Deep Learning Optimization for High Power Lasers Aberrations Correction

Fernandes, G.G.D.[1]; Alexandrino, D.[1]; Silva, E.[1]; Matias, J.[1]; Pereira, J.[1]
[1]Instituto Superior Técnico

Abstract

High harmonic generation (HHG) proves itself as a useful non-linear process that permits table-top generation of tunable high energy coherent very short radiation pulses, in the EUV to soft X-ray range, which can then be given multiple applications, like photoemission spectroscopy in condensed matter physics [1], pump probe spectroscopy for high energy density plasmas, etc. Typically one has to deal with optical aberrations resulting from the system used to manipulate the high-power laser necessary for HHG. Through the use of an SLM and several different machine learning algorithms, including a neural network, we continued previous work to optimize for these aberrations and improve the current setup being used for HHG.

1 Introduction

HHG is a non-linear process where the interaction between a high intensity laser pulse and a certain material generates harmonics of the laser of a high order (above the fifth) or, in other words, pulses with a frequency n times higher than that of the initial pulse.

One can use a simple model to explain HHG, known as the **three-step model** as described in [2]: an electron of the material tunnels out from the atomic potential which has been twisted by the intense field from the laser; the now free electron is accelerated away from the atom due to the driving field; after half a period of the laser, the electric field inverts and accelerates the electron back into the atomic potential that will then emit a high energy photon. The non-linearity of the process means that usually, the duration of the light pulse produced by this process is actually shorter than the pulse of light that hit the material initially. This is what allows HHG to be used for **attosecond physics**, where a very high time resolution is necessary.

2 Motivation

As the HHG involves the use of a high powered laser, it is a necessary part of using this process for anything to deal with an optical apparatus that manipulates this laser, and eventually focuses it down to a small point where the target gas, in which the actual HHG will occur, is located. Consequently, we'll have to deal with the associated optical aberrations that will arise in such an apparatus, that reduce the quality of the focus we need, and therefore decrease the efficiency of our HHG. These aberrations can be modeled using the Zernike polynomials. A zernike polynomial of a certain order determines a specific type of aberration and its coefficients determine its shape [3]. They can then be used for beam optimization.

For this, we utilize an **SLM** (spatial light modulator) as a controllable parameter. This device that can be characterized as a form of diffraction grating, acting like a ”programmable lens”, composed of 1000x1000 pixels, where in each pixel, through the use of a computer, can be applied a voltage that changes its refractive index. With the right voltage values the SLM can essentially ”flatten” an aberrated wavefront.

The crux of this experiment consisted in the development, testing and usage of several machine learning based methods that optimize the SLM voltages for the best focus of the laser.

3 Experimental Apparatus

An initial simple setup was used for the purposes of developing and testing the optimization algorithms composed of a set of different lenses, mirrors, the SLM, a laser and a camera. The laser paths through several lenses to induce aberrations, then goes through the SLM to be "corrected" and is finally focused on a camera.

Then, the main setup, was the one previously used for HHG, with the SLM implented. A high-power infrared pulse laser was utilized, and pumped into a vaccum chamber, where it would focus on a gas cell, hence creating the high order harmonics. At the end of the vacuum chamber there was an advanced UV Greateyes camera that was used for detection and visualization of the harmonics and would eventually provide data for the machine learning algorithms developed in the earlier setup to work with. The optical system was composed of several lenses, mirrors, other optical components, and, of course, the SLM. Of note are the attenuator that allowed control of the laser intensity; several irises that allowed for alignment and also some degree of control over the intensity, as well as a selection of the correct order beam[1]; a beam splitter that would split the laser onto an adjacent path into a Thorlabs amera that would gather data for an early form of the optimization algorithms that did not use data from the Greateyes UV camera yet. A diagram of the setup can be seen below, 1.

The implementation of the SLM into the HHG system was initially conducted with some attempts of manual optimization with data from the Thorvision camera, changing directly the aberration coefficients in the SLM settings.

Once HHG was initiated, the oblique astigmatism coefficient in the SLM was swiped from 0.03 to 0.24, with 0.03 increments. Then, the data, collected from the Greateyes camera, was analyzed to provide the following graphs.

As shown in figure 2, it is possible to manually find the coefficients that have the greatest effect on the to-

[1]The diffraction grating like behavior of the SLM causes it to have multiple orders of diffracted beams, (zeroth-order not being diffracted by the SLM at all, etc.) we had then to make sure we only had the first-order diffracted beam going into the vacuum chamber, blocking the rest with an iris.

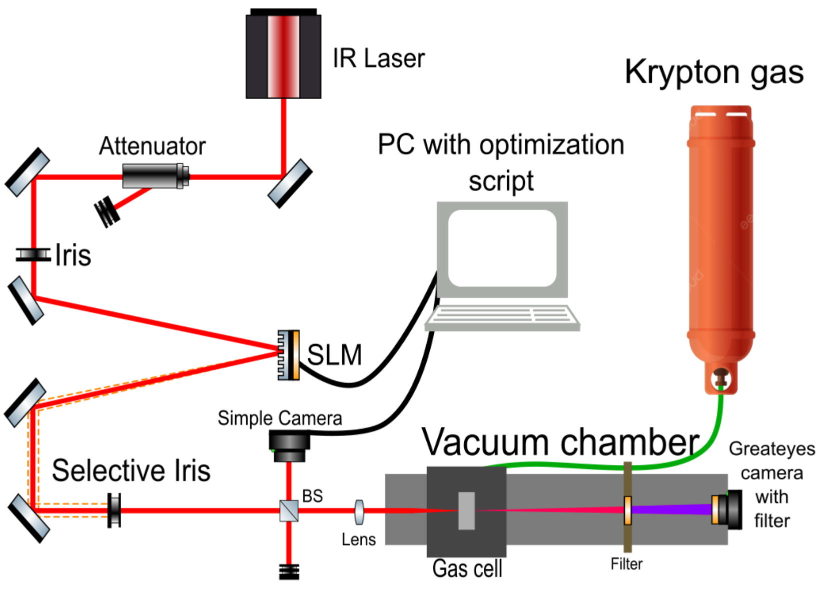


Figure 1: Final setup for HHG with SLM. As visible, the gas used in the cell is krypton and filters were used to remove all wavelengths from the beam post-HHG (represented in pink), except for the ones corresponding to the high order harmonics in the XUV range (represented in purple). The dashed orange lines represent the different unwanted orders of the diffracted beam coming from the SLM, that then get blocked by the iris.

tal signal. However, when dealing with higher dimensionality problems (there are many more parameters to optimize besides oblique astigmatism), this method is not optimal. Thus, to make this process faster, it is important to consider Machine Learning approaches, some of which will be detailed in sections 4 and 6.

4 Bayesian Optimization

4.1 The bayesopt Function

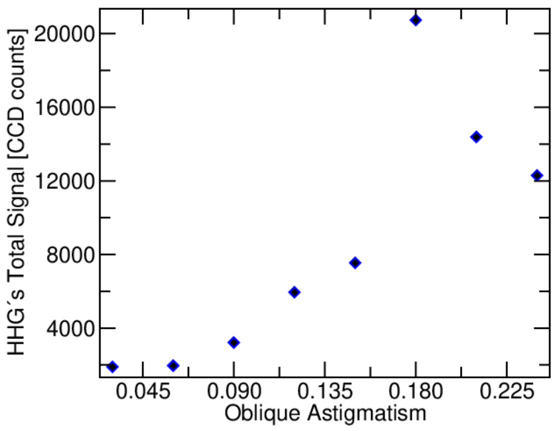
MATLAB´s `bayesopt` function selects optimal machine learning hyperparameters, encapsulating the entire Bayesian optimization workflow. Moreover, `bayesopt(fun,vars)` attempts to find the values of `vars` (representing a variable) that minimize `fun(vars)` (corresponding to a function of that variable). Although its main purpose is to minimize the function, it is possible to use it as a maximization tool.

4.2 Optimizing The Peak Intensity

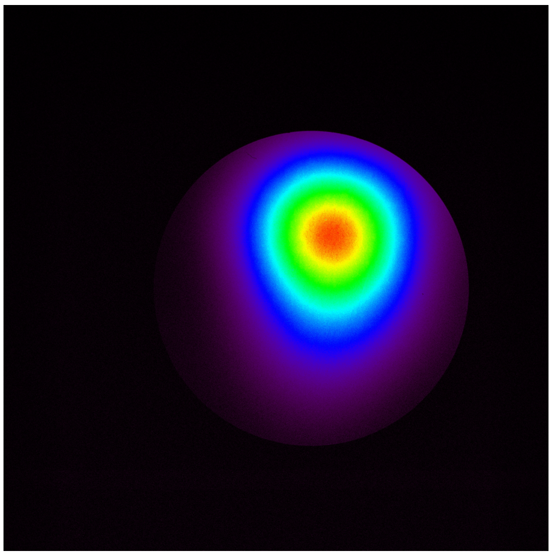
A previously written script was studied and adapted by the group, using this feature to optimize the performance of a system that includes a camera, a spatial light modulator (SLM), and a Zernike polynomial holographic correction. The main goal is to optimize the quality of a focal spot by adjusting the amplitudes of different Zernike modes. Specifically, we aimed to optimize five orders of the Zernike polynomials corresponding to the focus, vertical and oblique astigmatism, and vertical and horizontal coma.

As a first step, the code loads the TL-Camera (Thorlabs) and configures it regarding gain and exposure time. Additionally, the camera's bit depth is taken into account to determine the maximum pixel value (`maxPixelValue`).

Since `bayesopt` and the whole Bayesian optimization process are iterative, in order to achieve better re-



(a) HHG´s Total Signal vs Oblique Astigmatism



(b) Map of Intensities (0.18)

Figure 2

sults, we used loops. To optimize the focus, we start by defining a function called `foc_max_count` (which will give us the peak intensity detected in an image taken by the camera) and the variable `Zernikamplitudes_foc`, containing values within a given range (normally small values near zero, for example, $[-0.6, -0.2]$), using MATLAB´s function `optimizableVariable`. After this, `bayesopt` runs 30 times, corresponding to 30 different values contained in the previously mentioned range, to find which one of those values leads to a higher value of the `foc_max_count` function. The optimal value is then saved into a new variable called `opt_focus`. The same method is applied to the four other aberrations.

Additionally, we created a new and corrected range of values for each amplitude using the `opt_i` variables (i = focus, comma1, etc). A new function, `final_max_count`, is also built to return the peak intensity detected in an image using all five amplitudes.

Finally, `bayesopt` runs 30 times again so that it returns the best combination of the five Zernike amplitudes (testing 30 different arrangements of the values within the corrected ranges).

5 Fourier Transform and Harmonics Wavelenghts

5.1 New Camera

In addition, the group worked on the adaptation of the script mentioned in 4.2 to a different camera that was inserted into the vacuum chamber. After analyzing the available documentation (Greateyes instruction manual) and testing extensively, we finally succeeded. Through the use of this new code, we were able to perform a similar data acquisition process, enabling not only better image quality and definition but also the opportunity to analyze data obtained from the vacuum chamber.

In this section, we take a closer look into the frequencies and wavelength of the harmonics used in the laser system. After registering the image of the laser with the ALEX-i camera (associated with the vacuum chamber) using the new script, we ran two different algorithms to get: firstly, the Fourier transform of the input image; and then, using that transform as the new input, we used the Python Library `Matplotlib` to measure the distance between the peaks of the transform. In the end, we should be able to know through the algorithms what wavelengths are present in the harmonics.

5.2 Applying the Fourier Transform

Initially, using the ALEX-i camera, we were able to acquire a large dataset of many different images from the harmonic system, resembling the following case: Looking closely, it is possible to distinguish at least two different frequencies on the harmonic (seen by the different spaces between the vertical lines). When we have an interference pattern from several wavelengths, we want to measure the distance between the fringes, given by $w_i = z * \frac{\lambda_i}{d}$. Here, $z \sim 3m$ is the distance between the source and the camera, and $d \sim 75\mu m$ is the distance between both HHG sources. Therefore each value of λ_i will create fringes with separation w_i. However, it is hard to extract from the image above what are these separations, thus we do Fourier analysis (2D spatial Fourier transform), to get the result in figure 3.

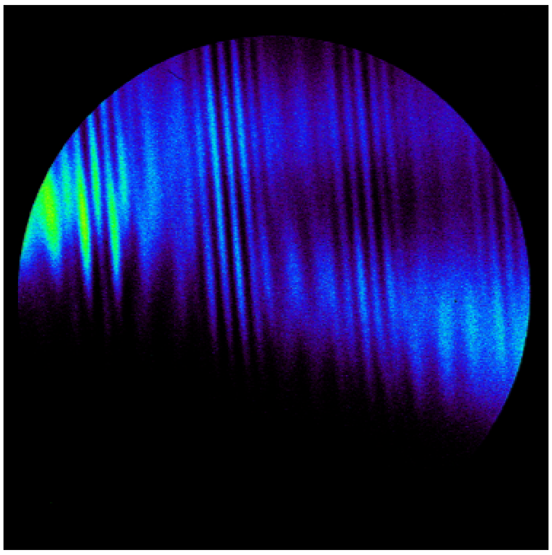
When we apply the Fourier transform, we clearly observe peaks corresponding to the spatial frequency of $k_i = 2 * \frac{\pi}{w_i}$. This way, it is possible to measure the intensity of each peak inside the red box on the image above and get to know the distance between the peaks. To get a graph that enables the measuring of k_i, we used another algorithm to look at the intensity value of every pixel along a line (that best suits the interference pattern on the 2D Fourier Transform), such as in figure 4, giving us the following plot:

Using the graph above, we can measure directly the values of k_i, and know the different wavelengths that produced the figure 3. To get the value of k_i we look at peaks pixel indexes and using the ALEX-i camera specifications (knowing that the distance of each pixel is 13.5 μm), it is possible to get the correspondent value of the spatial frequency. Using the dataset we had, we got the following table:

Image	1			2				3			
Peak Pixel Index	166	184	202	182	187	190	195	168	175	180	186
k_i	13165	10416	7813	10706	9982	9548	8825	12731	11718	10995	10127
w_i [mm]	0.477	0.603	0.804	0.587	0.629	0.658	0.712	0.493	0.536	0.571	0.621
λ [nm]	11.93	15.08	20.10	14.68	15.73	16.45	17.8	12.33	13.4	14.28	15.53

Table 1: Fourier Transform Analysis.

Looking at the table above, we get conflicting results, since the minimum distance between lines is $w_{min} \sim 0.5mm$ (what we expected), but the evolution of w_i and λ_i isn't constant for all cases. Looking at the values of both the space between the fringes and the wavelength of the harmonics, its possible to understand that those values don't follow the criteria of an harmonic. In conclusion, the algorithm implemented is great at discovering the values for the w_{min} and k_{max}, however due to chaos and noise form the signal, it is very hard to decide which peaks should be consider, creating a bad evolution and determination the λ values.


(a) Harmonics Signal using ALEX-i Camera.

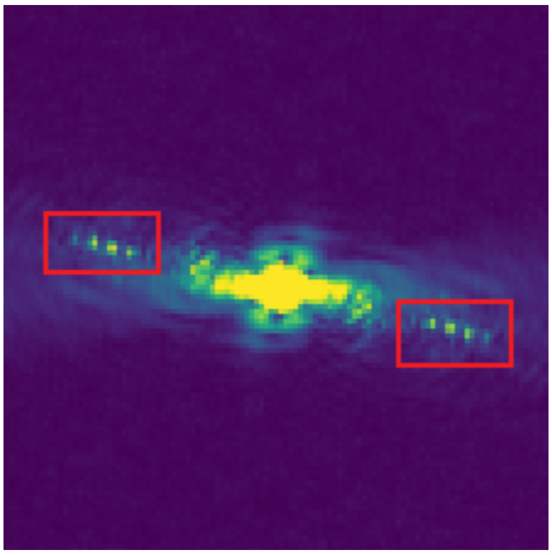

(b) Fourier Transform.

Figure 3

6 Neural Network

6.1 Motivation

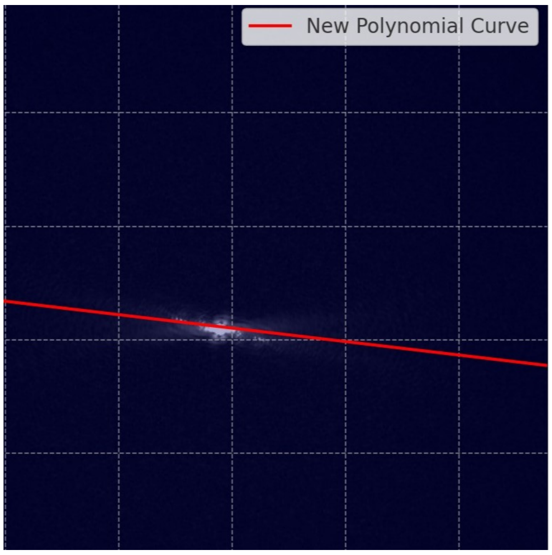
The Bayesian Optimization is a simple approach that only considers one objective function, the maximum pixel intensity. First, we tried to also impose roundness by applying a modified Gaussian filter that assures the total intensity is constant. However, there is information loss, and getting a better filtered image does not mean the SLM is correcting the real aberrations. This way, only a filter of sigma less than three would be helpful to filter noise and reduce sharp pixel intensities. A more versatile and powerful solution that we explored is deep neural networks.

Out of all the possible choices, we chose to use a Convolutional Neural Network (CNN). This is a powerful deep learning method for image processing that recognizes spatial patterns with translation invariance, and extracting meaningful features. One paper used the CNN to learn the mapping relationship between the object intensity distribution and its aberration phase. Then, the image was corrected by summing the conjugate of the wavefront aberration. However, to reconstruct the aberration phase we would need, for example, a Shack Hartman Sensor or to train on simulation data, which would wield problems with generalization [4].

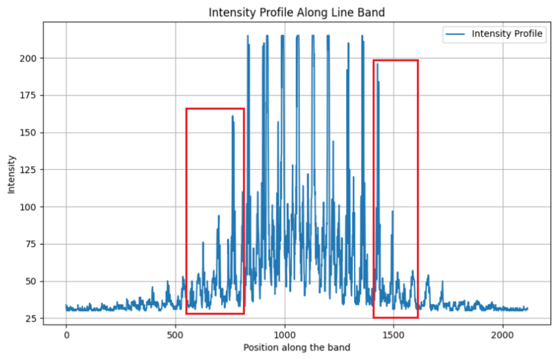
So, we sought to use a CNN to learn the mapping relationship of the Zernike coefficients and the point spread function. Then, with the symetric of the coefficients and the SLM we can correct the images. There is a paper where this was done [5]. For this they used for correction the 4th - 15th order polynomials. The reference of the polynomial order follows the University of Arizona Index presented in the attachment 8.

6.2 Data Set Script

In the script for obtaining the training images we introduce aberrations in our im-



(a) Second Algorithm used in the Fourier Space.



(b) Intensity of Pixels Along the Polynomial Curve.

Figure 4

ages by randomly generating 4th, 5th and 6th order Zernike polynomials that the SLM will use to modify the beam. Here again the order of the zernike polynomial is considering the University of Arizona Index. We did not consider higher order modes for simplicity. Furthermore, the piston, tip and tilt, which correspond to the 1st, 2nd and 3rd order polynomials were not considered. This is because they are not actually true optical aberrations, as they do not represent or model curvature in the wavefront.

Then, a key step is ensuring only one of the two duplicated spots of the laser image is chosen and that the images are cropped. For this we created a script that follows the steps: 1) Normalize the image to 8-bit range, 2) Threshold the image to get a binary image, 3) Find contours in the thresholded image, 4) Find the largest contour based on the area 5) Crop the image to the standard bounding rectangle which we found to be the best. For the neural network we need to assure a constant image size.

With this pre-processing we can have substantially more images for our data set. In some cases, the two spots are too close and the image might capture them both so they must be excluded.

With this script, we obtained 506 pairs of .pngs and .csvs that composed our data set. It is important to note that our data set is considerably smaller than the ones used in the papers mentioned which is a great limitation. For example, in the second paper the training data is composed of 18 thousand images 128x128. The images were acquired for the HHG generation set up and the camera used was from Thor Labs.

6.3 Neural Network Structure

The architecture of a neural network is crucial for the model's ability to extract and comprehend intrinsic patterns in the data. In this project, a specific architecture has been carefully outlined for the task of predicting Zernike coefficients. The following section highlights the es-

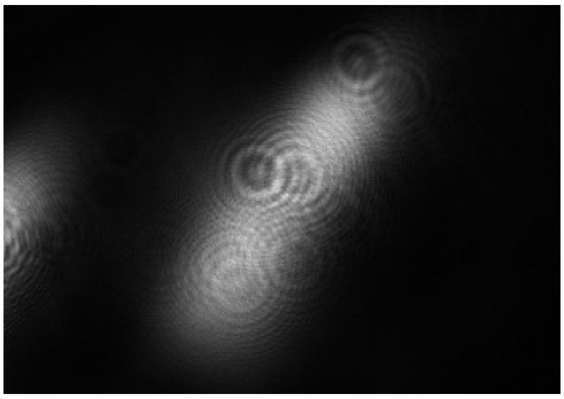


Figure 5: Example of training image generated

sential details of this neural architecture.

The adopted neural network model follows a sequential approach and incorporates convolutional, pooling, and dense layers. The detailed configuration of the network is presented thoroughly in Figure 9:

1) Convolutional and Pooling Layers:

- The first convolutional layer has 32 filters of size $(3,3)$ and uses the ReLU activation function.
- Next, a pooling layer (`MaxPooling2D`) with size $(2,2)$ is applied for dimensionality reduction.
- The second convolutional layer uses 64 filters of size $(3,3)$ and employs the ReLU activation function.
- A second pooling layer is then implemented.
- Finally, the third convolutional layer again has 64 filters of size $(3,3)$ with ReLU activation.

2) Dense Layers:

- The output of the convolutional layers is flattened (Flatten) to be fed into the dense layers.
- In addition, a dense layer with 64 neurons and ReLU activation is incorporated into the model.

3) Output Layer:

- The output layer consists of 3 neurons, corresponding to the Zernike coefficients to be predicted.
- Since the task is of a regression nature, the activation function in the output layer is linear.

This architecture was meticulously designed to capture patterns present in the images associated with Zernike coefficients, providing an internal representation capable of generating accurate predictions.

In the next section, the selected hyperparameters for training the network will be detailed, offering a comprehensive perspective on the model optimization and tuning process.

6.4 Hyperparameters and Training

In addition to the network architecture, hyperparameters play a crucial role in the effectiveness and performance of the model. The training process is governed by these parameters, influencing everything from model optimization to generalization for unseen datasets.

6.4.1 Model Compilation

The process begins with the compilation of the model, where the optimizer, loss function, and evaluation metrics are specified. In this case, the chosen optimizer is `adam`, widely used for its overall effectiveness. The adopted loss function is `mean_squared_error`, suitable for regression problems like predicting Zernike coefficients. Additionally, metrics such as mean absolute error (`mae`) and accuracy (`accuracy`) are monitored during training.

6.5 Results

The training results of the model, as presented in Tables 2 and 3, reveal the evolution of loss (`loss`), mean absolute error (`mae`), and accuracy (`accuracy`) metrics across training and validation epochs. The final evaluation on test data is documented in Table 3. The following graphic with this data (figure 6) shows very promising results.

The results indicate significant progress of the model during training, highlighting the learning capability of the neural network. Evaluation of test data provides critical insight into the model's generalization to new instances.

7 Conclusion

In conclusion, the results obtained from the training of the neural network demonstrate a highly promising performance. The training phase reveals a learning curve indicative of a model effectively capturing patterns within the data. The network consistently improves its accuracy over epochs, reaching a maximum accuracy of 0.9233 on the training dataset, 0.8431 on the validation dataset, and 0.8039 on the test dataset.

Looking ahead, the potential for even more favorable outcomes is anticipated with a larger dataset. Expanding the dataset to include thousands of images, as opposed to the 506 images used in this study, holds the promise of further enhancing the model's predictive capabilities. The observed trend of increasing accuracy suggests that a more extensive and diverse dataset could lead to even more robust and accurate predictions.

The current results lay a foundation for future endeavors in refining the model architecture and leveraging larger datasets.

8 Acknowledgements

We would like to thank the EAFEXP coordination and all the professors and teaching assistants for their support and guidance during this project.

References

[1] ZHONG, S. et al. **High harmonic generation**. (Reference details to be added)

[2] CORKUM, P. B. **Plasma perspective on strong field multiphoton ionization**. Physical Review Letters, v. 71, n. 13, p. 1994-1997, 1993.

[3] LAKSHMINARAYANAN, V.; FLEET, A. **Zernike polynomials:**

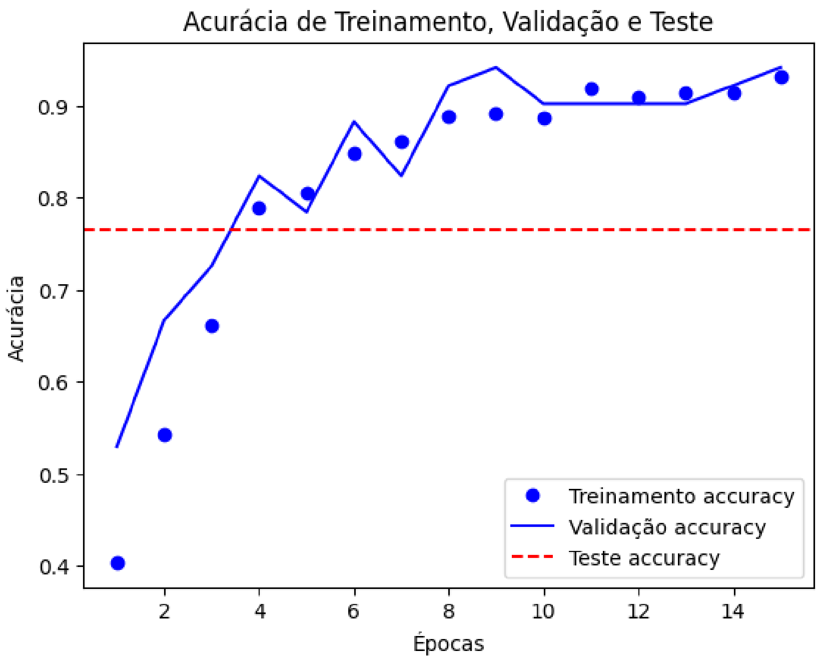


Figure 6: Training, Validation, and Test Accuracy

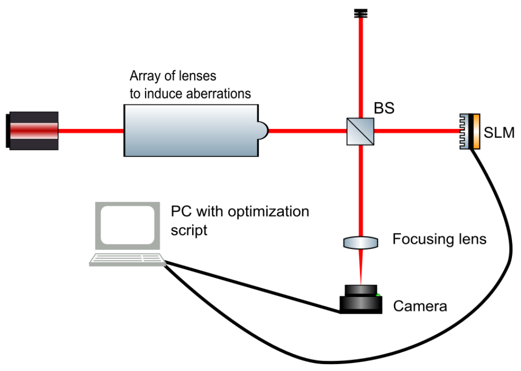
a guide. Journal of Modern Optics, v. 58, n. 7, p. 545-561, 2011.

[4] WANG, K. et al. **Deep learning for wavefront sensing: a review**. (Reference details to be added)

[5] JIN, K. et al. **Machine learning for adaptive optics**. (Reference details to be added)

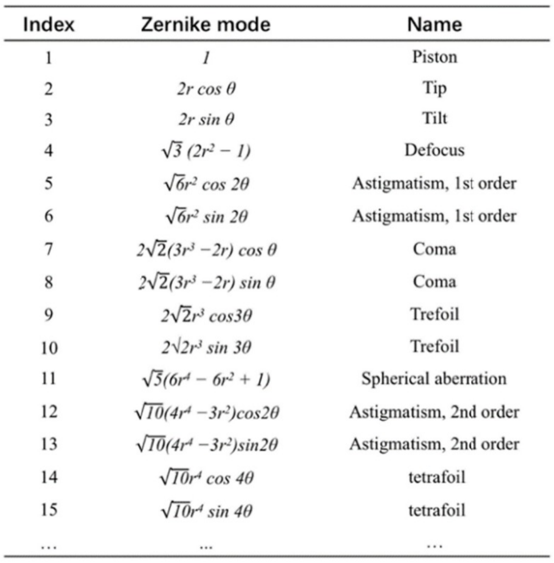
A Appendix

If it is of interest to the reader, additional figures and tables are shown below.



Array of lenses to induce aberrations
BS
SLM
PC with optimization script
Focusing lens
Camera

Figure 7: Initial setup for testing and developing optimization models.



Index	Zernike mode	Name
1	1	Piston
2	$2r\cos\theta$	Tip
3	$2r\sin\theta$	Tilt
4	$\sqrt{3}\,(2r^2-1)$	Defocus
5	$\sqrt{6}r^2\cos 2\theta$	Astigmatism, 1st order
6	$\sqrt{6}r^2\sin 2\theta$	Astigmatism, 1st order
7	$2\sqrt{2}(3r^3-2r)\cos\theta$	Coma
8	$2\sqrt{2}(3r^3-2r)\sin\theta$	Coma
9	$2\sqrt{2}r^3\cos 3\theta$	Trefoil
10	$2\sqrt{2}r^3\sin 3\theta$	Trefoil
11	$\sqrt{5}(6r^4-6r^2+1)$	Spherical aberration
12	$\sqrt{10}(4r^4-3r^2)\cos 2\theta$	Astigmatism, 2nd order
13	$\sqrt{10}(4r^4-3r^2)\sin 2\theta$	Astigmatism, 2nd order
14	$\sqrt{10}r^4\cos 4\theta$	tetrafoil
15	$\sqrt{10}r^4\sin 4\theta$	tetrafoil
…	…	…

Figure 8: Zernike Polynomials Scheme

Table 2: Training Results

Epoch	`loss`	`mae`	`accuracy`	`val_loss`	`val_mae`	`val_accuracy`
1	311.5240	3.6678	0.4431	0.2435	0.4036	0.5098
2	0.1999	0.3622	0.6559	0.1515	0.3165	0.8039
3	0.1369	0.2917	0.7426	0.1329	0.2934	0.7647
4	0.1061	0.2555	0.7995	0.0951	0.2453	0.7843
5	0.0663	0.2026	0.8119	0.0721	0.2039	0.8235
6	0.0468	0.1649	0.8441	0.0677	0.1995	0.7647
7	0.0307	0.1352	0.8564	0.0355	0.1449	0.8039
8	0.0219	0.1140	0.8911	0.0411	0.1450	0.8431
9	0.0155	0.0957	0.8985	0.0392	0.1440	0.8627
10	0.0137	0.0904	0.9233	0.0406	0.1391	0.8431

Table 3: Test Data Evaluation

`loss`	`mae`	`accuracy`
0.0315	0.1208	0.8039

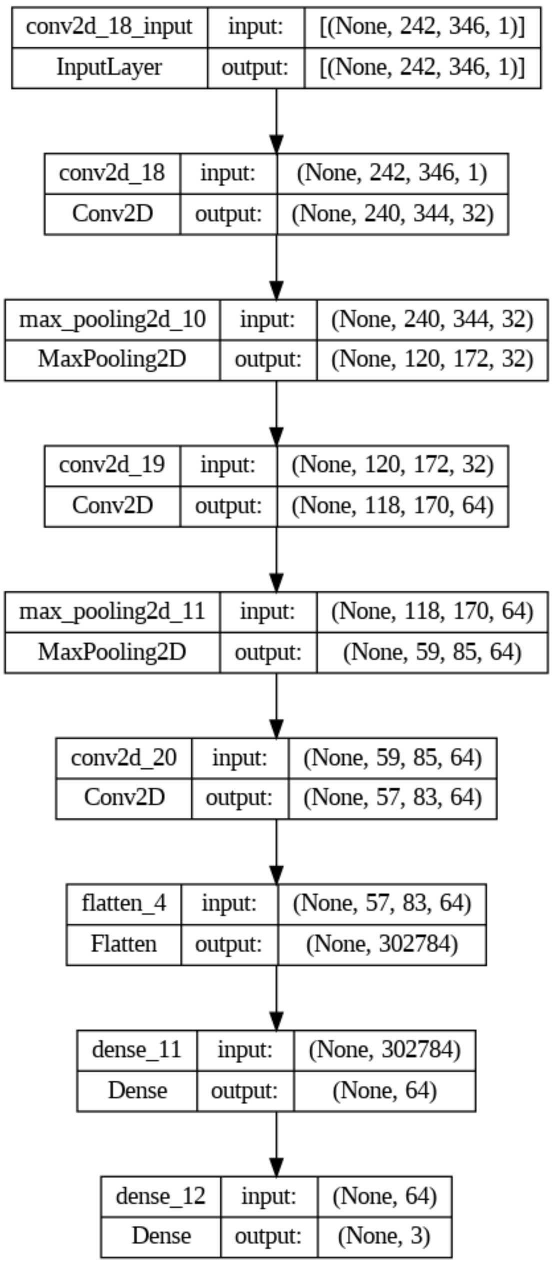


Figure 9: Neural Network Architecture