Medialogy Master Thesis
Computer Graphics
Thesis: MTA-161037
June 2016

Rendering the Light Field

User Evaluation of Light Field Rendering for Head Mounted Displays using Pixel Reprojection

Anne Juhler Hansen

Jákup Klein

Dept. Architecture, Design & Media Technology
Aalborg University
Rendsburggade 14, 9000 Aalborg, Denmark

This thesis is submitted to the Department of Architecture, Design & Media Technology at Aalborg University in fulfillment of the requirements for the degree of Master of Science in Medialogy - Computer Graphics.

Contact Information:
Authors:
Anne Juhler Hansen ajha11@student.aau.dk
Jákup Klein jklein11@student.aau.dk

Supervisor:
Prof. Martin Kraus martin@create.aau.dk
Department of Architecture, Design
& Media Technology

Dept. Architecture, Design & Media Technology
Aalborg University
Rendsburggade 14, 9000 Aalborg, Denmark

Web : http://www.aau.dk
Phone : +45 9940 9940
E-mail : aau@aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Department of Architecture, Design and Media Technology
Medialogy, 10th Semester

Title:

User Evaluation of Light Field Rendering for Head Mounted Displays using Pixel Reprojection

Project Period:

P10 Spring 2016

Semester Theme:

Master Thesis

Supervisors:

Martin Kraus

Project no.

MTA161037

Members:

Anne Juhler Hansen

Jákup Klein

Abstract:

Context. Render-optimization is an important phase of the light field rendering process to improve user experience while minimizing computational effort. It is concerned with CPU/GPU interplay, and requires both an understanding of optics and the equations behind the light field.
Objectives. In this study we investigate users evaluation of light field rendering for a head mounted display by comparing images created with different methods. Images made with high-precision but cost heavy methods are compared to images made with improved algorithms, in order to test users ability to perceive a difference.
Methods. We implement an improved method for light field rendering, where we instead of rendering all virtual cameras for each subimage instead render the four corner cameras and interpolate the rest of the views using pixel reprojection in the game engine Unity 3D.

Contents

Chapter 1

Introduction

Head mounted displays (HMDs) have developed increasingly in the last years, especially when looking at consumer markets and consumers' use of HMDs. One of the shortcomings and challenges of traditional HMDs is the lack of 3-dimensional cues, hereunder the parallax effect, and correct eye accommodation. Conflicting cues in vestibular and visual motion cues have been under suspicion of causing visual fatigue, eyestrain, diplopic vision, headaches, and other signs of motion sickness.

The future of virtual reality (VR) might be one in which visual discomfort and nausea can be eliminated, since a light field display can provide correct retinal blur, parallax and eye accommodation, which may balance out some of the conflicting cues which are experienced with traditional HMDs. The light field display allows for an observer to perceive a 2D image at different depths and angles by placing a distance-adjusted array of microlenses in front of a display. Nowadays, there is a strong trend towards rendering at high frame rates and to higher-resolution displays, and although graphics cards continue to evolve with an increasing amount of computational power, the speed gain is easily counteracted by increasingly complex and sophisticated computations. Computational complexity is also a concern for light field rendering, and especially when rendering for VR a relatively high frame rate is needed to avoid buggy implementations that creates noncontinuous motion and a bad user experience. When rendering for a light field display, several subimages have to be rendered from different views, as seen from an array of different cameras. Rendering all cameras is computational heavy, and this project investigates how rendering for a light field display does not necessarily imply a high workload. Instead of rendering high-cost high-precision virtual cameras, an array of views can be interpolated from only four rendered cameras. This project investigates methods that make use of this principle and provide practical and theoretical advice on how to use pixel reprojection for performance optimization.

(a) Title A

(b) Title B

Figure 1.1: Title for both

Chapter 2

Related Work

2.1 The Light Field

To understand the light field and its influence in computer graphics research, one must understand how to represent all light in a volume. The beginning of the light field and its definition can be traced back to Leonardo Da Vinci, who reffered to a set of light rays as radiant pyramids[1]: "The body of the air is full of an infinite number of radiant pyramids caused by the objects located in it. These pyramids intersect and interweave with each other during the independent passage throughout the air in which they are infused."

Later on the light field has been defined as the amount of light travelling in every direction through every point in space. Light can be interpreted as a field, because space is filled with an array of light rays at various intensities. This is close to the definition of the 5D plenoptic function, which describes all light information visible from a particular viewing position [1].

The plenoptic function allows reconstruction of every possible view, from every position, at every direction (See equation 2.1).

$$P(\theta, \phi, x, y, z) \tag{2.1}$$

Since radiance does not change along a line unless it is blocked, the 5D plenoptic function can be reduced to 4D in space free of occluders[2] (See figure 2.1).

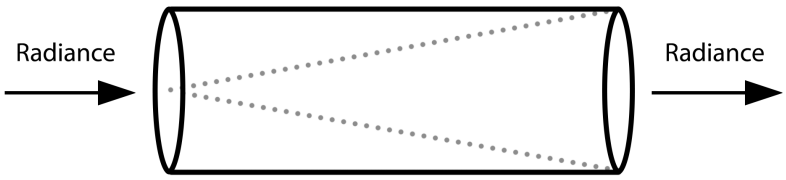

Figure 2.1: Radiance does not change along a line unless it is blocked, so in space free of occluders the 5D plenoptic function can be reduced to the 4D light field.

$$P'(\theta, \phi, u, v) \tag{2.2}$$

The 4D light field can explain the total light intensity and the direction of each ray as a function of position and direction (See equation 2.2).

2.1.1 Parameterization of the 4D Light Field

Levoy et al. described how a light field can be parameterized by the position of two points on two planes. This parameterization is called a light slab. A light ray enters one plane (the uv-plane) and exits another plane (the st-plane, which may be placed at infinity), and the result is a 2D array of images of a scene at different angles[2]. Since a 4D light field can be represented by a 2D array of images, it has the advantage that the geometric calculations are highly efficient. The line of all light rays can simply be parameterized by the two points.

When parameterizing the light field into 2D-images, the elemental images corresponds to images taken from different positions on the uv-plane, and each image represents a slice of the 4D light slab. In other words can the st-plane be thought of as a collection of perspective images of the scene, and the uv-plane corresponds to the position of the observer

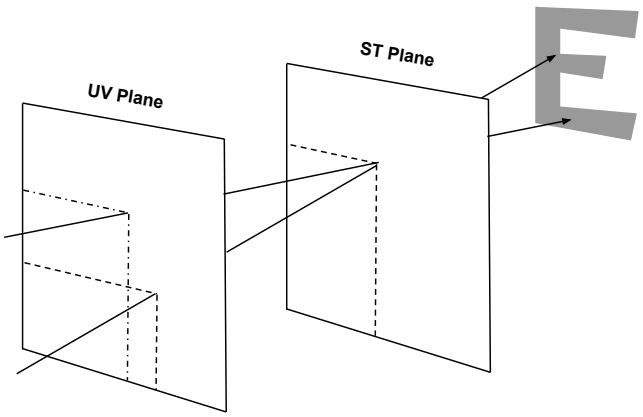

Figure 2.2: The light slab is a two-plane parameterization, where the st-plane can be thought of as a collection of perspective images of the scene, and the uv-plane corresponds to the position of the observer.

2.1.2 Capturing the Light Field

In light field photography a 2D representation of the 4D light field can be captured, and then sampled into a 2D image with a specified depth and angle within the limits of the stored light field. The light field can be captured in several ways; either with an array of cameras[3, 4], by moving a camera forward and backward[5], or by using a plenoptic camera containing an array of microlenses[6].

The first hand-held plenoptic camera that captures the 4D light field in one photographic exposure was created by Ng et al. [6]. The 4D light field is reconstructed into a 2D image in software post-capture, and can compute sharp photographs focused at different depths. In other words this method creates a synthetic aperture, that expands editing possibilities in post production by eliminating limitations related to a fixed aperture.

Naimark et al. created a stereo image capture rig, that captures a pair of stereo images[7]. From that a synthetic scene with depth could be calculated using cross dissolve. Since the light field gives a high accuracy of sampling it is possible to triangulate points into a point cloud, which provides the ability of tracking objects and semi-reconstruct objects and scenes 3-dimensionally.

This is one of the reasons why light field technology is gaining interest in the field of visual effect (VFX). Since light field photography essentially captures depth, it can be used to redefine previous methods (eg. chroma keying) and develop new approaches (eg. depth screen removal). Depth screen removal is one example of a new and improved technique for the VFX work flow, where the volumetric data from the light field can be used to disperse the object of interest from the background. The depth can be used to create semi-automated

segmentation and rotoscoping[8].

Interpolation strategies for optimizing resolution with light field photography are also being explored. Georgeiv et al.[9] have created an interpolation method that creates a better resolution in the final light field photograph, by virtually increasing the amount of views to be more than the amount of microlenslets. The Lytro Immerge is a new light field solution for cinematic virtual reality (VR), with a configurable dense light field camera array. With six degrees of freedom the solution allows virtual views to be generated with precise visual perspectives in a seamless capture that requires no stitching[10]. The goal of Lytro Immerge is to deliver lifelike presence for live action VR.

2.1.3 Optical Reconstruction

By reverse engineering the reconstruction of 2D images from light field capturing, the light field can be optically reconstructed by placing a distance-adjusted array of microlenses in front of a display (see Figure **??**). This is known as a light field display.

The light field display allows for an observer to integrate a correct 2D image of the light field at different depths and angles in accordance with the spatial and depth resolution that the light field contains. By using concepts from integral imaging displays and light field photography, Lanman and Luebke have introduced a light field display optimized for near eye viewing[11]. Every single lenslet of the full microlens array take in light from the screen, and light initially focused in the lens then gets reprojected, while containing the full amount of the angular information. This is equivalent to reconstructing individual images, where converging rays at the entrance pupil is divided by the number of pixels behind each microlens.

The result of rendering the light field is a holographic effect, and is being researched in many areas; 3D-display[12], light field projection[13], and holography[14]. The image seen through a light field display have focus cues, where the convergence point is the point in focus, and rest of the image blurs just like the real world. Even a monocular experience of the light field will give appropriate depth and focus cues, since the eye will converge behind the screen at the correct distance (see Section 2.3.1). Since distances can be virtually manipulated, the light field can be optically reconstructed to account for near- and far-sightedness in users.

2.2 Light Field Rendering

One of the first times light fields were introduced into computer graphics were by Levoy et al. in 1996, where they used image based rendering to compute new views of a scene from pre-existing views without the need for scene geometry[2].

The technique showed a real-time view of the light field, where it was possible to view around, see a scene with correct perspective and shading, and being able to zoom in and out. When zooming in, the light samples disperse throughout the array of 2D slices, so the perceived image is constructed from pieces from several elemental images.

Using direct light field rendering, Jeoung et al. has introduced an image-based rendering method in the light field domain[15]. The method attempts to directly compute only the necessary samples, and not the entire light field, to improve rendering in terms of complexity and memory usage. The algorithm consists of solving linear systems of two variables, requiring only limited complexities. Reducing complexity is highly desired when working with light fields, and (re)construction of overlapping views is a good place to start, since this is where light fields contain a lot of redundant information.

Much of the data is repetitive, especially when looking at a scene placed at infinity, where all sub images are created from parallel light rays. Instead of creating a virtual camera or capturing an individual sub image for each elemental image, interpolation can be used to reduce the computational effort. The interpolation between images will be more effective when knowing the depth of the three-dimensional geometry of the scene. Since light field rendering is one form of image-based rendering, a set of two-dimensional images of a scene can be used to generate a three-dimensional model. This way of obtaining depth is beneficial in the case of captured images, or if depth is not already easily obtained through the virtual 3D scene (see Section 2.1.2).

MORE ON RENDERING lacking computer power - can just be waited out, future development

2.2.1 Head-Mounted Light Field Displays

Head-Mounted Displays are still struggling with being heavy and having big and bulky optics[16]. Most HMDs do not account for the vergence-accommodation conflict (see Section 2.3.1), and they suffer from low resolution and a low field-of-view (FOV). Since light fields consist of more information than usual 2D images, light fields can improve on some of the limitations of traditional fixed-focus HMDs.

With the benefits from using microlenslet arrays in HMDs, Lanman et al. have shown that a light field display can be integrated into an HMD, which can both minimize the size of HMDs and potentially allow for much more immersive VR solutions compared to the fixed focus displays used in most common HMDs[11]. Near-eye light field displays have been created with a thickness of 1 cm.[17], and Shaulov et al. demonstrated that ultracompact imaging optical relay systems based on microlenslet arrays can be designed with an overall thickness of only a few millimeters[18].

Light field displays can also be created via stacked liquid crystal panels instead of microlenslets. Huang et al. have recently created a light field stereoscope

consisting of two semi-transparent LCD screens placed in series over a back-light[19]. With two separate 2-D images, one on top of the other, the light multiplicative recreates the entire light field. Simple images could be rendered at about 35 frames per second. Normally a minimum of 60 frames per second (fps) is what it takes to perceive convincingly smooth motion, and in HMDs the minimum frame rate is considered 90 fps.

The potential of the use of light field in VR goes beyond computer generated images. With an omnidirectional image capture, VR can integrate live-action footage. The Ricoh Theta is an omnidirectional camera that with two fish-eye lenses captures 360° with a single shot[20]. The captured images overlap, and can therefore be stitched together, taking every photo from that single point of view. A 360° spherical image will though only create a flat panorama in VR, and will get no 3D and parallax effect. A possible solution could consist of using many omnidirectional cameras, and putting one camera at every possible x, y, and z, in order to capture the 5D data set[21]. Since it will not be possible to have an infinite amount of omnidirectional cameras, it will be problematic to show continuous motion (eg. head tracking or when moving around). A workaround to the problem is to generate the in-between views using interpolation. With an interpolated omnidirectional light field capture it is then theoretically possible to implement footage from the real world into VR.

2.3 Visual Perception

In reality the human ocular system will adapt when focus is changed between different distances, such that the point of interest remains binocularly fused. Vergence and accommodation are parameters that influence our perception of depth and focus.

When accommodating the shape of the lens inside the eye changes to allow for a focused image at that distance. Humans can change the optical accommodation of their eyes by up to 15 diopters (the inverse of the focal length in meters), but the accommodation diversity is reduced with age[22].

The vergence mechanism continually adjusts the angle between the two eyes such that features at the focus distance remain fused in the binocular vision.

"As the distance of the point of interest decreases from infinity, the pair of eyes will converge along their vertical axis, or on the other hand, they will divergence when looking further away from a point. Both systems are driven by the logical cues relating to their area: retinal blur prompts an oculomotor accommodation adjustment, and stereo disparity drives the eyes to converge or diverge. Unfortunately for stereographic systems, there is a secondary set of cues for both systems consisting of reciprocal signals from one another; a feedback loop for both systems. Suryakumar et al. have shown that visual disparity in isolation elicits a fully comparable accommodation response to that of retinal blur[23],

strengthening the argument that these systems are very tightly coupled"[24].

2.3.1 Vergence-Accommodation Conflict

"In natural viewing conditions the reciprocal secondary cues between accommodation and vergence serve to better coordinate the final accommodative response[25]. However, in traditional stereo imaging where the depth is fixed, vergence towards a different distance will elicit irreconcilable cues between the two systems. This signal conflict has been linked to discomfort[26], visual fatigue, and reduced visual performance[27]. Research in resolving this conflict is still ongoing, with several proposals across the spectrum between hardware and software[28]"[24].

One of the consequent benefits of a light field display is that it allows for natural accommodation. The light field image can be rendered to be perceived as if they are at natural (or unnatural) distances away from the viewer. By adjusting eg. FOV, the depth of the optically reconstructed image will be influences and by taking advantage of this fact, the virtual distances can correct for near- and far-sightedness in users[11]. Accommodation at different distances determines which parts of the 2D image slices that are focused onto the retina.

Chapter 3

Implementation and methods

3.1 The Light Field Display

The head mounted near-eye light field display is constructed using an array of lenses (a Fresnel Technologies #630 microlens array) in front of a similar size adjusted array of rendered images (see Figure **??**).

Each of the lenslets in the microlens array can be seen as a simple magnifier for each of the elemental images in the array. Depicting the individual lenslets as a thin lens is though only an approximation, since the lenslets are influenced by parameters of a thick lens; curvature, its index of refraction, and its thickness. Since we are working with precision on micrometer scale there are many sources of error, and therefore our approach is designed as an empirical study, where we calculate with the lenslets as being thin lenses.

The lens separation d_l can then be found using the Gaussian thin lens formula (see Equation 3.1) where $0 < d_l \leq f$ and where d_l is the distance between the lens and the display, f is the focal length, d_0 is the distance to the virtual image, and d_e is the eye relief.

$$\frac{1}{f} = \frac{1}{d_l} - \frac{1}{d_0 - d_e} \Leftrightarrow d_l = \frac{f(d_0 - d_e)}{f + (d_0 - d_e)} \tag{3.1}$$

The lens separation d_l is one of the parameters in the formula with the greatest impact on the perceived image, since the microlens array should be placed at a distance $0 < d_l \leq f$. The lens separation should in most cases be ≈ 3.29mm or in other words just below the focal length $f = 3.3$ mm. With an eye relief of 35mm and d_0 set to 1 meter, then the lens separation $d_l = 3.2888$mm.

The lens separation was manually adjusted to the best possible alignment ≈3.29mm using a 3D printed spacer. Since the microlens array have a thickness of 3.3mm they had to be turned with the flat side up, which might as previously mentioned cause sources of error, since it is difficult to confirm the distance $d_l = 3.2888$mm (see Appendix **??**).

"The magnification factor tells us the magnification of the image on the screen to the image plane at d_0, and is used to calculate the field of view"[24]. With

$f = 3.3$mm and $d_0 = 1000$mm the magnification factor is M=293.42 (see Equation 3.2), where w_0 is the width of the virtual image at the plane of focus, and w_s is the width of the microdisplay.

$$M = \frac{w_0}{w_s} = \frac{d_0 - d_e}{d_l} = 1 + \left(\frac{d_0 - d_e}{f}\right) \tag{3.2}$$

3.2 Perceived resolution

The FOV is either limited by the extent of the lens (lens-limited magnifier) or it is limited by the dimensions of the display (display-limited magnifier). The lens-limited magnifier is influenced by $\frac{w_l}{2d_e}$, whereas the display-limited magnifier is influenced by $\frac{Mw_s}{2d_0}$, and since our FOV only can be limited by the lens (see Equation 3.3), we can then calculate the FOV for each of our virtual cameras in the array.

Field of view α (from the lens) per camera:

$$\alpha = 2\arctan\left(\frac{w_l}{2d_e}\right) \tag{3.3}$$

The FOV per rendered camera is then ?? 17.28°.
"Since a microlens array can be interpreted as a set of independent lens-limited magnifiers, the total field of view from the viewers eye can be found by substituting the lens separation d_l with the eye relief d_e, and the lens width w_l with the array width $N_l w_l$. The total FOV α_t is then given by Equation 3.4, where N_l is the number of lenses"[24].

$$\alpha_t = 2\arctan\left(\frac{N_l w_l}{2d_e}\right) \tag{3.4}$$

The vertical FOV for 15 lenses is ?? $FOV_v = 24.2°$ and the horizontal FOV for 8 lenses is ?? $FOV_h = 13°$ (see Equation 3.4).

Maximum spatial resolution N_p is given by:

$$N_p = \left(\frac{2d_0 \tan(\alpha/2)}{Mp}\right) \tag{3.5}$$

We get a spatial resolution of 120x64 ?? px (see Equation 3.5), but since α is expanded by the number of lenses N_l, and part of the rendered subimages are repeated across some or all of the elemental images, this repetition will reduce the perceived spatial resolution. Also, since the virtual cameras are quadratic, we either will have to cut off the top and bottom to fill the 15x8 ratio of the screen, or we will show the complete quadratic view plus extra views of the light field on the sides.

3.2.1 Depth of field

In ray optics focus appears at an image point, which is the point where light rays from the scene converge. The point is on the focus plane when it is in perfect focus. If the point is not on the focus plane the point will form a circle due to the light converging either in front or behind the image plane. This is called the circle of confusion. Due to the circle of confusion the focus plane is the only section of a scene being in focus. Since the size of the circle of confusion decreases (approaching zero) when a point approaches the focus plane, then any circle of confusion below the lowest level of detail that the system is able to distinguish will appear to be in focus. On a screen the smallest distinguishable detail is the pixel, so if the circle of confusion is equal or smaller to one pixel, the point shows the highest focus resolution that it can.

The circle of confusion c'_0 is therefore dependent on the optical characteristics that determine how the size of the circle of confusion changes over distance d'_0. Additionally the circle of confusion depends on the screen since the circle can not be smaller than a single pixel (see Equation 3.6, where p is the pixel pitch.) Note that the circle of confusion being calculated is not the circle of confusion on the image plane but rather on the focus plane.

$$c'_0 = \max\left(\left(\frac{d'_0 - d_0}{d_0 - d_e}\right) w_l, \left(\frac{d'_0 - d_e}{d_l}\right) p\right) \tag{3.6}$$

The depth of field is the area surrounding the focus point that appears to be in focus due the the circle of confusion being smaller that the smallest distinguishable detail (pixel pitch p). Figure 3.1 shows the two factors in the circle of confusion: actual circle of confusion from the lens, and the smallest detail possible due to pixel pitch. As long as the optical circle of confusion is smaller than a pixel the point appears to be in focus. In our setup the depth of field stretches from 22.9cm and continues to infinity.

3.2.2 Real and virtual distances

Through the Unity engine [29], a virtual image is rendered for every lenslet that is within the bounds of the microdisplay (see Appendix **??**), so the light field will be perceived as a holographic image with focus cues.

Each elemental image is rendered to a portion of the microdisplay; optimally 15mm × 8mm out of 15.36mm × 8.64mm to utilise most possible of the spatial resolution. The center of the elemental image should be calibrated to correspond to the center of the lenslet, and the virtual camera array should form a grid that would ideally be spaced with the same distance as that between each lenslet (1mm × 1mm). Any spacing is usable, as long as the relationship follows the physical lens-spacing in both axes. Scaling the grid spacing essentially scales the virtual world size accordingly. For our rendering engine we increase this grid by a factor

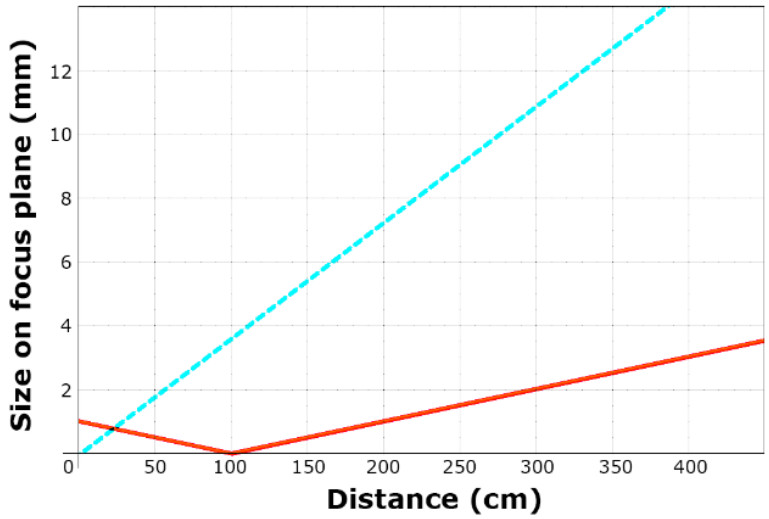

Figure 3.1: The red line shows the actual circle of confusion while the dotted cyan line shows the smallest detail that is possible to show (see Equation 3.6).

of 1000 to move the world further away from the nearest possible camera clipping plane. Object distances can be adjusted to correct for near- and far-sightedness

3.3 Rendering the Light Field

The light field is computed by extracting the two-dimensional slice from the 4D light field. Since the percieved image is constructed from pieces from several elemental images, we need to render all these elemental images in an array corresponding to the dimensions of our microlens array ($15x8$ lenlets). The secure and reliable solution would be to render $15x8$ different virtual cameras, where each camera have the same alignment as the lenlets. Since this method is so computational heavy that a standard laptop anno 2016 is not able to get an usable frame rate even when rendering low-cost scenes, our goal is to investigate how the computational effort can be reduced. Our approach is to render only the four corner cameras of the subimage array, and then interpolate between these four views in order to create all subimages of the light field. We will improve the real-time view of the light field, while maintaining correct perspective and shading, and investigate where short-comings of the interpolation might occur (eg. when objects are really close to the near clipping plane).

3.3.1 Image reprojection

To capture an image of a scene consisting of vertices in a 3D volume (world space), the vertices must be transformed to the cameras space (camera/eye space), where

Figure 3.2: Image reprojection

a 2D image with perspective distortion within near and far plane can be generated. When pixels are transformed to an image with a different pixel size, the result will be an approximation unless the pixel sizes are exact integers with no offset. Image reprojection involves the redistribution of information from a set of input pixels to a set of output pixels.

Using image reprojection the first step is to understand how to create an output image which is as close as possible to the rendered image of the virtual camera. The input pixel energy must be redistributed to the output pixel based on the exact overlap between these pixels. Not having a correct calculated image for one or more of the four corner cameras is a good way of debugging, since flaws in the corner views will always produce incorrect in-between views.

The interpolation of the subimages is accomplished using pixel reprojection, where the pixels from the corner images are copied to the corresponding place in the interpolated sub-image. To achieve this the pixel must be placed back to the 3D world and be "captured" to the interpolated sub-image. Here a projection matrix is used to convert from the view space to the image plane that will be displayed on the screen. The view space renders through a camera centered in the origin, hence view space is also referred to as camera space or eye space.

The transformation goes back and forward between the projection plane (the 2D generated image) and the eye/camera space (the 3D scene with the camera as the center). All sub-views (interpolated cameras) have an individual position in world space, and need to do a transformation between these spaces in order to generate the interpolated subimages. If the projection plane for one camera and the transformation matrix in relation to the other camera is known, then the pixels can be reprojected to the other camera.

Finally the view space is projected onto the 2D screen, where near and far

clipping plane are obtained via frustum culling (clipping), and the clip coordinates are transformed to the normalized device coordinates.

The clip coordinate system projects all vertex data from the view space to the clip coordinates by comparing x_{clip}, y_{clip}, and z_{clip} with w_{clip}. Any clip coordinate vertex that is less than $-w_{clip}$ or greater than w_{clip} will be discarded, and then the clipping occurs.

The x-coordinate of eye space, x_e is mapped to x_p, which is calculated by using the ratio of similar triangles (see Equation **??**).

$$x_e = \frac{x_p \cdot z_e}{n} \tag{3.7}$$

Likewise the transformation from eye/camera space to the projection plane are influenced by the position in eye space x_e, the position on the projection plane x_p, the near clipping plane n, and the depth z_e (see Equation **??**).

$$x_p = \frac{n \cdot x_e}{z_e} \tag{3.8}$$

3.3.2 Depth

We need the depth information of the scene to effectively interpolate between the images. Using perspective projection the relation between z_e and z_n is non-linear, which means high precision at near plane and little precision at the far plane. We need to account for the non-linear relationship (from vertex position in object space to vertex position in clip space) to get correct distances and depth in a normalized [0,1] range[30].

Calculation of depth (See Equation 3.9).

$$z_e = \frac{x - 0}{1 - 0} \cdot (f_c - n_c) + n \tag{3.9}$$

When saving the depth information of the subimages, 8 bit depth is not enough, since we will loose a lot of important information. 32 bit (float32) has more precision, and is sufficient enough for saving the depth. Using the method with four corner cameras, we can expect to see fewer mistakes in the corners, and several mistakes elsewhere because of the interpolation of the in-between images. When rendering objects close to near clipping plane, we might experience holes, where neither of the four cameras can see.

3.3.3 Shader programming

The pixel reprojection was programmed with CG pixel shaders in the Unity3D game engine. Pixel shaders can be executed fast while performing hundreds to

thousands of arithmetic operations. Intrinsic parallelism in the code can be better exploited with advancement in compiler technology.

We have chosen to divide the interpolation into three steps: interpolation of the x-axis, interpolation of the y-axis and at down-scaling (and thereby a little bit of antialiasing) the image to the resolution of the display ($1280x720$px). Three shaders are easy to debug, since x-axis and y-axis can be controlled one by one, but is ineffective since the steps must be send back and forward between the Graphics processing unit (GPU) and the Central processing unit (CPU).

The interpolation is implemented as an image postprocessing effect. Unity3D work with image effects as scripts attached to a camera to alter the rendered output, and the computation is done in the shader scripts[31]. The rendered images can be accessed via render textures, which are created and updated at runtime[32].

Since shaders works better with math calculations than logic, nested for-loops and if-statements can make the compiler crash, and should therefore be avoided or their use reduced. Cg has built-in functions that will work as good alternatives: The lerp function returns the linear interpolation of a and b based on weight w (return a when $w = 0$, return b when $w = 1$). The step function returns one for each component of x that is greater than or equal to the corresponding component in the reference vector (return 1 when $x >= a$, return 0 when $x < a$).

3.3.4 Stereoscopic Rendering

Stereoscopic rendering simulates the natural stereoscopy of the human eyes by rendering two images representing the view of each eye eg. the left part of the screen is used for the left eye and vice versa[33]. The Quad-buffer in OpenGL is a double-buffer side-by-side method that uses 100% of the display resolution. Simply splitting the screen will reduce the perceived resolution to only 50%.

Chapter 4

Experiments

4.1 User test

This experiment wants to statistically compare if subjects can discriminate between the image with virtual cameras (VC) and the interpolated image (II), but since usual statistical hypothesis tests can test for a significant difference in population, this complies with true hypothesis testing of rejecting the null hypothesis.

4.1.1 Two-alternatives forced choice test

A forced choice test is one that requires the test participant to identify a stimulus by choosing between a finite number of alternatives, usually two. With two possible choices this is referred to as a two-alternative forced choice (2AFC) procedure. [34] If the test subjects can do no better than a random guess then the test is passed, meaning that we can conclude that the test participants experience no difference between VC and II.

The 2AFC tasks are:

1. two alternative choices presented in random order (e.g. two visual stimuli)
2. a delay interval to allow a response
3. a response indicating choice of one of the images

The probability of mass function:

$$f(i; n, pnull) = \frac{n!}{x!(n-i)!} p^x (1 - p_{null})^{n-x} \tag{4.1}$$

where i is the number of incorrect answers, p_{null} is the probability of II incorrectly identified as VC, and n is the number of trials.

$$i_c(n, p_{null}) = min\{i | \sum_{n}^{j=i} f(j|n, p_{null}) < 0.05\} \tag{4.2}$$

4.2 Technical test

pixel match evaluating the pixel colour difference

Chapter 5

Results and Analysis

5.1 Results

5.2 Analysis

Chapter 6

Conclusions and Future Work

6.1 Conclusion

6.2 Future Work

Bibliography

[1] Edward H Adelson and James R Bergen. The plenoptic function and the elements of early vision. *Computational models of visual processing*, 1(2): 3–20, 1991.

[2] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42. ACM, 1996.

[3] Bennett S Wilburn, Michal Smulski, Hsiao-Heng K Lee, and Mark A Horowitz. Light field video camera. In *Electronic Imaging 2002*, pages 29–36. International Society for Optics and Photonics, 2001.

[4] Hartmut Schirmacher, Li Ming, and Hans-Peter Seidel. On-the-fly processing of generalized lumigraphs. In *Computer Graphics Forum*, volume 20, pages 165–174. Wiley Online Library, 2001.

[5] Cha Zhang and Tsuhan Chen. Light field capturing with lensless cameras. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–792. IEEE, 2005.

[6] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, and Pat Hanrahan. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report CSTR*, 2(11), 2005.

[7] Michael Naimark, John Woodfill, Paul Debevec, and Leo Villareal. Immersion'94. *Interval Research Corporation image-based modeling and rendering project from Summer*, 1994.

[8] Jon Karafin. Light field imaging: The future of vr-ar-mr - part 4, 11 2015. URL `https://www.youtube.com/watch?v=_PVok9nUxME`. Presented by the VES Vision Commitee. Presentation by Jon Karafin, Head of Light Field Video for Lytro, followed by a QA with all presenters moderated by Scott Squires, VES. [Accessed June 1, 2016].

[9] Todor Georgiev, Ke Colin Zheng, Brian Curless, David Salesin, Shree Nayar, and Chintan Intwala. Spatio-angular resolution tradeoffs in integral photography. *Rendering Techniques*, 2006:263–272, 2006.

[10] https://www.lytro.com/press/releases/lytro-immerge-the-worlds-first-professional-light-field-solution-for-cinematic-vr. Accessed June 1, 2016.

[11] Douglas Lanman and David Luebke. Near-eye light field displays. *ACM Transactions on Graphics (TOG)*, 32(6):220, 2013.

[12] Gordon Wetzstein, Douglas Lanman, Matthew Hirsch, and Ramesh Raskar. Tensor displays: compressive light field synthesis using multilayer displays with directional backlighting. *ACM Trans. Graph.*, 31(4):80, 2012.

[13] Matthew Hirsch, Gordon Wetzstein, and Ramesh Raskar. A compressive light field projection system. *ACM Transactions on Graphics (TOG)*, 33(4): 58, 2014.

[14] Andrew Jones, Ian McDowall, Hideshi Yamada, Mark Bolas, and Paul Debevec. Rendering for an interactive 360 light field display. *ACM Transactions on Graphics (TOG)*, 26(3):40, 2007.

[15] Young Ju Jeong, Hyun Sung Chang, Yang Ho Cho, Dongkyung Nam, and C-C Jay Kuo. Efficient direct light field rendering for autostereoscopic 3d displays.

[16] JP Rolland and Hong Hua. Head-mounted display systems. *Encyclopedia of optical engineering*, pages 1–13, 2005.

[17] Douglas Lanman and David Luebke. Supplementary material: Near-eye light field displays. *ACM Transactions on Graphics (TOG)*, 32(6):220, 2013.

[18] Vesselin Shaoulov, Ricardo Martins, and Jannick P Rolland. Compact microlenslet-array-based magnifier. *Optics Letters*, 29(7):709–711, 2004.

[19] Fu-Chung Huang, Kevin Chen, and Gordon Wetzstein. The light field stereoscope: immersive computer graphics via factored near-eye light field displays with focus cues. *ACM Transactions on Graphics (TOG)*, 34(4):60, 2015.

[20] https://theta360.com/en/about/theta/. Accessed June 1, 2016.

[21] Paul Debevec. Light field imaging: The future of vr-ar-mr - part 1, 11 2015. URL `https://www.youtube.com/watch?v=Raw-VVmaXbg`. Presented by the VES Vision Committee. Introductions by Vision Committee Chair, Toni Pace Carstensen and Moderator, Scott Squires, VES followed by the first presentation by Paul Debevec, Chief Visual Officer, USC Institute for Creative Technologies.[Accessed June 1, 2016].

[22] W Neil Charman. The eye in focus: accommodation and presbyopia. *Clinical and experimental optometry*, 91(3):207–225, 2008.

[23] Rajaraman Suryakumar, Jason P Meyers, Elizabeth L Irving, and William R Bobier. Vergence accommodation and monocular closed loop blur accommodation have similar dynamic characteristics. *Vision research*, 47(3):327–337, 2007.

[24] Jon Aschberg, Jákup Klein, and Anne Juhler Hansen. Experimental evaluation of the perceived accommodation range of a near-eye light-field display. 2015.

[25] Clifton M Schor, Jack Alexander, Lawrence Cormack, and Scott Stevenson. Negative feedback control model of proximal convergence and accommodation. *Ophthalmic and Physiological Optics*, 12(3):307–318, 1992.

[26] Takashi Shibata, Joohwan Kim, David M Hoffman, and Martin S Banks. Visual discomfort with stereo displays: Effects of viewing distance and direction of vergence-accommodation conflict. In *IS&T/SPIE Electronic Imaging*, pages 78630P–78630P. International Society for Optics and Photonics, 2011.

[27] David M Hoffman, Ahna R Girshick, Kurt Akeley, and Martin S Banks. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of vision*, 8(3):33, 2008.

[28] Gregory Kramida and Amitabh Varshney. Resolving the vergence-accommodation conflict in head mounted displays.

[29] http://unity3d.com/5?gclid=CObkm4rW4MUCFYUM cwod4C0ACg. Accessed June 1, 2016.

[30] http://beta.unity3d.com/talks/Siggraph2011$_{S}pecialEffectsWithDepth_{W}ithNotes.pdf.Acces$

[31] http://docs.unity3d.com/Manual/WritingImageEffects.html. Accessed June 1, 2016.

[32] http://docs.unity3d.com/Manual/class-RenderTexture.html. Accessed June 1, 2016.

[33] http://docs.unity3d.com/Manual/StereoscopicRendering.html. Accessed June 1, 2016.

[34] Suzanne P McKee, Stanley A Klein, and Davida Y Teller. Statistical properties of forced-choice psychometric functions: Implications of probit analysis. *Perception & Psychophysics*, 37(4):286–298, 1985.