Optical Imaging as a Linear System

David Barbosa
Department of Electrical and Systems Engineering
Washington University in St. Louis
St. Louis, USA
barbosa@wustl.edu

Paul Pham
Department of Electrical and Systems Engineering
Washington University in St. Louis
St. Louis, USA
p.h.pham@wustl.edu

***Abstract*—This report details the design and implementation of a linear optical imaging system in MATLAB to simulate and analyze the behavior of light rays, specifically the formation of images in free space. This study establishes the ray tracing methodologies to simulate the propagation of light rays through free space and thin lenses using ray-transfer matrices. By utilizing this linear system framework to compute ray trajectories, the model enabled clear visualization of image formation and facilitated the investigation of ray convergence as a function of propagation distance, d_2, and focal length, f. Furthermore, the optical model was implemented to address the following practical imaging challenges: [1] A visualization of ray propagation through free space; [2] A reconstruction of a sharp image from a light field; and [3] A computational analysis to identify three hidden objects within the holographic dataset (`lightField.mat`).**

I. Introduction

Optical imaging systems, ranging from simple telescopes to complex microscope imaging systems for high-resolution imaging of small structures, are fundamental to sensing and acquiring data in every field. Light's propagation through free space can be accurately described using ray optics, also known as geometrical optics. This case study's simplified model treats light as rays traveling in straight lines, creating a set of geometrical rules that can describe how the rays travel, allowing the application of linear algebra to predict the system's behavior.

The core foundation of this case study is the representation of an optical system as a linear transformation. By defining the ray's state by its position, y, and angle, θ, relative to the optical axis, a model of the ray's propagation can be formed using ray-transfer matrices. This approach relies on the approximation $\tan\theta \approx \theta$, which linearizes refraction and travel distance. The relationship between the input ray (y_1, θ_1) and the output ray (y_2, θ_2) is given by:

$$\begin{bmatrix} y_2 \\ \theta_2 \end{bmatrix} = M \begin{bmatrix} y_1 \\ \theta_1 \end{bmatrix} \tag{1}$$

The primary objective of this report is to design and implement a MATLAB-based simulation of an optical system to analyze ray propagation through free spaces and lenses and visualize how it can propagate through an imaging system. Deriving and utilizing ray-transfer matrices (M_d) and thin lenses (M_f), we can model the formation of the images. Finally, by extending this framework to the field of computational imaging, we are able to analyze a holographic light field dataset (`lightField.mat`) and apply these linear models to digitally refocus the rays and identify the three hidden objects within the data.Key stages to be successful include:

- Implementation of the ray-transfer matrix, M_d, to simulate the linear trajectory of rays over a distance d_1 in free space.
- Implementation of the thin lens matrix, M_f, to simulate the refraction of rays and the formation of focused images based on focal length f.
- Utilization of the function `rays2img` to simulate a camera sensor and to visualize the ray data from the `lightField.mat` data set.
- Implementation and designs of methods of an optical imaging system by calculating the necessary image plane distance, d_2, to successfully identify the three hidden objects in the holographic light field. (change this after we find the correct methods to see the images clearly)

II. Methods

A. Ray Tracing

To simulate the behavior of a ray passing through a three-dimensional optical system, the ray transfer matrix method was used. A ray is defined by its position and angle in both the x and y planes relative to the optical axis (z-axis), which is represented by the vector $\mathbf{r}$:

$$\mathbf{r} = \begin{bmatrix} x \\ \theta_x \\ y \\ \theta_y \end{bmatrix} \tag{2}$$

The system transforms these ray vectors using 4×4 matrices. The paraxial approximation, $\tan\theta \approx \theta$, allows the propagation of the rays through the optical system to be modeled as a linear transformation.

As a ray travels through free space over a distance d, the angle remains constant while the position changes linearly based on the angle value. The 3D ray-transfer matrix M_d, representing the propagation in free space and the distance the ray travels in the x and y directions, is modeled as:

$$M_d = \begin{bmatrix} 1 & d & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

The thin lens matrix, M_f, alters the angle of the ray based on its entry height and angle, modeling the focal effect. The exit angle is linearly modeled by:

$$\theta_2 = -\frac{y}{f} + \theta_1 \tag{4}$$

This formulation showcases how the lens changes the angle of the rays while leaving the position unchanged. The focal point matrix is defined as:

$$M_f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1/f & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/f & 1 \end{bmatrix} \tag{5}$$

The simulation was initialized with two point sources located at $x = 0$ and $x = 0.01$ m. Rays were generated with angles varying between $-\pi/20$ and $\pi/20$. The optical system was then modeled with three distinct steps:

1) First, the rays were propagated a distance $d_1 = 0.2$ m from the source to the lens plane using the matrix M_{d1} (Figure 1).
2) Second, a thin lens set with the focal length $f = 0.15$ m and radius $r = 0.02$ m was placed at the $z = 0.2$ m mark. A logical indexing filter `idx` was implemented to separate rays that struck the lens from those that were missed. Rays that satisfied the condition $|x_{ray}| \leq r_{lens}$ were transformed by M_f and were bent. Rays outside the radius continued to be straight lines.
3) Lastly, the rays that hit the lens propagated a second distance, d_2. To ensure a sharp image, a compatible distance d_2 was calculated via the focal lens equation:

$$\frac{1}{d_2} = \frac{1}{f} - \frac{1}{d_1} \implies \frac{1}{0.15} - \frac{1}{0.2} \implies d_2 = 0.6 \text{ m} \tag{6}$$

B. Simulating a hologram

In this section, the linear system framework was applied to computational imaging using the holographic dataset containing 3×10^6 rays, `lightField.mat`, to process the rays and recover a recognizable 2D image. The provided camera simulation function, `rays2img`, was utilized to visualize the raw rays directly. The sensor width was initially set at 0.005 m with a resolution of 200 pixels.

To determine if the image could be clarified by moving the sensor, the rays were propagated at a distance $d_1 = 0.2$ m using the free-space matrix M_{d1}.

In order to achieve an optimal and efficient optical imaging system, the input rays would have to be computed using a series of ray transfer matrices M_{d1} (3), M_{d2} (3), M_f (5), and multiplying these input rays by these matrices to achieve a desired result of a sharper image, which can be expressed in the form:

$$\begin{bmatrix} x_2 \\ \theta_{x2} \\ y_2 \\ \theta_{y2} \end{bmatrix} = M_{d2} M_f M_{d1} \begin{bmatrix} x_1 \\ \theta_{x1} \\ y_1 \\ \theta_{y1} \end{bmatrix} = M \begin{bmatrix} x_1 \\ \theta_{x1} \\ y_1 \\ \theta_{y1} \end{bmatrix} \tag{7}$$

In this system, the order of matrix operations is critical due to the non-commutative nature of matrix multiplication. The system is modeled with the initial propagated distance $d1$ (denoted by M_{d1}), followed by transmission through a thin lens with focal length f (M_f), and lastly propagated over distance $d2$ (M_{d2}). The entire order of matrix operations can be denoted as M.

With the fixed value of $f = 0.08$ by using MATLAB to simulate multiple cycles of expressions (6)(7) with different values of $d2$ and then simulating the images over a range of 0.05 to 0.30. Afterwards, the images was scored by a Laplacian edge detector with a 3x3 matrix representing the kernel K of the system to detect and enhance the edges of the outputs [1]:

$$K = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{8}$$

Using this scoring system as well as the simulation of the images, the best value of $d2$ was determined to be $d2 \approx 0.1000997$, and the Laplacian score was determined to be approximately 837.6. By modifying the values of $d2$ and thus changing the values of $d1$, which would then be used in the expressions (6)(7) it was found that this created the sharpest image from the optical system.

C. Competition

The aim of the competition was to identify and separate the three objects that the rays create to create a separate, complete image for each object. It was previously observed that the rays have a separate factor in addition to distance and focal length that separates them. Referring back to the expression (7) for the input of an optical system, it can be observed that angles and direction of the rays are the other key factors in the propagation. In this instance, the angles in the x and y directions dictate the depth of the rays compared to one another. Figure 5 demonstrates this, as the image of Professor Bruno Sinopoli is dimmer than the Unitree robot dog or the Washington University Crest. This demonstrates that while the images reach the same point, they have differing angle directions at that point, which causes the altered depth between the images.

To isolate each image and simulate it as a separate figure, MATLAB was used to detect and separate the angles at which the rays lie using a k-means algorithm. This algorithm was employed to group similar angles and determine the exact values of each point. The algorithm successfully partitioned the rays into three clusters centered at approximately -0.105 radians (WashU Crest), 0 radians (Professor Sinopoli), and 0.105 radians (Boston Dynamics Robot Dog) (See Figure 6).

Although the three objects were successfully spatially separated and were distinguishable, the initial raw output was observed to be highly granular. This noise resulted from the reduced ray density within each cluster, leaving gaps between pixels. This observation led to the implementation of a multi-stage filtering approach. To smooth the "salt and pepper" granular noise, a median filter (`medfilt2`) was first applied.

Consequently, a Gaussian smoothing kernel (`imgaussfilt`) was implemented to lead to a smoother transition between the pixel values. Lastly, Contrast Limited Adaptive Histogram Equalization (`adapthisteq`) was implemented to enhance the dynamic range, ensuring that fainter details such as the logo on the robot dog was clearly visible against the dark background. [2]

III. Results & Discussion

A. Ray Tracing Results

The resulting ray trajectories in Figure 1 confirm that the rays diverge from distinct sources and converge back to unique points at $z = d_1 + d_2 = 0.8$ m.

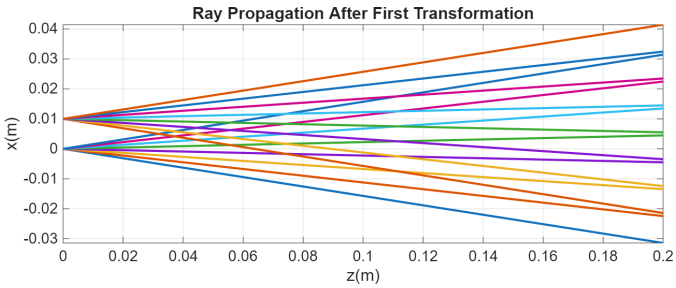

Fig. 1. Ray propagation after the first transformation (M_{d1}) over a distance $d_1 = 0.2$ m. Rays diverge from two distinct point sources at $x = 0$ and $x = 0.01$ m.

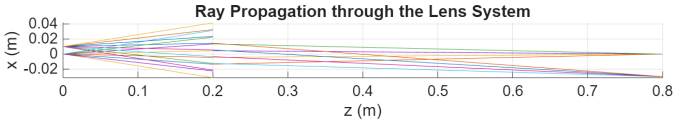

Fig. 2. Complete ray propagation through the lens system. Rays diverge over $d_1 = 0.2$ m, are refracted by the lens at $z = 0.2$ m, and converge at the image plane after $d_2 = 0.6$ m.

The observed convergence point at $z = d1 + d2 = 0.8$ precisely matches the Gaussian lens equation (6) $\frac{1}{0.2} + \frac{1}{0.6} = \frac{1}{0.15}$, confirming the validity of the matrix-based simulation.

B. Results of Hologram Simulation

The resulting visualization displayed a field of incoherent noise, as shown in Figure 3. Modifying the sensor's pixel count caused the image to darken when increased or resulted in a blockier, blurred image when decreased. Modifying the sensor's width acted as a digital zoom but did not improve image clarity. It was concluded that the issue was not the sensor configuration, but rather that the rays were unfocused.

The matrix M_{d1} linearly models the rays' trajectory from d_1 to the focal plane; however, the propagation cannot "bend" the diverging rays back toward a converging point, resulting in an image with the pixels spread further apart (Figure 4). This

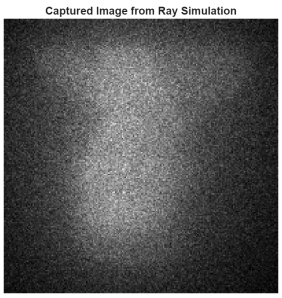

Fig. 3. Initial visualization of the raw light field data displaying incoherent noise.

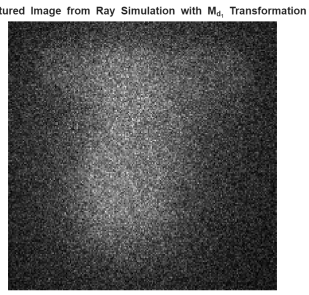

Fig. 4. Image resulting from free-space propagation ($d_1 = 0.6$ m) without a lens. The image remains blurry with no discernible objects.

confirmed that a full optical imaging system was required to create a sharp image.

The resulting image presented in Figure 5 reveals that at the point where the image is the clearest, two other objects generated by the rays in the system cause the initial image to be overlapped with other figures. The overlap implies that by altering the distances with a fixed focal length, the distances between rays become nearly indistinguishable, and the resulting images created by these rays overlap one another. With the distances and focal lengths being constant, this implies that the "holographic" nature of the dataset encodes spatial information not just in ray position, and that there is another factor that separates the images from one another and keeps the rays from melding together. Since the true unknown distance $d1$ was unknown, there was no direct way of solving the thin lens equation (6) directly. The iterative search for $d2$, guided by the Laplacian sharpness metric, was essential to reverse-engineer the scene geometry. By implementing these systems, as well as a fixed focal length, it was able to clearly approximate the closest value of $d2$ and validate that value with a score that demonstrates the peak of the images sharpness. These values were found to be $d2 \approx 0.1000997$, $score \approx 837.6$.

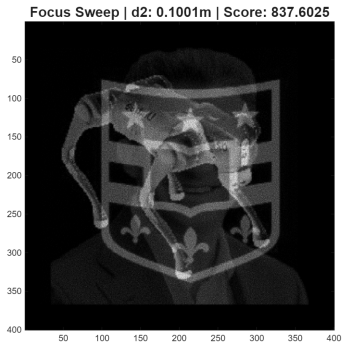

Fig. 5. Optimal focus result ($d_2 \approx 0.10$ m) showing the superposition of the Professor Sinopoli, Unitree Go2 quadruped robot, and WashU Crest.

C. Competition Results

The angles were input into the k-means algorithm and demonstrated the separation presented in Figure 6.

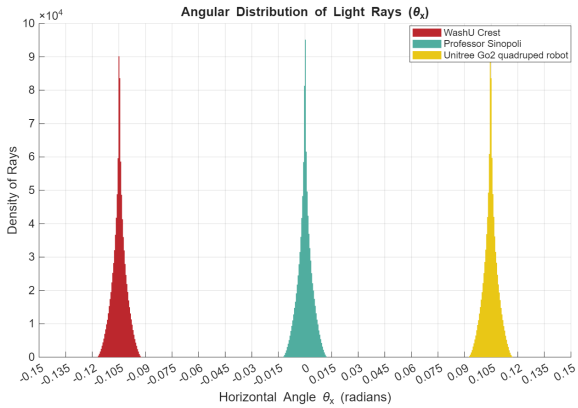

Fig. 6. K-means clustering of ray angles (θ_x) revealing three distinct centroids at $-0.105, 0$, and 0.105 rad.

The successful separation of objects was achieved not just by spatial masking, but by exploiting the parallax inherent in the light field. By filtering for specific ray angles (simulating a sub-aperture view), we could 'look behind' occlusions and isolate rays corresponding to specific depths.

Once separated, each cluster was processed through the optical system (7). The resulting images are presented in Figure 7.

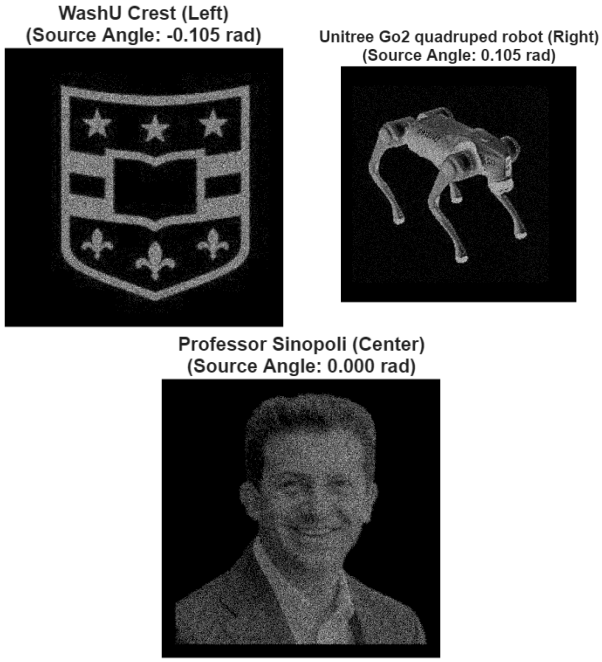

Fig. 7. The three hidden objects separated via K-means clustering.

The raw reconstruction suffered from 'shot noise' due to the discrete sampling of rays. The Gaussian filter was effective at smoothing this high-frequency granularity, while CLAHE was critical for revealing low-contrast features (like the robot dog's logo) that were otherwise lost in the shadows. The resulting final images are presented in Figure 8.

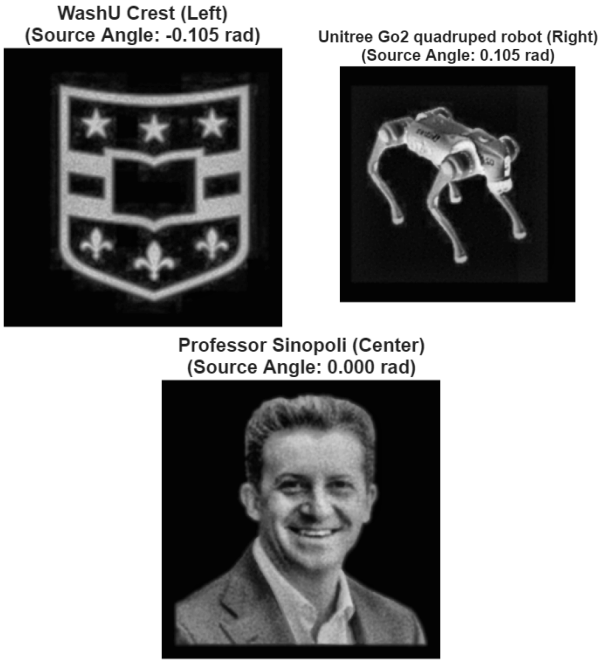

Fig. 8. Final filtered results.

IV. Limitations and Future Work

While the matrix-based simulation and K-means clustering successfully reconstructed and separated the holographic images, several limitations are inherent to this approach. First, the ray transfer matrix method relied on the small-angle approximation. In a real-world optical system with larger lenses, this approximation would break down. Furthermore, the results of this case study were limited by the unknown factor of the first distance $d1$ throughout simulations, which made the design of the system hinge on finding the closest approximation of $d2$ to find the clearest image. Additionally, the addition of a fixed focal length may have caused the system to not find the clearest image due to its imprecise nature.

Moreover, the angle approximation utilizing the k-means algorithm is not fully precise and may not have found the exact angle which the rays converged at. Consequently, this algorithm required the number of clusters (k=3) to be predefined. In a blind scenario where the number of objects is unknown, this method would require an iterative approach, such as the "elbow method," to determine the optimal cluster count.

Upon revisiting the optical systems, perhaps these systems could be fine-tuned to make more precise measurements, allowing for sharper image quality and better ray approximations.

V. Conclusion

This case study successfully demonstrated the development and implementation of an optical imaging system to model different sets of rays across multiple angles, thereby creating various images. The key findings of this case study include:

1) The development of an expression to model and simulate rays at various different angles.
2) Analysis of the system demonstrated that modifying the parameters of $d1, d2, f$ would not develop the accurate, sharp image desired by the optical imaging system.
3) The successful implementation of an optical imaging system that would iterate through the various rays and determine the optimal distance $d2$ from a fixed focal length f, that would develop the sharpest possible image from the given rays
4) The successful identification and separation of the angles within the various rays in the competition set using an optimal k-means algorithm, identifying the Washington University Crest at -0.105 rad, Professor Sinopoli at 0 rad, and the Unitree Robot Dog at 0.105 rad.

Overall, this project demonstrates how a successful optical imaging system can be used to determine and evaluate information from a complex, multidimensional dataset of rays, allowing for detailed observation of optical phenomena and how this can be observed with greater detail using the implementation of ideal distances and lenses with differing focal lengths.

CONTRIBUTIONS

Paul Pham contributed to the design and implementation of the ray transfer matrices and developed the MATLAB visualization of the rays propagation through a thin lens, specifically implementing the indexing to distinguish between rays that hit the lens and those that miss. He also contributed to the implementation of the K-means clustering algorithm to sort the rays by angle and assisted in the multi-stage image filtering. Furthermore, he contributed to the formatting and design of the LaTeX report.

David Barbosa contributed to the design of the ray transfer matrices and developed the iterative analysis of the images with differing distances with a fixed focal length to find the sharpest possible image. He also implemented the Lagrangian scoring metric to analyze the precise distances that would demonstrate the sharpest image. Moreover, he discovered the separation of the rays by angle that would later inform the k-means algorithm. Additionally, he contributed to the writings and designs of the LaTex report.

REFERENCES

[1] J. L. Pech-Pacheco, G. Cristobal, J. Chamorro-Martinez, and J. Fernandez-Valdivia, "Diatom autofocusing in brightfield microscopy: a comparative study," in *Proceedings 15th International Conference on Pattern Recognition*, vol. 3, pp. 314-317, IEEE, 2000.

[2] K. Zuiderveld, "Contrast limited adaptive histogram equalization," in *Graphics Gems IV* (P. S. Heckbert, ed.), pp. 474-485, San Diego, CA: Academic Press, 1994.

APPENDIX

A. Ray Transfer Matrices

The fundamental system matrices were defined as follows for the simulation:

```
% 1. Input Propagation Matrix (Distance d1)
M_d1 = [1, d1, 0, 0;
        0, 1, 0, 0;
        0, 0, 1, d1;
        0, 0, 0, 1];

% 2. Thin Lens Matrix (Focal Length f)
M_f = [1, 0, 0, 0;
      -1/f, 1, 0, 0;
       0, 0, 1, 0;
       0, 0, -1/f, 1];

% 3. Output Propagation Matrix (Distance d2)
M_d2 = [1, d2, 0, 0;
        0, 1, 0, 0;
        0, 0, 1, d2;
        0, 0, 0, 1];

% Total System Matrix
M = M_d2 * M_f * M_d1;

% Laplacian Kernel for Sharpness Scoring
k = [0 -1 0; -1 4 -1; 0 -1 0];
```

B. K-Means Clustering for Angular Separation

The angular separation of the three objects (WashU Crest, Sinopoli, Robot Dog) was achieved using the built-in `kmeans` function on the ray angles (θ_x, θ_y):

```
% Extract angles from ray data
angle_data = [rays(2,:)' rays(4,:)'];

% Partition into k=3 clusters
k = 3;
[cluster_idx, cluster_centers] = kmeans(angle_data, k);

% The centroids C corresponded to [-0.105, 0, 0.105] rad
```

C. Image Reconstruction Filtering Process

The following filtering pipeline was applied to the raw output to reduce granular noise and enhance contrast:

```
% 1. Median Filtering (Salt & Pepper Noise Removal)
img_denoised = medfilt2 (img, [2 1]);
img_proc = double(img_denoised) / 255;

% 2. Gaussian Smoothing (Granularity Reduction)
img_smooth = imgaussfilt(img_proc, 3);

% 3. Contrast Enhancement (CLAHE)
img_final = adapthisteq(img_smooth, 'ClipLimit', 0.01);
```