical(z_theoretical, w0, min_z, zR, alpha)
plt.plot(z_theoretical, avrg_theoretical,  - , color= gray , linewidth
   =2)

# Save the figure
plt.savefig("/home/monika/Documents/Egyetem_mesteri/diplomamunka/program
   /results/folia_10ms_Z40_Z60/area_avrg.jpg")
plt.show()
```

Bibliography

[1] *Beam Diagnostics,*
https://www.photonics.com/Articles/Beam_Diagnostics_Meeting_the_Need_for_High/a25162,
Accessed: June 12, 2025

[2] *Laser Beam Profile Measurement,*
https://www.newport.com/n/laser-beam-profile-measurement,
Accessed: June 12, 2025

[3] Duarte Estrada, *Optical beam propagation in nonlinear media*, Instituto Superior Técnico, Av. Rovisco Pais, Portugal

[4] Anthony E. Siegman, *Lasers*, University Science Books, Mill Valley, California (1986) 626-695

[5] Simon Xinmeng Liao, *Image Analysis by Moments* (1993)

[6] *Endurance,*
https://endurancelasers.com/diode-lasers/10w-endurance-laser/,
Accessed: June 10, 2025

[7] *Ender 5-Pro manufactured by Creality,*
https://www.creality.com/products/ender-5-pro-3d-printer,
Accessed: June 10, 2025

[8] *Image processing in Python,*
https://scikit-image.org/,
Accessed: June 18, 2025