Light Field Frame Translation

by

Anahita Nik aien

A thesis submitted to the

Department of Computer Science

to fulfill the requirements for the degree of

Bachelor of Science (Honour's).

Supervisor: Dr. Matthew Hamilton

Memorial University of Newfoundland

St. John's, NL, Canada

January 5, 2025

St. John's Newfoundland

Abstract

In this project, we implement an algorithm for light field frame translation, a technique that focuses on translating frames within a light field to achieve optimization of depth resolution of light field images of a given 3D scene. By translating the frame of reference for a light field image of a 3D scene, the 3D scene can often be imaged at a similar level of detail but with a light field image with a much lower directional resolution.The algorithm is based on 4D representation of a light field and ray tracing techniques to translate the frames and shift the depth range of the captured images. We calculated the ray direction and the perspective intersection point on the introduced frame while adjusting for the frame's changes in size, focal length, directional resolution and preserving the image quality. The proposed method is then tested on synthetic graphics that are pre-rendered into a light field image and tentatively on real-world light field images captured of a static scene using a controlled camera. We employ a simulator developed based on ray tracing techniques to visualize the images to evaluate the effectiveness and accuracy of our methods in producing high-quality light field frames applicable to virtual reality, augmented reality, and computational photography. The suggested solution aims to maintain the quality and realism of the translated frames while enhancing computational efficiency. This can further extend to more complex non-planar 4D representations of light fields and shifting the respective frames to achieve maximum optimization.

Acknowledgments

- I would like to express my deepest gratitude to my supervisor, Dr. Matthew Hamilton, for his invaluable guidance, continuous support, and patience during my thesis research. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life.
- I would also like to thank Nicholas Wells, for his insightful comments, support and encouragement which have greatly improved the quality of this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

The concept of a light field was popularized by Levoy and Hanrahan [1] when they proposed line parametrization by their intersections with two planes in arbitrary positions 1.1. This laid the groundwork for many subsequent advancements in light field technology.

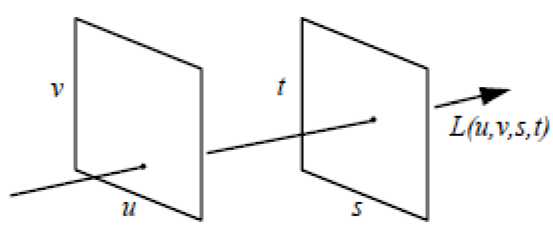

Figure 1.1: 4D representation of the light field (Levoy and Hanrahan [1]).

Building on this foundation, our project focuses on designing and implementing an algorithm for light field frame translation, a process that involves shifting the frame of reference within a light field representation to modify the depth and perspective of the captured scene, relative to its coordinate frame of reference. This translation technique utilizes ray tracing to accurately recalculate the intersection points and direction vectors of light rays, thereby generating new images from the adjusted frame and enabling optimization of depth resolution.

It is known that light field representations have limited resolution at depth from their planar reference frame [4]. This region of full resolution can be conceived of as a frustum (Figure 1.2). However, frustum regions often contain a significant amount of free space that cannot be efficiently removed through compression, leaving the front or back frustum empty of valuable data. Therefore, frame translation can be used to optimize rendering efficiency by concentrating resources on the relevant parts of the scene that need to be rendered, improving overall performance.

To evaluate our algorithm, we utilized light field images generated from high-quality virtual static scenes. These scenes were created using pre-rendered images produced by the advanced rendering engine Otoy's Octane.

We use an advanced simulator developed by Wells et al. [3], that enables dynamic generation of views from pre-rendered light field images and supports real-time ray tracing, allowing us to render scenes represented by light field images from multiple angles using light ray information.

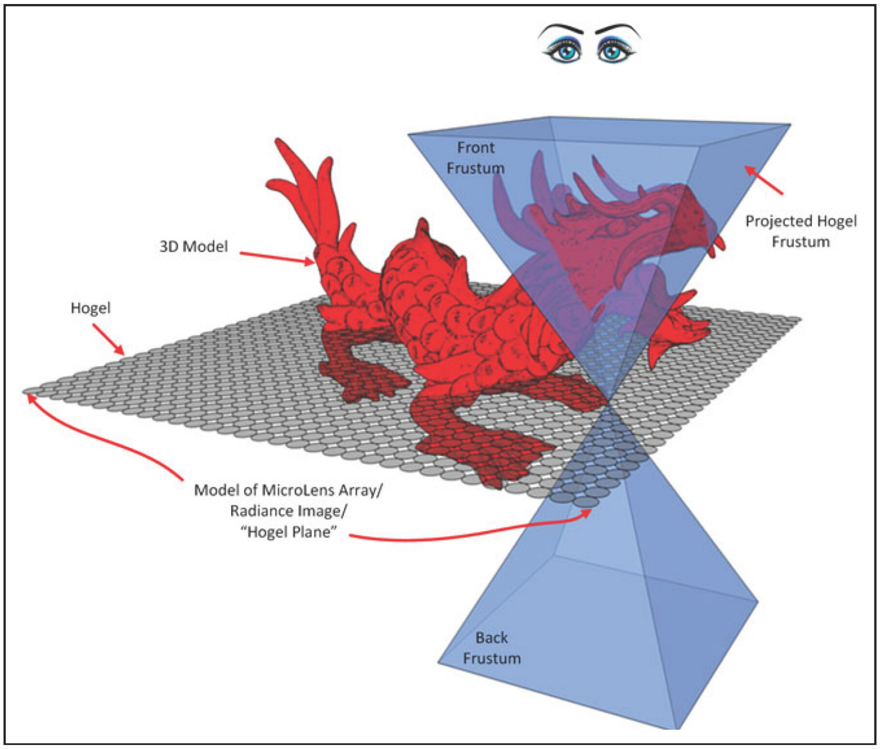

Figure 1.2: Representation of a double-frustum [2].

To evaluate the effectiveness and accuracy of our algorithm, we manually translated the original scene created in Maya and conducted a comparative analysis with our translated output (Figure 1.3). The results demonstrate that the translated frames preserve quality and realism, while the algorithm successfully shifts the depth range of the light field frames and translates them without introducing significant artifacts.

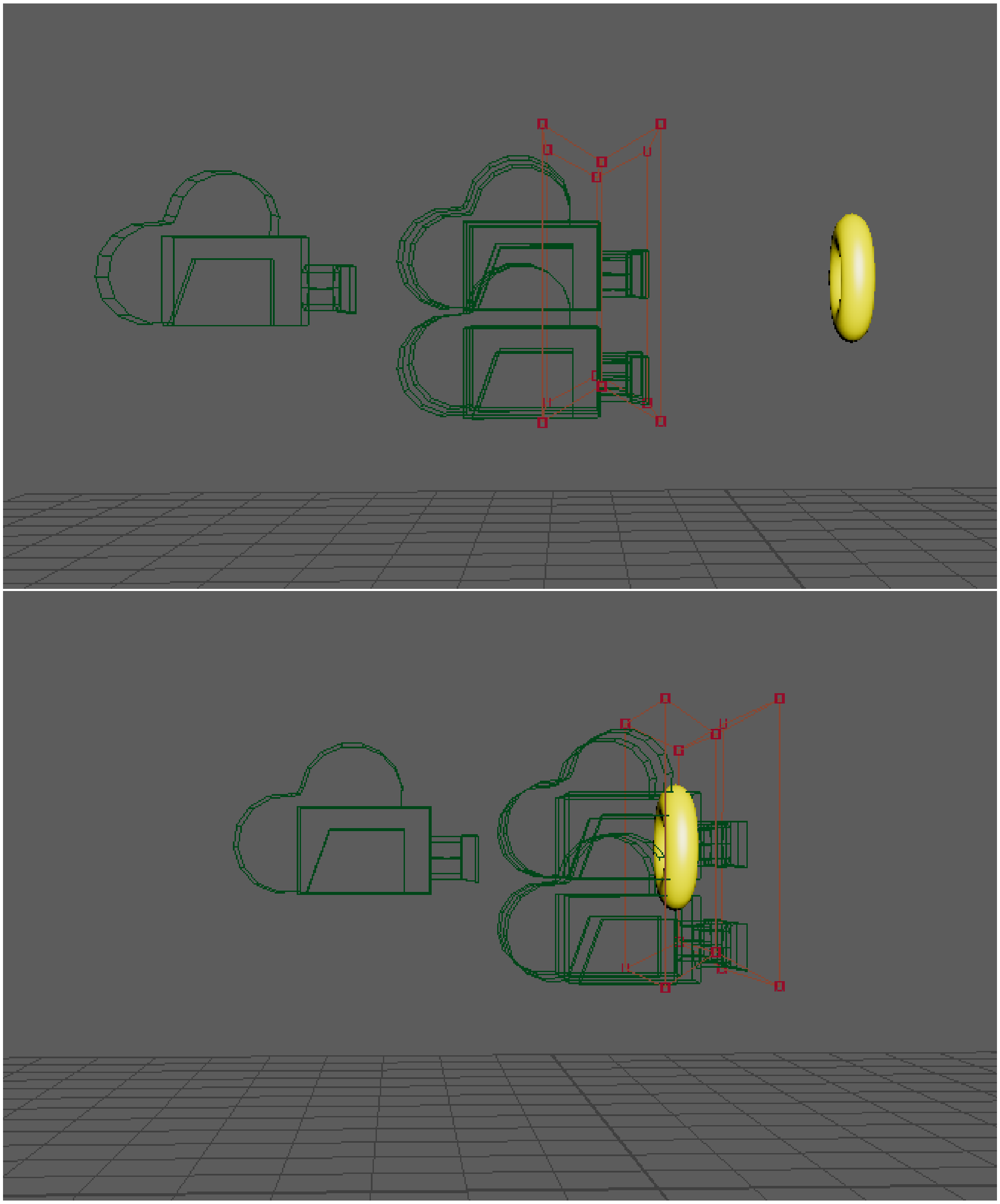

Figure 1.3: Visual Representation of Light Field Frame Translation Using Maya.

Chapter 2

Related Work

2.1 Related Work

2.2 Light field representation

The concept of light fields and rendering techniques has been covered in various research papers, significantly evolving over the years. Levoy and Hanrahan's [1] work pioneered the concept of light field rendering by introducing a robust representation of the light entering a camera (from multiple directions). Using this representation, the coordinate system on the first plane is denoted as (u, v) and on the second plane as (s, t), with light rays traced from a starting point on the first plane to a corresponding point on the second plane. Their approach treated the interaction of light with any object as a higher- dimensional matrix, effectively capturing the flow of light through unobstructed space. Levoy [5] expanded the theory and application of light fields, demonstrating that by capturing a large number of images from 4 various positions, it is possible to reconstruct new views of a scene. Building on their representation of the interaction of light with objects; this method, known as light field rendering, interprets the scene as a 4D array of pixels enabling the creation of accurate, perspective-shifted images by extracting the specifically oriented 2D slices from the 4D light field array, through pixel selection and interpolation. As highlighted by Geng [6], the plenoptic function also provides a foundational framework for understanding how light field displays reconstruct directional light rays to produce 3D images and the integration of physical depth cues is vital for ensuring realistic 3D perception in light field frame translation.

2.3 Light Field Sampling and Depth Resolution

Well-considered sampling can optimize the capture and reconstruction of light fields by balancing spatial and angular sampling rates. It ensures sufficient data is collected to avoid aliasing while accurately representing depth and perspective. Depth resolution can be defined as the ability to distinguish fine variations in spatial depth within a scene. Higher depth resolution enables clearer and more precise 3D visualizations. Zwicker et al [4] offered a unique perspective on sampling and depth resolution that is crucial for understanding light field optimization. By leveraging ray-space analysis, their work extends prior research on light fields and plenoptic sampling, emphasizing the role of the sampling grid in determining the display's depth resolution. This framework highlights the relationship between angular resolution which enables detail representation across viewing angles, and depth of field, providing insights into how sampling impacts the achievable depth resolution. Additionally, they propose resampling methods to adapt multi-view inputs for displays with limited resolution, showcasing how depth range can be adjusted to fit within the depth of field of the display. Borer [7] notes that the depth of field and image quality are governed by the number of elementary pixels in each hogel. A higher number of elementary pixels results in finer angular sampling, enabling sharper images across a broader depth range.

2.4 Light Field View Renderer and Display Simulator

A light field display simulator, developed by Wells [3], is a real-time ray tracing system designed for light field image view generation. It dynamically generates multiple views from pre-rendered light field images, enabling efficient scene rendering from various angles. This system provides a flexible and robust platform for exploring light field applications, including rendering, translation, and visualization tasks.

Built upon the work of Hamilton et al. [8], this design minimizes the computational intensity of a fully ray-traced solution while maintaining high accuracy in ray direction control, making it suitable for testing algorithms that require both precision and performance.

2.5 Summary

In summary, prior works on light fields, such as those by Levoy, Hanrahan, and Wells, have established critical concepts and methodologies for capturing, processing, and rendering light field images. Our project builds on these advancements, focusing specifically in optimizing light field frame translation.

Chapter 3

Background

3.1 Background

Light field frame translation is a computational method that shifts the frame of reference in a light field image to optimize depth resolution and perspective by recalculating light ray intersections and direction vectors using ray tracing, enabling efficient rendering and preserving image quality. This method relies on the 4D representation of light field frames [1], where parallel planes are spaced 1 unit apart, and ray tracing techniques are employed to calculate the directional vectors of light rays in the scene [5]. By shifting the plane along the z-axis, the method changes the frame's position. Ray tracing ensures accurate mapping and depth adjustments, making this technique effective for optimizing perspective and depth in light field images.

Chapter 4

Methodology

4.1 Preliminaries

A light field function $LF(u,v,s,t)$ maps a point (u,v) with direction (s,t) to a color value (R,G,B). A light field image captures the intensity of light traveling in multiple directions through every point on a 2D plane within 3D space. Each point on the plane, defined by coordinates (u,v), contains information about the intensity of light coming from multiple directions (s,t), referred to as hogels $H(s,t)$. Here, (u,v) represents a point on the light field plane, and (s,t) represents a direction in a 2D directional space.

The number of hogels in a light field image, denoted as NH, is a function of the pixel resolution (P_R) and the hogel resolution (H_R). Let $P_R = W \times H$, where W and H represent the width and height of the light field image in pixels, respectively, and let $H_R = A_x \times A_y$, where A_x and A_y represent the number of angular samples per hogel in the horizontal and vertical directions. The number of hogels can then be given by:

$$N_H = \frac{P_R}{H_R}$$

This formula shows that N_H is the ratio of the total number of pixels in the light field image to the number of angular samples per hogel. Thus, increasing angular resolution results in fewer hogels, whereas increasing pixel resolution results in more hogels.

Consider a plane associated with a light field $LF(u, v, s, t)$, situated within 3D space and located at $C = (C_x, C_y, C_z)$ where $(u, v) = (0, 0)$ and the plane being perpendicular with the vector along the z-axis, $(0, 0, 1)$, without loss of generality. Thus, we may map any (u, v) in the ambient space by $(C_x + u, C_y + v, C_z)$

We may consider light field frame translation in a restricted sense for practical purposes. We may limit shifting to the axis perpendicular to the plane (the z axis in this case), without loss of generality. Thus, we consider a plane located at $C = (C_x, C_y, C_z) + \Delta z$, for some shift amount Δz.

Light field frame translation then asks for us to re-sample the light field LF onto the new shifted plane, thus creating a shifted light field $LF_s(u, v, s, t)$. More formally, for a given (u, v, s, t) in LF, we must calculate a mapping $T(u, v, s, t) = (u_t, v_t, s_t, t_t)$ where (u_t, v_t, s_t, t_t) is the intersection of the ray located at $(C_x + u, C_y + v, C_z)$, directed via (s, t) and intersected with $(C_x + u_t, C_y + v_t, C_z + \Delta Z)$ and $LF(u, v, s, t) = LF_s(u_t, v_t, s_t, t_t)$.

4.2 Loading The Light Field

The first step in light field frame translation process involves loading the light field image and utilizing a JSON configuration file that contains the necessary attributes for the light field image required in ray tracing 4.1. This file must be loaded and parsed to extract the light field parameters, which will then be used to perform accurate ray tracing calculations. We initialize a numpy array uv' that represents the translated frame to store each calculated point P'(u,v). P' is the pixel position that is mapped from uv onto uv', and will be assigned the color values. The size of this array is set according to the new frame's hogel dimension and directional resolution. The size of this array is set according to the new frame's hogel dimension and directional resolution. The directional resolution D_R can be defined as a 2D resolution with two components:

$$D_R = (D_x, D_y)$$

where D_x and D_y represent the number of directional samples in the horizontal and vertical directions, respectively. This 2D representation allows for more explicit control and clarity in defining the resolution along each axis.

Each hogel position refers to a precise location on the image plane. The pixel position refers to the corresponding location in the final image visible to the user, where each hogel captures the rays of light contributing to a specific pixel. This creates the effect of viewing the scene from multiple angles [9].

To accurately calculate the rays of light associated with each hogel, we set the starting

```json
},
"simulatorAttributes": {
    "type": "lightfield",
    "position": [0,0,0],
    "orientation": [0,0,0],
    "physicalDimensions": [10,10],
    "sigma": 0.0067,
    "windowSize": 25
},
"lightFieldAttributes": {
    "spatialResolution": [200,200],
    "hogelDimensions": [200,200],
    "directionalResolution": [100,100],
    "displayFOV": [60,60],
    "file": "cameraSource.png"
}
```

Figure 4.1: Json file containing the required light field attributes.

point of all rays at the origin $(0, 0, 0)$, with the center of the image plane positioned at $(0, 0, -1)$ along the z-axis. Now we need to compute the position of the current pixel, CP, within the hogel in the image.

$$\mathrm{CP} = \mathrm{HL} \times \mathrm{PPH} + \mathrm{DR} \tag{4.1}$$

Where PPH is the number of pixels per hogel and HL represents the hogel location index. We need to scale the pixel position, SPP, in order to account for changes in focal length, FL, and physical dimension, PD. First, the pixel positions are normalized within the hogel, meaning the pixel's location is scaled to fit within a range of -1 to 1. This normalization ensures that the pixel positions are relative to the center of the hogel:

$$\mathrm{SPP} = \left(\frac{\mathrm{CP}}{\mathrm{PPH}}\right) \times 2 - 1 \tag{4.2}$$

With normalized pixel positions, ray mapping becomes simpler, as the pixel's location is now relative to the hogel's center. The ray tracing method in this case

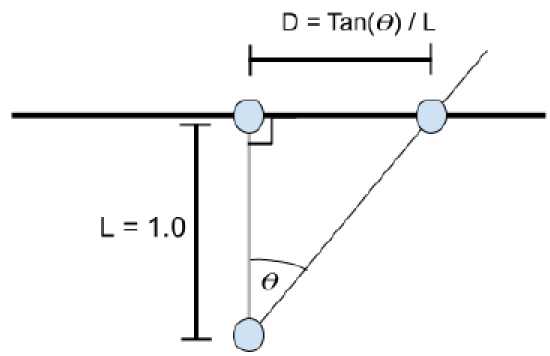

Figure 4.2: Angle to Vector Direction [3].

is based on a redefined field of view (FOV) representation, where a line and three points are used to describe the view . The focal length is normalized to 1, creating an angle equal to half of the field of view (FOV) between the two sides of the triangle 4.2. This normalization allows us to calculate the maximum distance, MD, from the center of the hogel to the edge of Field of View using the tangent.

$$\mathrm{MD} = \tan\left(\frac{\mathrm{FOV}}{2}\right) \times \mathrm{FL} \tag{4.3}$$

Once the maximum distance is calculated, the next step is to determine how far the scaled pixel lies from the center, DTC, of the field of view to accurately trace how light rays travel and intersects in the scene.

$$\mathrm{DTC} = \mathrm{SPP} \times \mathrm{MD} \tag{4.4}$$

This calculation is crucial for accurately tracing the rays and their intersections in the 3D scene. All these calculations must be performed for both the x-axis and y-axis to ensure accurate ray direction and perspective in the light field image.

4.3 Frame Translation

The goal of this step is to accurately translate the frame of reference, considering the new spatial resolution, directional resolution, field of view (FOV) and focal length adjustments. The method we use to calculate the distance from the center on the new plane is based on triangle properties and proportionality theorem. To integrate the concept of proportionality into the frame translation, the two sides of the triangle are considered to be the old and new focal length. The formula to compute the new distance from the center is:

$$DTC_N = \mathrm{DTC} \times \frac{FL_N}{\mathrm{FL}} \tag{4.5}$$

Where the N subscripts represent the translated frame distance to center and focal length. Calculating this distance for each hogel, results in a normalized value. To transform this into real-world light field image values, we must consider the actual physical dimensions and changes in the field of view (FOV). First, we compute the new maximum distance, MD, which represents the farthest distance a ray can travel from the center to reach the edge of the field of view (FOV) in the new frame:

$$MD_N = \tan\left(\frac{FOV_N}{2}\right) \times FL_N \tag{4.6}$$

To transform the pixel positions from the original frame to the new frame, we first need to find the in-hogel position, IHP. Now, scaling the old distance from the center (DTC) relative to the new maximum distance (MD_N) we get the Scaled In-Hogel Position, SIHP:

$$\mathrm{SIHP} = \frac{\mathrm{DTC}}{\mathrm{MD}_N}$$

This scaling ensures that the pixel's distance from the center of the original hogel is normalized to fit the new field of view (FOV) and hogel dimensions. The scaled position is then converted back into pixel coordinates within the hogel by adjusting it to the pixel grid:

$$\mathrm{IHP} = \frac{(\mathrm{SIHP} + 1)}{2} \times \mathrm{PPH}_N$$

- SIHP is a value normalized within the range $[-1, 1]$.
- PPH_N is the number of pixels per hogel in the new frame.

This process effectively maps the old pixel positions into the new frame, adapting to changes in field of view (FOV), hogel dimensions, and spatial resolution, while maintaining the relative positions of the pixels within each hogel. This mathematical transformation ensures that the light rays corresponding to each pixel are correctly aligned in the new frame.

To account for the physical dimension, we need to calculate each physical hogel location. The hogel pitch helps translate the pixel locations from the image coordinate system (in pixels) to the real-world physical coordinates. The hogel pitch is important in this process because it defines the spatial resolution of the light field in physical space for both the old and new images. Respectively we need to calculate these for the new frame and compute the old physical hogel location.

The Hogel Pitch, HPt, is defined as the ratio of the physical size of each hogel, HD, to the total physical dimension of the light field display, PD. This relationship is given by:

$$\text{HPt} = \frac{\text{HD}}{\text{PD}}$$

The New Hogel Pitch, HPt_N, is similarly defined as the ratio of the new hogel dimension, HD_N, to the total physical dimension of the new display (PD_N):

$$\text{HPt}_n = \frac{\text{HD}_n}{\text{PD}_N}$$

The Physical Hogel Location for the old frame, PHL, can be determined by multiplying the Hogel Pitch, HPt, by the hogel location, HL. This is expressed as:

$$\text{PHL} = \text{HPt} \times \text{HL}$$

These equations define the transformations required to compute the spatial properties of hogels in both the old and new light field frames, enabling accurate mapping of hogel locations between coordinate systems. Adjusting the physical location of hogels is done by adding the scaled distance from the center, represented in real-world values, to the old physical hogel location, resulting in the new physical hogel location, PHL_N, as:

$$\mathrm{PHL}_N = \mathrm{PHL} + \mathrm{DTC}_N$$

PHL represents the physical hogel location in the old frame, and DTC_N is the distance from the center of the new frame. The new hogel location, HL_N, is calculated by dividing the new physical hogel location by the new hogel pitch, HPt_N:

$$\mathrm{HL}_N = \frac{\mathrm{PHL}_N}{\mathrm{HPt}_N}$$

HPt_N refers to the physical pitch of each hogel in the new frame, determining the spacing of hogels in physical space. Finally, the new current pixel position, CPP_N, is determined by centering the hogel correctly and incorporating the in-hogel pixel position, IHPP_N:

$$\mathrm{CPP}_N = \mathrm{HL}_N \times \mathrm{PPH}_N + \mathrm{IHPP}_N$$

These transformations ensure that each pixel is accurately positioned in the new frame, maintaining spatial consistency and adapting to the updated frame parameters. This process ensures that each pixel in the new image aligns accurately with its corresponding position from the original image, adapting to the new physical dimensions, focal length, and field of view.

4.4 Light Field Reconstruction

After accurately shifting every hogel to its corresponding position on the new frame, the next step is to transfer the color information of each pixel and reconstruct the final image. This process ensures that the visual representation in the new frame is an accurate translation of the original light field image, with adjustments to accommodate the new dimensions and parameters. As mentioned before, we calculated the current pixel position on the old image in order to map the color values onto the final image. Now that the new pixel positions are determined based on the old location onto the new frame we can carry the color and brightness information from the original image to the new frame. For image reconstruction, we use the numpy array uv' as the canvas where the translated image will be drawn. The method described successfully performs the frame translation by:

1. Shifting the frame of reference from the original to the new one.

2. Maintaining the visual details, including the color information and pixel arrangement.

3. Adjusting the depth range of the light field, ensuring that the perception of depth and the 3D structure captured in the original image is preserved in the translated version. This process allows the light field image to be reconstructed in a new frame, accurately reflecting the changes in focal length, field of view, and hogel dimensions, while still maintaining the visual characteristics.

Chapter 5

Results

5.1 Results and Discussion

The result of translation is expected to be a shift in depth, creating a zoom-in or zoom-out effect on the object but not in the traditional sense. The frame translation algorithm accounts for changes in depth, perspective, directionality of light rays, and adjusts the focal length and field of view (FOV), which directly affects the perceived distance between the viewer and the objects in the scene. By changing the focal length, the light rays that hit the pixels in the image change in such a way that it mimics moving the observer's viewpoint closer to the objects in the scene. This creates a zoom effect or depth shift, as if physically moving towards the object.

The objects themselves remain stationary in the scene, but the point of view (the camera or virtual observer) shifts closer or further from them, altering how large or

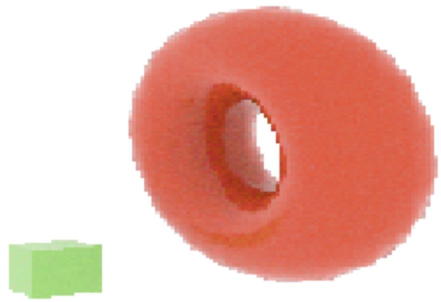

Figure 5.1: The synthetic scene created in Maya, used as original input for the frame translation process

small the objects appear. This means that objects will appear larger (if brought closer) or smaller (if moved further away), but the overall perspective and the directional light rays remain consistent with the original viewpoint.

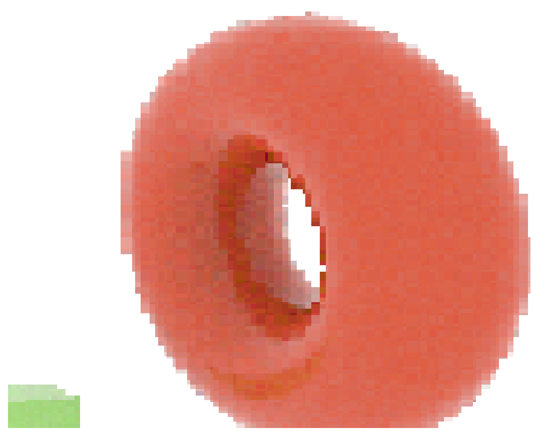

Figure 5.2: Manually translated light field in Maya.

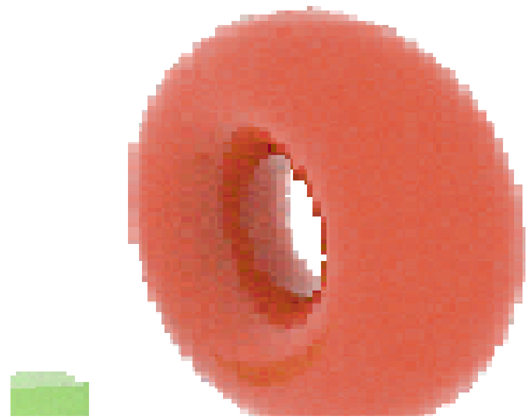

Figure 5.3: Translated light field image by 1 unit.

We have synthetically created a scene in Maya, where the object is initially viewed from a specific point of view. By adjusting the position of the light field camera within the scene, we can generate a frame translation and produce a visual representation of how the manually translated result should appear.

To verify our method, we utilized the real-time light field simulator created by Wells [2] that is based on advanced ray tracing techniques, allowing digital rendering of light field images in real-time. It offers the ability to view the light field from various perspectives, making it an ideal tool for visual comparison of the original and translated frames.

The accuracy of the generated image is also evaluated by calculating the Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE), and image difference comparison. As shown in figure 5.4, there is higher accuracy in high-frequency sampling regions, MSE of 24.54089381481 and PSNR of 34.23189984556272 indicates a moderate level of deviation between the ground truth and translated images.

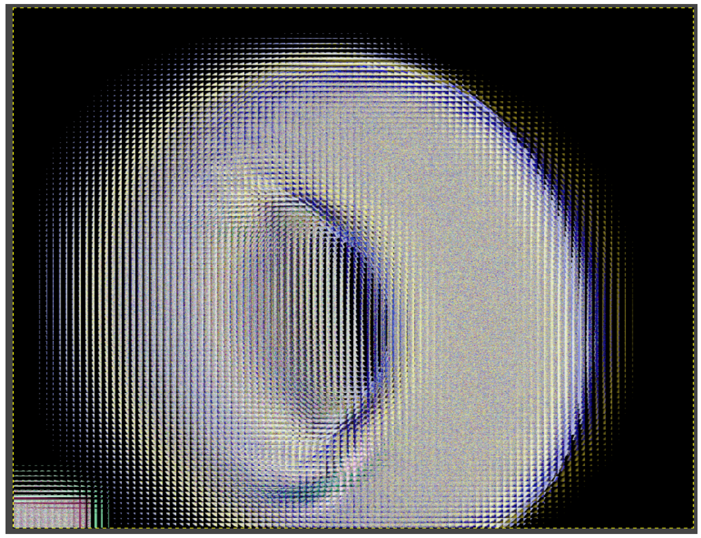

Figure 5.4: Comparison of the two integral images.

We observe that the translation algorithm accurately reproduces the expected result. The objects in the scene appear closer and larger in both the simulator and Maya, demonstrating the effectiveness of our translation method. Additionally, the quality and realism of the translated frames are maintained across both platforms, showcasing the precision of our approach in handling changes in focal length, field of view, and pixel positioning. These results demonstrate the applicability of our methods for generating moderate-quality light field frames, which can be extended to various fields such as virtual reality (VR), augmented reality (AR), and computational photography.

Table 5.1: Changes in light field parameters

	FOV (°)	Focal Length	Physical Dimension	Spatial Resolution	Hogel Dimensions	Directional Resolution
Original Image	60	1	10	200	200	60
Translated Image	30	2	6	100	100	30

Chapter 6

Conclusion

In this project, we successfully developed a method to perform frame translation of light field images with new parameters such as focal length, field of view (FOV), and hogel dimensions. Through careful pixel mapping and adjustments based on ray tracing principles, the translated frame and reconstructed light field image accurately reflects the original scene's structure at a different distance.

Using a synthetically created scene in Maya, we visually demonstrated the effect of shifting the point of view closer to the object and compared this with the output generated in a real-time light field simulator. The results from both platforms closely align, validating the accuracy of our approach. By translating the frame of reference for a light field image of a 3D scene, the 3D scene can often be imaged at a similar level of detail but with a light field image with a much lower directional resolution. This work opens possibilities for further exploration of light field image processing, including improved techniques for depth manipulation, perspective shifts, and broader applications in emerging visual technologies.

Chapter 7

Future Work

The next step in optimizing depth resolution in light field images involves using more complex, non-planar 4D representations of light fields instead of the traditional parallel plane-based 4D representation. By shifting to non-planar surfaces, we can more accurately align with the scene's geometry and improve the efficiency of frame translation.

As previously discussed, frustum regions in parallel plane representations often contain significant amounts of free space that cannot be efficiently compressed, leaving parts of the front or back frustum devoid of valuable data. Employing non-planar 4D surfaces reduces this free space, thereby enhancing rendering efficiency and achieving greater optimization of the light field data. This approach maximizes the potential for more precise depth resolution and overall image quality.

Bibliography

[1] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 31–42, New York, NY, USA, 1996. Association for Computing Machinery.

[2] Thomas Burnett. Light-field displays and extreme multiview rendering. *Information Display*, 33(6):6–32, 2017.

[3] Nicholas William Wells. Real-time ray traced cinematic quality light field viewer, 2021. Honours thesis, Memorial University of Newfoundland, Department of Computer Science.

[4] Matthias Zwicker, Wojciech Matusik, Frédo Durand, Hanspeter Pfister, and Clifton Forlines. Antialiasing for automultiscopic 3d displays. In *ACM SIGGRAPH 2006: Sketches, SIGGRAPH '06*, 2006.

[5] Marc Levoy. Light fields and computational imaging. *Computer*, 39, 2006.

[6] Jason Geng. Three-dimensional display technologies. *Adv. Opt. Photon.*, 5(4):456–535, Dec 2013.

[7] Tim Borer. Sampling requirements for lightfield displays with a large depth of field. 2019.

[8] Matthew Hamilton, Chuck Rumbolt, Thomas Butyn, Donovan Benoit, Robert Lockyer, and Matthew Troke. Light field display simulator for experience and quality evaluation. In *Digest of Technical Papers - SID International Symposium*, volume 49, 2018.

[9] Xingpeng Yan, Chenqing Wang, Yunpeng Liu, Xi Wang, Xinlei Liu, Tao Jing, Song Chen, Pei Li, and Xiaoyu Jiang. Implementation of the real–virtual 3d scene-fused full-parallax holographic stereogram. *Optics Express*, 29, 2021.