Case Study 4: Imaging and Ray Tracing

Kaan Dincer and Nick Falshaw

***Abstract*— In this case study the formation of images was described and modeled in MATLAB using a function that simulated propagation of light rays from an object to an image. Using a plot and a theoretical curve, the distance of an object from the lens, z1 was compared to the corresponding distance of the image produced from the lens, z2 over a range of z1 vales from 1 mm to 1m, and the difference between a "real" image and a "virtual" image was analyzed. Also, the concept of depth of field and how the f-number affects images was explored by simulating the rays of two images and plotting one on top of the other. Lastly, given a large dataset of image rays with three visible images and one hidden image. The rays were simulated to show the images in focus at specific points and the hidden image was uncovered.**

I. Introduction

Optical imaging and ray tracing are topics that have many applications and that happen all the time without being noticed or thought about. Any time a person is seeing something their eye is working as an optical imaging system by bending light so that all the rays hit a specific spot of the retina and an image is formed. Another common use of optical imaging is the formation of images through the use of cameras, which bend light rays through a lens. Optical imaging also has many medical applications because it creates a non-invasive way to look inside the body using visible light and special properties of photons to obtain detailed images of organs and tissues as well as smaller structures including cells and even molecules (NIBIB). Using the paraxial approximation ($sin\theta \approx \theta \; and \; tan\theta \approx \theta$) a 2x2 matrix called a transfer matrix can be used to simulate the transformation of rays through an optical imaging system. Multiple matrices for different parts of the system can be combines to make one overall ray transfer matrix (Osorio).

II. Methods

A. Part 1: Ray Optics

The task of part 1 was to complete the "simRayProp" function to simulate the propagation of rays through free space and through a lens. This function takes in a transfer matrix, M, with dimensions 2x2, and vectors "y_in" and "theta_in", each with dimensions 1xN, where N is the number of rays. "y_in" represents the locations of N rays before the optical component M and "theta_in" represents the angles of N rays before the optical component M. This functions outputs "y_out" and "theta_out", which are also 1xN vectors, and they represent the locations and angles of N rays after the optical component M. In order to build this function, first "y_in" and "theta_in" had to be combines to make a 2xN matrix for the dimensions to agree with M for the matrix multiplication. Row 1 was "y_in" and row 2 was "theta_in". Multiplying M by the combined "y_in" and "theta_in" matrix produced a 2xN output matrix. The top row of this output matrix was set to be "y_out" and the bottom row was set to be "theta_out".

B. Part 2: A Simple Imaging System

In this part of the case study a simple imaging system was simulated for an objects 250 mm and 45 mm in front of a lens with a focal length of 100 mm. In order to implement this simulation "y_in" and "theta_out" were initialized and then "simRayProp" was called three times using three different matrices: M_z1, M_f, and M_z2. It was called three times in order to simulate the propagation of the rays from the object to the lens, the bending of the rays through the lens, and then the propagation of the rays after the lens to the location of the image. Then the 2D ray diagram was plotted piece by piece. This process was done twice, once using z1 = 250 mm and, then using z1 = 45 mm. After the 2D ray diagrams were produced, a plot was produced to show z2, the location of each image, corresponding to z1, the distance of the object from the lens for a range of z1 values from 1 mm to 1 m. To do this the a for loop was used to iterate through the specified range of z1 values, increasing z1 by 1 mm every iteration and calculating z2 for each z1 value. These points were then plotted and then a theoretical curve was added. The theoretical curve was given by the imaging equation: $z_2 = (\frac{1}{f} - \frac{1}{z_1})^{-1}$. Finally, for this part of the case study the magnification m vas plotted against the range of z1 values from 1 mm to 1 m. Again, a for loop was used and z1 was incremented by 1 mm each iteration. For each z1 value the m value was calculated using $m = \frac{y_2}{y_1}$ and these points were plotted. A theoretical curve given by $m = (1 - \frac{z_1}{f})^{-1}$ was added to the plot.

C. Part 3: Depth of Field

For this part, the first task was to plot the radius of the circle of confusion vs the f-number. The first step was to assign values for frequency, z1 of object, z1 of lens and initial height, so these variables could be used later on. As the f-numbers range from 1.4 to 22, the variable fNum was incremented by 0.02601 from 1.4 to 22, and the radius was initialized to a vector of zeros of size fNum. As the goal is to plot the values of the radius as the f-number changes, a for loop from 1 to size of fNum was implemented. Using given z1 (4m) and focal point (f = 0.2 m) values and calculated z2 value a transfer matrix, 'M_trans', was built. From the equations for finding the height (y) and the angle (theta) the vectors for initial y and theta were created. The equations used were y = f/(2*f-number) and θ= y/z1 of object. Then the transfer matrix, the vector for initial y and the vector for theta were passed in as parameters for the simRayProp method built in Part 1 of the case study. The y_out values for each f-number were taken from the simRayProp method and plotted against the changing f-number values.

The second task, was to combine two images and then to focus on one of the images, while blurring the other, as if a camera was focusing on an object. The images were first converted into rays, using the built-in MATLAB function 'img2rays'. The name of the image, width, number of rays, and the maximum angle were passed in as parameters for the function. The x and y values, as well as the angle values received from this method were then

passed in as parameters for the simRayProp method. A transfer matrix derived from given values of z1 and frequency, and calculated value of z2, was also passed in as the parameter. This process was done by using different f-numbers. A large f-number corresponds to a larger depth of field (DOF), from the equation # = f/D. For the first imaging system, a relatively large f-number of 22 was passed in for the Turkey image, giving it a large DOF, and a relatively small f-number of 1.4 was given to the snow globe image, giving it a small DOF. This produced an image where the Turkey image was focused and clear, and the snow globe image was blurry and at the back of the turkey image. For the second imaging system, relatively small f-numbers were passed in for both images, giving them small DOF values, which produced an image where both images were focused and clear. In order to concatenate these images, the built-in MATLAB function cat was used, and in order to convert these rays back to images the built-in function rays2img was used, and the output images were displayed.

D. Part 4: Computational Imaging Challenge (Depth of Field)

First, the given .mat files of images were imported into MATLAB. The images in this file were converted into rays. This file contained multiple images, so z1 needed to be changed in a certain range in order to find the optimal z1 for each image (the z1 that produced a clear image). So z1 was incremented by 0.001 from 0.6 to 0.7, and a for loop from 1 to the length of z1 was implemented. In the for loop, rays were simulated using simRayProp for each z1 value, and the output values of ray x and ray y from this function was passed in as parameters to the built-in function rays2img. The output image was displayed for each z1 value.

III. Results

A. Part 1: Ray Optics

The completed "simRayProp" function was proven to be correct through the completion of the other parts of this case study. In other parts of the case study the rays would not have been properly simulated if "simRayProp" did not work.

B. Part 2: A Simple Imaging System

As seen from Figures 1 and 2 the initial distance of the object from the lens, z1 influences the distance of the image from the lens, z2. When z1 was 250 mm the rays passed through the lens produced an image below the optical axis on the other side of the lens as seen in Figure 1. This is defined as a "real" image since z2 > 0. However, in Figure 2, when z1 was 45 mm the rays were reflected off the lens and produced an image further in front of the lens. Since z2 < 0 this image is defined as a "virtual" image. Figure 3 supports this definition as it shows the relationship between z1 and z2. For z1 values less than 0.1 m (the focal length), z2 was negative and a "virtual" image was produced, but for z1 values greater than 0.1 m the z2 value was positive and a "real" image was produced. As the z1 values got further away from the focal point, the image was produced closer to the lens.

In both Figure 3 and Figure 4 the calculated MATLAB simulations agree with the given theories. In both figures all the plotted points that were calculated in MATLAB fit right on the theoretical curve. An "upright" image is defined as an image where m > 0, and conversely an "inverted" image is defined as an image where m < 0. In Figure 4 the "upright" images occur when z1 < 0.1 m which corresponds to when virtual images are produced. Likewise, "inverted" images are produced when z1 > 0.1 m which is when "real" images are produced. This makes sense because looking at Figure 1, a 2D diagram where a "real" image is produced and z1 > 0.1 m, the image is located below the optical axis, indicating that it has been inverted. In Figure 2, z1 < 0.1m and a "virtual" image is produced so it makes sense that the location of the image is above the optical axis since it is an "upright" image.

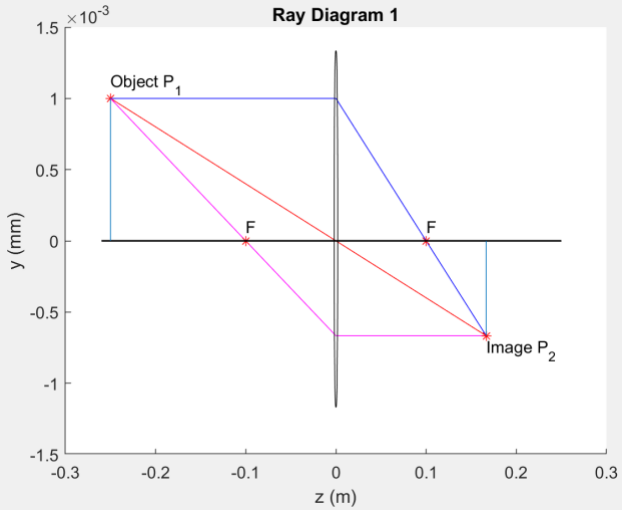

Figure 1: Image formation through a thin lens with Object P1 250 mm in front of the lens and a lens focal length of 100 mm.

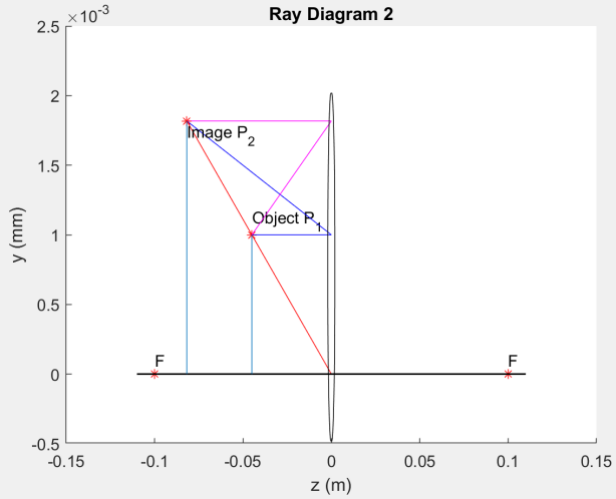

Figure 2: Image formation through a thin lens with Object P1 45 mm in front of the lens and a lens focal length of 100 mm.

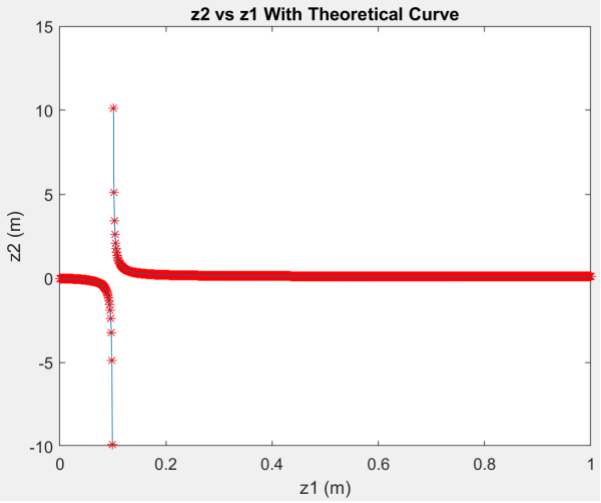

Figure 3: Corresponding location of each image (z2) for a range of z1 distances from 1 mm to 1 m and a theoretical curve given by $z_2 = (\frac{1}{f} - \frac{1}{z_1})^{-1}$.

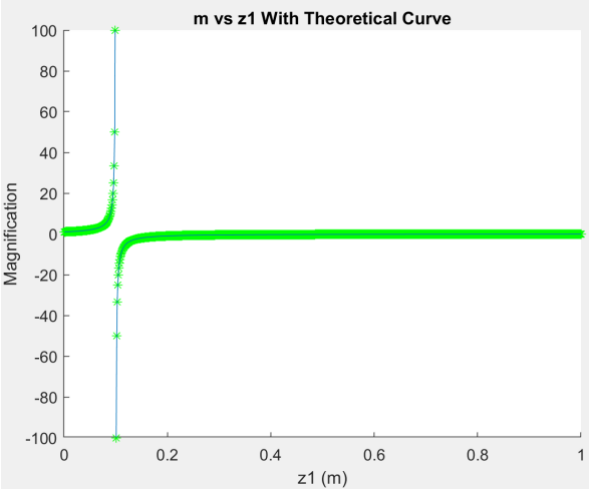

Figure 4: Corresponding magnification of each image for a range of z1 distances from 1 mm to 1 m and a theoretical curve given by $m = (1 - \frac{z_1}{f})^{-1}$.

C. Part 3: Depth of Field

As it could be seen from Figure 5, the radius of circle of confusion decreased as the f-number increased. Larger f-numbers correspond to smaller apertures, which lead to smaller radius of confusion, and vice versa. When radius of confusion is bigger objects in the background tend to be blurred or focused.

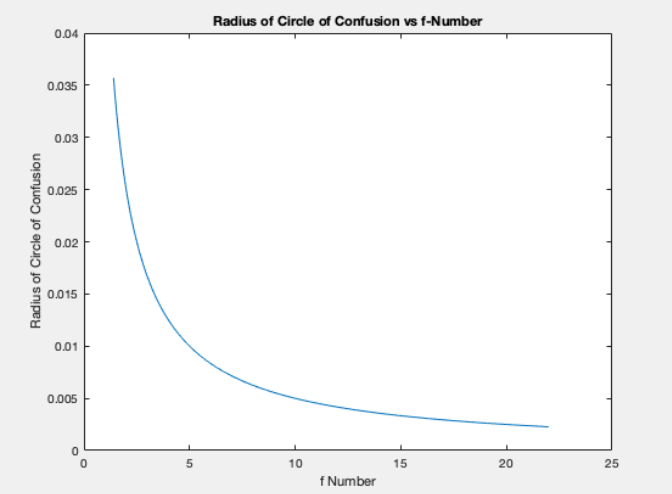

Figure 5: Plot of Radius of Circle of Confusion vs the f-Number, given a f-Number range from 1.4 to 22

For initial images of same and small DOF values the two images were observed to be focused and clear at the same time, as seen in Figure 6. However, when the turkey image was given a large DOF compared to the relatively small DOF value of the snow globe image, it was observed that the turkey image was in front of the snow globe image and clear, while the snow globe image was blurry and in the background, as seen in Figure 7. This is because when the DOF of an image is relatively smaller, this means it has a larger radius of confusion, which leads to the image being blurry.

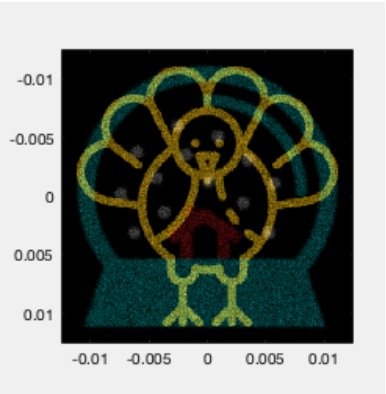

Figure 6: Display of an image produced from two initial images. The produced image was produced using small DOF values for both initial images, making them both clear and in focus.

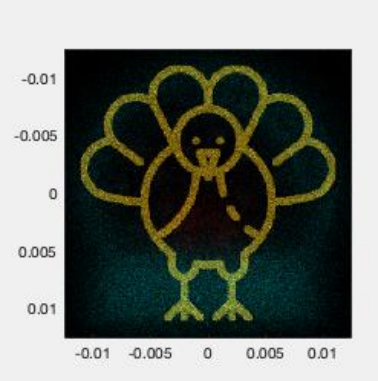

Figure 7: Display of an image produced from two initial images. The turkey image was given a large DOF and the snow globe image was given a small DOF, making the turkey image focused and clear, and the snow globe image blurry and out of focus.

D. Part 4: Computational Imaging Challenge (Depth of Field)

The methodology to find the 4 hidden images was explained in Part 4 of the Methods section in this report. The reason we know our methodology was successful in finding the images is that we take in the rays of each images and each image is made up of rays with similar quantities. This is because the theta values of the rays of an image should theoretically be similar, as they must converge. Additionally, compared to other images the x and y values of the rays of an image should be similar compared to the rays of other images. Therefore, working on these rays and converting them into images, we know our methodology returned the right images.

The mathematical metric used to determine that the found images were correct, was to see that the rays converged to same point. When the images were sharp and clear it was determined that the found images were correct. However, even though our method returned the correct image, it wasn't perfectly efficient, as the secret image and the blackhole image were not completely clear and focused.

Traditional cameras cannot refocus images after they're taken, and cannot create sharp images without physically

changing the lens because the theta values of x and y are not taken in by the camera. Therefore, cameras cannot simulate the path of the ray. So, when an image is taken by a camera the image is stuck in the position it was taken in, and there is no way of refocusing it.

In order to construct a camera that doesn't have a lens, that camera should be able to take in data of x and y positions, as well as x and y angles. A transfer matrix could be created that would take in these data, and simulate the path the rays. Therefore, it might be possible to create a camera without a lens.

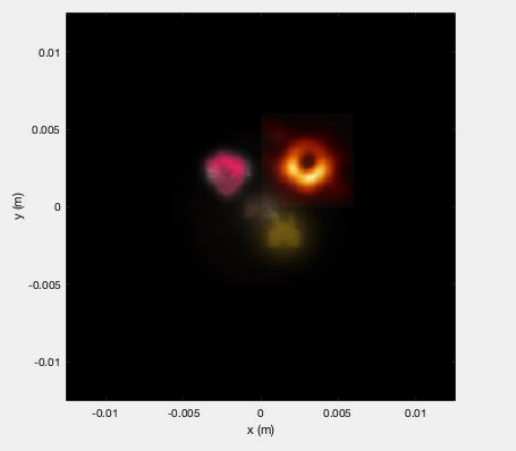

Figure 8: Display of clearest image of blackhole from the file lightField.mat.

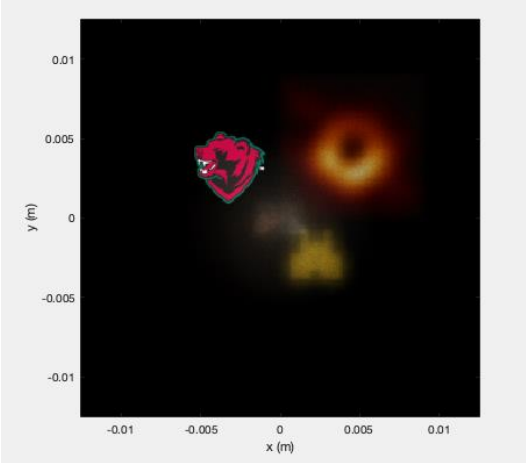

Figure 9: Display of clearest image of WashU bear logo from the file lightField.mat.

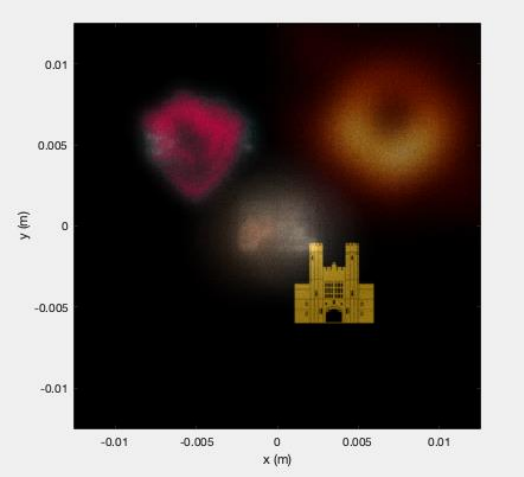

Figure 10: Display of clearest image of Brookings from the file lightField.mat.

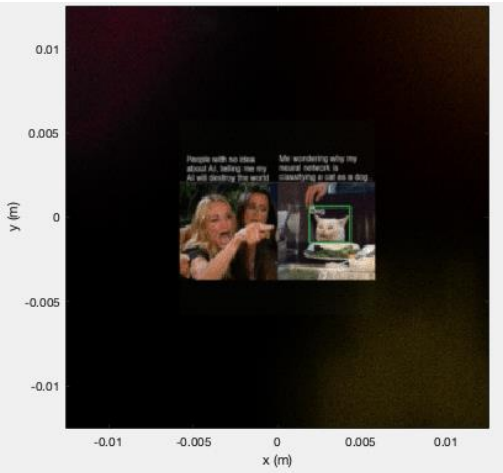

Figure 11: Display of clearest image of the "secret image" from the file lightField.mat.

IV. Conclusions

From the results in part 2 comparing z1 with z2 and z1 with m it was determined that the classification of an image as either a "real" image or a "virtual" image depends on the object's location in relation to the focal point, f. If an object in front of a lens is further away from the lens than the focal point is, z2 will be positive, the magnification will be negative, the image will be inverted, and a "real" image will be produced. Contrarily, if an object in front of a lens is nearer to the lens than the focal point is, then z2 will be negative, the magnification will be positive, the image will be upright, and a "virtual" image will be produced. Part 3 results showed how f-numbers relate inversely to the size of the radius of confusion and that changing f-numbers affects which objects are in focus. Finally, it was shown that given initial theta values of rays, a blurry image can be made clearer using basic linear algebra and matrix operations.

References

[1] "Optical Imaging." *National Institute of Biomedical Imaging and Bioengineering*, U.S. Department of Health and Human Services, www.nibib.nih.gov/science-education/science-topics/optical-imaging.

[2] Osorio, Clara I, et al. "Ray Transfer (ABCD) Matrix Analysis." *RayLab*, www.raymak.com/wp/user-guide/ray-transfer-matrix-analysis/.