DOI:10.15774/PPKE.ITK.2015.005

RENDERING AND INTERACTION ON PROJECTION-BASED LIGHT FIELD DISPLAYS

Theses of the Ph.D. dissertation

Vamsi Kiran Adhikarla

Scientific adviser:

Péter Szolgay, D.Sc.

Pázmány Péter Catholic University

Faculty of Information Technology and Bionics

Roska Tamás Doctoral School of Sciences and Technology

Budapest 2015

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

Abstract

Current generation glasses-free 3D displays strive to present a 3D scene from multiple viewing positions. Many efforts have been made in recent past to make the discrete viewing positions as continuous as possible as it increases the degree of reality in displaying. Deploying a hologram, projection-based light field displays provide a *photorealistic* means to display a 3D scene and overcome the shortcomings of other known glasses-free 3D displays. With increased *field of view* (FOV) and more closer viewing positions/angles, these displays allow multiple users to freely move in front of the display to perceive 3D without any obstacles. Projection-based light field displays, therefore, have a high potential to be the future 3D display solution and novel methods to process the visual data required by such displays are highly needed. The main aim of this work is to explore this visual data to enable deeper understanding and efficient representation. In particular, the work investigates various rendering techniques for light field displays and proposes the requirements for future technologies that deal with light field rendering, distribution and interaction in the context of projection-based light field displays. Using 3D computer graphics, it is possible to acquire and render light rays from any given arbitrary positions to define a light field. However, a stable means to acquire, process and render light field from real-world scenes that meet the requirements of such light field displays in real-time are not fully available yet. The current work outlines the available real-time rendering procedures for light field displays from *multiview* content and presents the possible directions for improvements. In addition, the work also reports on interaction experiments dealing with interactive visualization in synthetic environment. This work has applications in free view point television, tele-presence, virtual reality and video conferencing systems.

Keywords: Light field displays, HoloVizio, 3DTV, Multiview, telepresence, Real-time rendering, Collaborative virtual environments, 3D Interaction, Leap Motion, Human Computer Interaction, On-the- y depth retargeting, GPU, Multiprojector light field display, Visually enhanced live 3D Video, Multi-view capture and display

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

Acknowledgments

This research has received funding from the DIVA Marie Curie Action of the People programme of the EU FP7/2007- 2013/ Program under REA grant agreement 290227. The support of the TAMOP-4.2.1.B-11/2/KMR-2011-0002 is kindly acknowledged.

This dissertation work is a result of three years of research and development project that has been done in collaboration with Holografika and Pázmány Péter Catholic University, Hungary. It is a delight to express my gratitude to all, who contributed directly and indirectly to the successful completion of this work.

I am very grateful to Tibor Balogh, CEO and founder of Holografika Ltd., Zsuzsa Dobrányi, sales manager, Holografika Ltd., Péter Tamás Kovács, CTO, Holografika Ltd., and all the staff of Holografika for introducing Hungary to me and providing me an opportunity to work with a promising and leading edge technology - the HoloVizio.

My special thanks and sincere appreciation to Attila Barsi, lead software developer at Holografika for holding my hand and walking me through this journey. I would like to further extend my thanks to Péter Tamás Kovács for his unconditional support and supervision despite his hectic schedule. Their mentoring and invaluable knowledge is an important source to this research and so to this dissertation.

I am grateful to Prof. Péter Szolgay, Dean, Pázmány Péter Catholic University for letting me to start my Ph.D. studies at the doctoral school. I thankfully acknowledge him for his guidance and support throughout the work. His precious comments and suggestions greatly contributed to the quality of the work.

I am much indebted to all the members of Visual Computing group, CRS4, Italy and in particular to Enrico Gobbetti, director and Fabio Marton, researcher, CRS4 Visual Computing group for their exceptional scientific supervision. The work would not be complete without their guidance. I feel very lucky to have had the opportunity to work with them.

I am thankful to Jaka Sodnik and Grega Jakus from University of Ljubljana for sharing their knowledge and valuable ideas.

I would like to show my love and record my special thanks to my parents and to my sister who kept me motivated during my work with their unconditional support.

I would like to express my deep love and appreciation to my beautiful Niki for her patience, understanding and for being with me through hard times.

Finally, I would like to thank everybody who was significant to the successful understanding of this thesis, as well as expressing my apology that I could not mention personally one by one.

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

Contents

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

List of Figures

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

List of Tables

DOI:10.15774/PPKE.ITK.2015.005

DOI:10.15774/PPKE.ITK.2015.005

Abbreviations

2D	Two-Dimensional
3D	Three-Dimensional
3DTV	Three-Dimensional Television
3DV	Three-Dimensional Video
AVC	Advanced Video Coding
CPU	Central Processing Unit
DIBR	Depth Image Based Rendering
DPB	Decoded Picture Buffer
FBO	Frame Buffer Object
FOV	Field Of View
fps	frames per second
FTV	Free-viewpoint Television
GPU	Graphics Processing Unit
GLSL	OpenGL Shading Language
HD	High Definition
JPEG	Joint Photographic Experts Group
LDV	Layered Depth Video
LDI	Layered Depth Images
LF	Light Field
LFD	Light Field Display
LMC	Leap Motion Controller
LOD	Level Of Detail
MPEG	Moving Picture Experts Group
MVC	Multiview Video Coding
MVD	Multiview plus Depth
MVV	MultiView Video
OpenGL	Open Graphics Library
OpenCV	Open Source Computer Vision
PSNR	Peak Signal-to-Noise Ratio
RAM	Random Access Memory
RGB	Red, Green, Blue
SSIM	Structure SIMilarity index
ToF	Time-of-Flight
V+D	Video plus Depth
WXGA	Wide eXtended Graphics Array

DOI:10.15774/PPKE.ITK.2015.005

Chapter 1

Introduction

1.1 Preface

The term *light field* is mathematically defined as a *plenoptic function* that describes the amount of light fared in all directions from every single point in space at a given time instance [1]. It is a seven dimensional function as described in the following equation:

$$Lightfield = PF(\theta, \phi, \lambda, t, V_x, V_y, V_z) \tag{1.1}$$

where (V_x, V_y, V_z) is the location of point in 3D space, θ,ϕ are the angles determining the directions, λ is the wavelength of light and t is the time instance. Given a scene, the plenoptic function using light rays, describes the various objects in a scene over a continuous time. Thus the plenoptic function parameterizes the light field mathematically which enables the understanding of the processes of capturing and displaying. Precisely presenting a 3D scene involves capturing of the light wavelength at all points in all directions, at all time instances and displaying this captured information. However, it is not possible in reality due of practical limitations such as complex capturing procedure, enormous amount of data in every single time instant, unavailability of means to display the smooth and continuous light field information. Without loss in the generality, few assumptions and simplifications can be made to reduce the dimensions of the plenoptic function and proceed further with light field representations:

- Light wavelength (λ) can be represented in terms of Red, Green and Blue (RGB).
- Considering static scenes reduces the dimension of time (t) and a series of such static scenes can later constitute a video.
- The current means of displaying the captured plenoptic data involves defining the light rays through a two dimensional surface (screen) and thus we may not need to capture the light field over 360 degree *field of view*.

DOI:10.15774/PPKE.ITK.2015.005

- For more convenience we can use discrete instead of continuous values for all parameters (sampling).
- A final assumption involves considering that air is transparent and radiance along any light ray is constant.

After the simplifications and assumptions, we can represent the light field as a function of 4 variables:

(x,y) - location on a 2D plane &
(θ, ϕ) - angles defining the ray direction.

In practice, most of the existing 3D displays produce horizontal only parallax and thus sampling the light rays along one direction is valid. Varying the parameters remaining after these simplifications, several attempts were made for simplifying and displaying slices of light field.

1.1.1 Displaying Simplified Light Field

Stereoscopy which was invented in early 18th century showed that when two pictures captured at slightly different viewpoints are presented to two eyes separately, they are combined by the brain to produce 3D depth perception. With the growing usage of Television in 1950s a variety of techniques to produce motion 3D pictures have been proposed. The main idea is to provide the user with lenses/filters that isolate the views to left and right eyes. Some of the popular technologies that are still in practice today are:

- **Anaglyph 3D system** Image separation using color coding. Image sequences are preprocessed using a distinguished color coding (typical colors include Red/Green, Blue /Cyan) and users are provided with corresponding color filters (glasses) to separate the left/right views.
- **Polarized 3D system** Image separation using light polarization. Two images from two projectors filtered by different polarization lenses are projected on to the same screen. Users wear a pair of glasses with corresponding polarization filters.
- **Active shutter 3D system** Image separation using high frequency projection mechanism. Left and right views are displayed alternatively. User is provided with active shutter glasses. One view is presented to the first eye blocking the second eye and a next view is presented immediately after it to the second eye blocking the first eye. This is done at very high frequency to support continuous 3D perception.

With rapid growth in the communication technology during the second half of 19th century, it became possible to transmit huge amount of video information to the remote users, for e.g., High

DOI:10.15774/PPKE.ITK.2015.005

Definition (HD) video. Stereoscopic imaging in HD has emerged out to be the most stable 3D displaying technology in the entertainment market during recent times. But still the user has to wear glasses to perceive 3D in a stereoscopic setting.

In a glasses free system, the process of view isolation has to be part of display hardware and such displays are, therefore, generally called *autostereoscopic displays*. To achieve the separation of views, the intensity and color of emitted light from every single pixel on the display should be a function of direction. Also, to appeal to a variety of sectors especially the design industry, it is needed to support additional depth cues such as *motion parallax* which enables the looking behind experience. *Parallax* is a displacement in the apparent position of an object viewed along two different lines of sight. Aligning multiple stereoscopic views as a function of direction produces the required parallax and will lead us to more realistic 3D experience.

Autostereoscopic display technology incorporating lens arrays was introduced in 1985 to address motion parallax in horizontal direction. In an autostereoscopic display, the view isolation is done by the lens arrangement and hence the user need not possess any additional eye wear. By properly aligning the lens arrays it is possible to transmit/block light in different directions. A drawback of this approach is that the user sees the light barrier from all the viewpoints in the form of thin black lines. Due to the physical constraints of lens size, there are viewing zones where the user sees a comfortable 3D and the transition from one viewing zone to other is not smooth. So, users should choose in between the possible viewing positions. In addition the *Field Of View* (FOV) of autostereoscopic displays with lens arrays is narrow.

A different approach to produce 3D perception is introduced in the *volumetric displays*. Rather than simulating the depth perception using motion parallax, these devices try to create a 3D volume in a given area. They use time and space multiplexing to produce depth. For example a series of LEDs attached to a constantly moving surface and producing different patterns that mimic the depth slices of a volume to give an illusion of volume. A main problem with this approach is that due to the mechanically moving parts, it is hard to avoid micro motions in the visualization and depending on the complexity it may not possible to produce a totally stable volume.

Apart from above, there were few other approaches such as : head mount displays and displays based on motion tracking and spatial multiplexing etc., that reduce the number of dimensions of the light field function to derive a discrete segment of light field [2]. However, practical means to produce highly realistic light field with *continuous-like* depth cues for 3D perception are still unavailable.

1.1.2 Projection-based Light Field Display

A more pragmatic and elegant approach for presenting a light field along the lines of its actual definition has been pioneered by HoloVizio: projection-based light field displaying technology [3] [4] [5]. Taking inspiration from the real-world, a projection-based light field display emits

DOI:10.15774/PPKE.ITK.2015.005

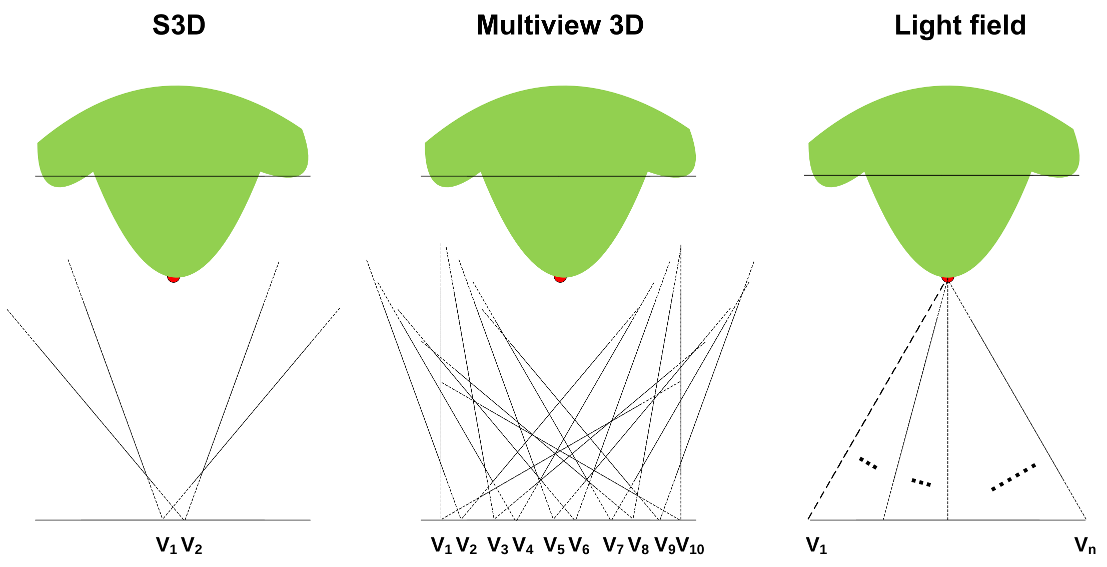


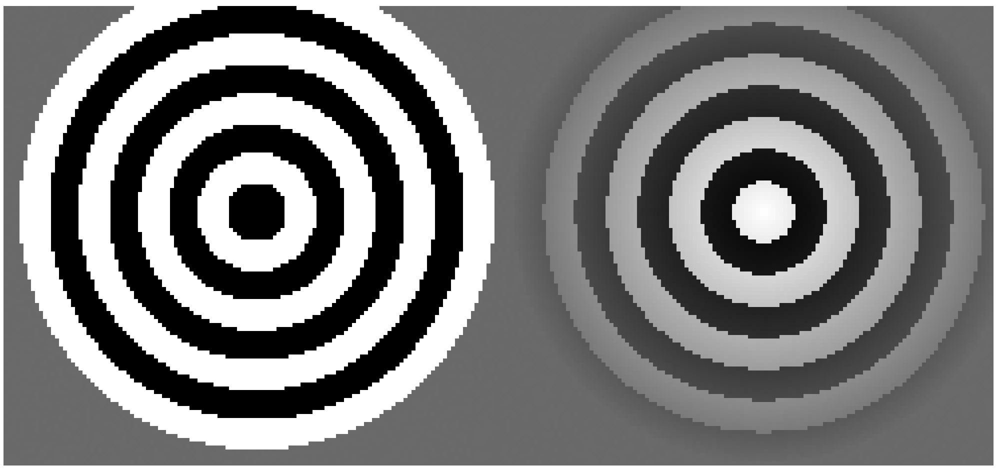
Figure 1.1: Displaying in 3D using Stereoscopic 3D (S3D), multiview 3D and light field technologies.

light rays from multiple perspectives using a set of projection engines. Various scene points are described by intersecting light rays at corresponding depths.

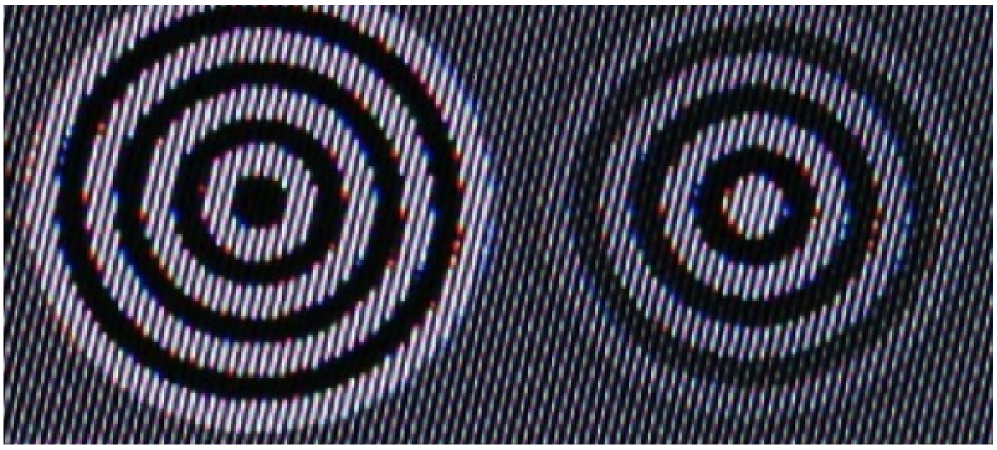
Recent advances in computational displays showed several improvements in various dimensions such as color, luminance & contrast, spatial and angular resolution (see [2] for a detailed survey of these displays). Projection-based light-field displays, are among the most advanced solutions. The directional light emitted from all the points on the screen creates a dense light field, which, on one hand, creates stereoscopic depth illusion and on the other hand, produces the desirable motion parallax without involving any multiplexing. Figure 1.1 gives an overview of traditional S3D, multiview 3D and light field displaying technologies.

As shown in Figure 1.1, consider a sample scene (shown in green) and a point in the scene (shown in red). From the rendering aspect, the major difference is that S3D and multiview rendering do not consider the positions of 3D scene points. Therefore we have only two perspectives of a given scene on a S3D display and multiple but a still limited number of perspectives on a multiview 3D display.

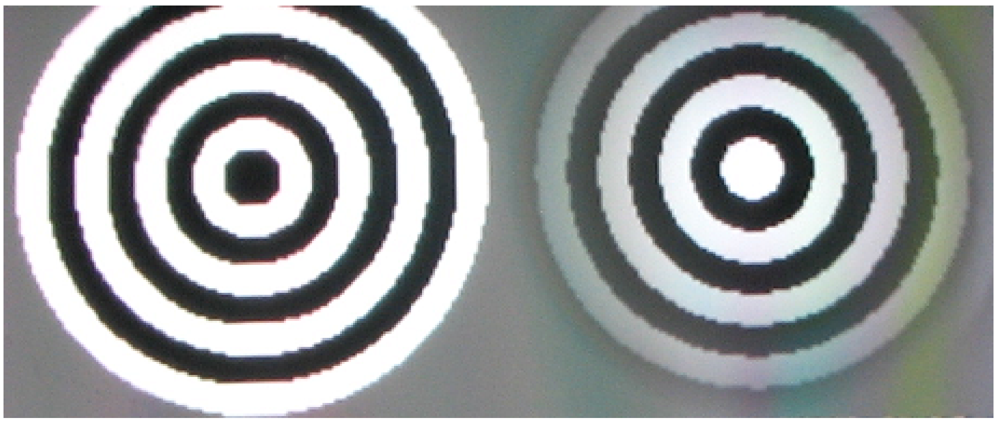
DOI:10.15774/PPKE.ITK.2015.005



(a)



(b)



(c)

Figure 1.2: Light field and multiview autostereoscopic display comparison (a) Original 2D input patterns; (b) Screen shot of multiview autostereoscopic display; (c) Screen shot of projection-based light field display.

DOI:10.15774/PPKE.ITK.2015.005

In both the cases, all perspectives are actually 2D projections of the 3D image, which collectively define the scene. Light field displays in contrast define the scene using directional light beams emitted from the scene points. Thus each scene point is rendered differently from other scene points resulting in more realistic and accurate 3D visualization. A direct advantage of light field displays can be clearly observed from Figure 1.2. Figure 1.2(a) shows two patterns of concentric circles lying in a plane. Figure 1.2(b) shows the screen shot of the patterns visualized on a barrier based multiview autostereoscopic display while Figure 1.2(c) shows the screen shot of a light field display.

As the number of views and effective FOV in horizontal direction are different for two displays, for a fair comparison all the views are engaged with the same 2D pattern when recording the screen shots. In case of multiview autostereoscopic displays, we have a limited number of comfortable 3D viewing zones called sweet spots. Within these zones a user can see an accurate 3D image while within the transitions between two neighboring sweet spots the image is blurry and disturbing. A user located anywhere within the FOV of multiview displays can always see mesh-like parallax barrier (1.2(b)). The size of the barrier is a major limitation in the display hardware and the perspective shift for motion parallax is neither smooth nor uniform. Another inherent drawback of the parallax barrier based approach is the limitation of the total achievable 3D FOV. In case of light field displays, there are no sweet spots and no light barriers.

1.2 Open-Ended Questions

In order to meet the FOV requirements of a light field display and to have a realistic and natural 3D displaying, we need to capture views from several cameras. In the current generation of light field displays, horizontal only parallax is supported and thus it is enough to consider the camera setups in one dimensional configuration. It is clear that for providing high quality 3D experience and supporting motion parallax, we require massive input image data. The increased dimensionality and size of the data opened up many research areas ranging through capturing, processing, transmitting, rendering and interaction. Some of the open ended questions are

- **Content creation** - what are the optimal means to acquire/capture suitable light field data to fully meet the requirements of a given light field display?
- **Representation, coding & transmission** - what is the optimal format for storing and transmitting acquired light field data?
- **Rendering & synthesis** - how the captured light field data can be used for rendering and how to create/synthesize the missing light rays?
- **Interaction** - how the interaction can be extended for light field content to appeal to the design, gaming and media industry?

DOI:10.15774/PPKE.ITK.2015.005

- **Human visual system** - how the human visual system functioning can be exploited to improve the quality of light field rendering?

- **Quality Assessment** - what are the ways to measure/quantify the quality of a light field rendering and how can we automatically detect any annoying artifacts?

1.3 Aims and Objectives

The main aim of the current work is to assess the requirements for suitable and universal capture, rendering and interaction of 3D light field data for projection-based light field displays. Specifically, the emphasis is on *representation*, *rendering* and *interaction* aspects. Considering the existing rendering procedures, the first part of the work aims to derive the requirements for light field representation for given display configurations and also presents a rendering prototype with light weight *multiview* image data. The second part of the work aims to optimize the rendering procedures to comply with the display behavior. In the last part, the objective is to experiment and evaluate interactive rendering using low cost motion sensor device.

1.4 Scope of Work

Concentrating on projection-based light field displays, the work has three branches : assessing the requirements for a suitable representation, rendering visual quality enhancement through depth retargeting and exploring direct touch interaction using Leap Motion Controller.

- By implementing and testing the state-of-art rendering methods, requirements for a future light field representation are presented for projection-based light field displays. The scope of this work does not include deriving a standard codec for encoding/decoding the light field.

- The depth retargeting method solves a non-linear optimization to derive a novel scene to display depth mapping function. Here, the aim is not to acquire/compute accurate depth, instead use a real-time capturing and rendering pipeline for investigating adaptive depth retargeting. The retargeting is embedded into the renderer to preserve the real-time performance. For testing the performance, the retargeting method is also applied to synthetic 3D scenes and compared with other real-time alternatives. Comparison is not carried out with methods that are not real-time.

- The direct touch interaction setup provides a realistic direct haptic interaction with virtual 3D objects rendered on a light field display. The scope of this work does not include any modeling of an interaction device.

DOI:10.15774/PPKE.ITK.2015.005

1.5 Workflow

The work began with a study of several light field representations for real-world scenes that were proposed in the literature. The main idea behind investigating a representation is not deriving an efficient and real-time light field encoding/decoding method, but rather explore the light field display geometry to investigate a suitable representation. Existing and well-knowns representations are based on the notion of displaying 3D from multiple 2D. However, projection-based light field displays represent a scene through intersecting light rays in space i.e., instead of several discrete 2D views, any user moving in front of the display perceives several *light field slices*. Thus, the existing light field representations are not optimized for such displays and the required representation should consider the process of light field reconstruction (rendering) on these displays.

A popular approach for light field rendering from real-world scenes is to acquire images from multiple perspectives (*multiview images*) and re-sample the captured database to address the display light rays [6, 7, 8]. Due to the geometrical characteristics of light field displays, it is more reliable to render using images acquired from several closely spaces cameras. Depending on the display geometry, we may need 90-180 images. By carefully examining how several light rays leaving the display are shaded, I derived a data reduction approach that eliminates the unwanted data during reconstruction process. Based on the immediate observations, a light weight representation based on discrete 2D views with each view having a different resolution than the other is followed. These investigations also opened up the requirements for a novel light field representation and encoding schemes.

A different approach for light field rendering in the literature is all-in-focus rendering [9]. Instead of directly sampling the captured data base, this rendering approach also computes the depth levels for various display light rays. A major advantage using this approach is that, it is possible to achieve a good quality light field with less number of cameras. However, the visual quality is highly dependent on the accuracy of the available depth. Currently, there are not any real-time methods that can deliver pixel precise accurate depth for novel light field reconstruction. Thus, this rendering approach is not taken forward for investigations related to light field representation.

One of the important constraints of any 3D display is the available *depth of field (DOF)*. If the extent of scene depth is beyond the displayable extent, a disturbing blur is perceived. An important goal of the work was to address the problem of depth retargeting for light field displays (constraining the scene depth smartly to fit to the display depth of field). Warping based approaches are proposed in the literature to address the problem of depth retargeting in the context of Stereoscopic 3D. These methods do not explicitly consider the scene [10] and they need further adaption to suit to the full parallax behavior of a light field display. Furthermore with methods based on warping, distortions are inevitable, especially, if there are vertical lines in the scenes. An alternative is to compute and work on the scene depth directly to achieve retargeting. As depth calculation is an integral part of the all-in-focus rendering pipeline, this approach is taken

DOI:10.15774/PPKE.ITK.2015.005

further and the behavior is adapted to achieve content adaptive and real-time depth retargerting on a light field display.

Interaction and rendering are intertwined processes. As light field displays represent a novel technology in the field of 3D rendering, they also require design and evaluation of novel interaction technologies and techniques for successful manipulation of displayed content. In contrast to classic interaction with 2D content, where mice, keyboards or other specialized input devices (e.g., joystick, touch pad, voice commands) are used, no such generic devices, techniques, and metaphors have been proposed for interaction with 3D content. The selected interaction techniques usually strongly depend on individual application requirements, design of tasks and also individual user and contextual properties. One of the goals of the current work is to enable accurate, natural and intuitive freehand interaction with 3D objects rendered on a light field display. For this purpose a basic and most intuitive interaction method in 3D space, known as "direct touch" is proposed. The method directly links an input device with a display and integrates both into a single interface. The main aim was not to research a hand or motion tracking hardware, but use commercially available *Leap Motion Controller* sensor for motion sensing and achieve an interactive rendering for manipulation of virtual objects on a light field display.

All the work is implemented in C++ using OpenGL SDK [11]. The rendering code for light field depth retargeting is written in CUDA [12]. For programming the interaction part, Leap SDK [13] is used.

1.6 Dissertation Organization

The dissertation is organized as follows: Chapter 2 gives an overview of projection-based light field displays and content displaying. State-of-art methods for rendering light field on these displays from real-world as well as synthetic scenes are detailed. Chapters 3, 4 & 5 are self-contained. Each of them starts with a little introduction, related works and presents main contributions and results on light field representation, depth retargeted rendering and free hand interaction respectively. New scientific results from the current work (described in Chapters 3, 4 & 5) are summarized in Chapter 6. Chapter 7 briefly discusses the overall applications of the work. Finally, conclusions and future work are presented in Chapter 8. It is important to note that in this thesis description, the words: *HoloVizio & holographic light field display* are used interchangeably and represent a *projection-based light field display*.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 2

Projection-Based Light Field Displays - Background

2.1 Display Components

Light field displays are of high resolution (order of magnitude of one million pixels) and can be used by several users simultaneously. There is no head-tracking involved and thus the light field is available from all the perspectives at any given instance of time. The light field display hardware used for this work - *HoloVizio* was developed by Holografika.

The *HoloVizio* light field display in general uses a specially arranged array of optical modules, a holographic screen and two mirrors along the side walls of the display (see Figure 2.1). The screen is a flat hologram and the optical modules are arranged densely at a calculated distance from the screen. Light beams emitted from the optical modules hit the holographic screen, which modulates them to create the so-called light field. Two light rays emitted from two optical modules crossing in space define a scene point. Thus, for realization of the light field function, light beams are defined over a single planar surface. In real-world, the directions and light beams emitted from a point in space are continuous. In practice, however, it is not possible to imitate such continuousness due to the non-negligible size of the display hardware, which results in the discretization of the light beam directions. For more closer approximation to the real-world, optical modules are arranged densely to yield sufficient angular resolution that creates an illusion of continuousness. Furthermore, the display screen is holographically recorded and has randomized surface relief structure that provides controlled angular light divergence. The optical properties of the screen enable sharp directive transmission along horizontal direction and allow us to achieve sub-degree angular resolution.

The discretization of direction incorporated by light field displays leaves us a parameter to choose, the angular resolution. High angular resolution drives us closer to real world at the cost of increased data to handle and vice versa. The angular resolution and the total *field of*

DOI:10.15774/PPKE.ITK.2015.005

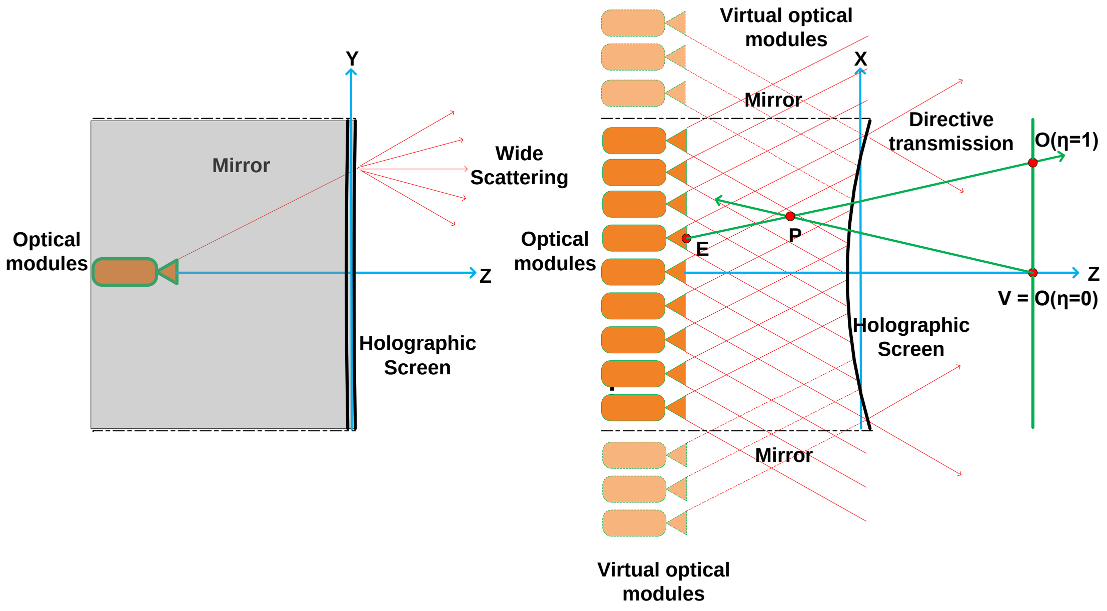


Figure 2.1: Light field display model and optical characteristics. The display hardware setup consists of three parts: spatially arranged optical modules, a curved (cylindrical section) holographic screen and a pair of mirrors along display side walls. Left: vertically, the screen scatters widely and users perceive the same light field from any height. Right: horizontally, the screen is sharply transmissive with minimum aliasing between successive viewing angles.

view supported by a *HoloVizio* are directly proportional to the number of optical modules. As described earlier, the optical properties of the screen allow directional transmission of light in horizontal direction with minimum aliasing (see Figure 2.1 left). If the input light beam is perfectly coherent, there will be no aliasing. In vertical direction, after hitting the screen the light beams scatter widely and thus users see exactly the same image at any height on the display. Such an arrangement can be used to create horizontal only parallax display (see Figure 2.1 right). Mirrors covering the display side walls reflect back any light beams hitting them towards the screen, giving an illusion that they are emitted from a virtual light source outside the display walls (see Figure 2.1 right). Thus the side mirrors increase the effective *field of view* by utilizing all the emitted light rays.

2.2 Light Field Display Geometry

For rendering light field content on holographic light field displays, it is important to understand the display geometry that enables us to model light rays. In practice, however, geometry modeling itself may not be sufficient for rendering due to any mechanical misalignment of the optical modules from the ideal design. Thus, it is required to have an additional step - *display geometry calibration* for precisely tracing the display light rays. Note that the aim of the current work is not to improve display design or geometry calibration. They are only described here to provide a

DOI:10.15774/PPKE.ITK.2015.005

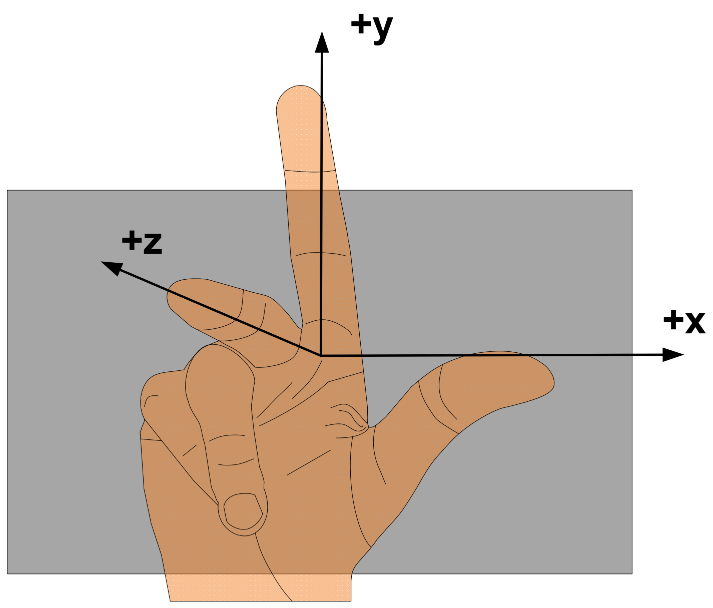


Figure 2.2: Right handed co-ordinate system used by OpenGL

foundation for understanding the rendering part.

2.2.1 Modeling Display Geometry

The physical screen (hologram) center is always assumed to be the origin in display Cartesian co-ordinate system and as the case with OpenGL, a right hand co-ordinate system is assumed (see Figure 2.2). The screen plane is assumed to be the XY plane. Projection units are arranged along negative Z direction and the viewers/observers are along the positive Z direction. After determining the physical size of the optical modules, the following parameters are derived: exact *positions of individual optical modules* and the *observer distance* from the screen. These parameters are used during the rendering phase to trace a given display light ray.

2.2.2 Display Geometry Calibration

From a more general perspective, geometry calibration is built on the classic two-step approach in which position and frustum of each display optical module are found through parametric optimization of an idealized pinhole model and any remaining error is corrected using a post-rendering 2D image warp that 'moves' the pixels to the correct idealized position [9]. For the first calibration step, projector positions are assumed to be known, and vision based techniques are used to find their orientation and projection matrices. Usually, the whole calibration procedure is performed by covering the holographic surface with a standard diffuser and photographing it with a camera. For each optical module, an asymmetric pattern is projected and the projected

DOI:10.15774/PPKE.ITK.2015.005

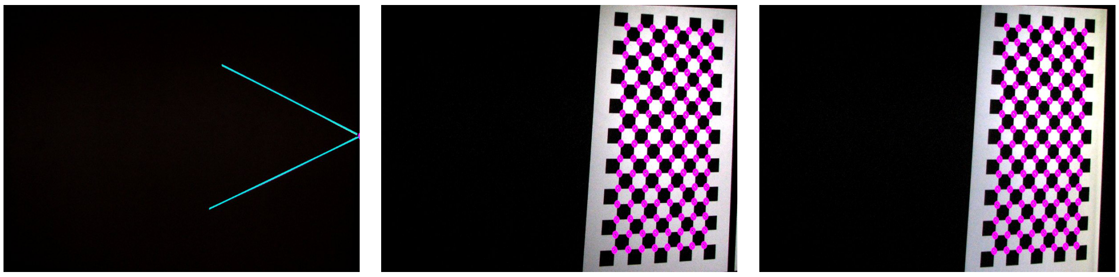

(a) (b) (c)

Figure 2.3: Light field display calibration - An asymmetric pattern detects a mirror. Checkerboard patterns are then projected in the direct and mirror image to calibrate the projector

images are analyzed to determine mirror positions (see Figure 2.3(a)). If no mirror is detected, single viewport is associated to the optical module; otherwise, a "virtual projector" is created on the other side of the mirror and calibrated by separately handling the mirrored and direct view rendered in different viewports. In this calibration step, a checkerboard pattern is projected onto the screen (see Figure 2.3(b) and (c)). The full orientation and the perspective matrix are derived from the detected corners. The remaining errors are then corrected using a cubic polynomial approximation method for mapping undistorted to distorted coordinates. For more details on display geometry calibration, please see [14] and [15].

2.3 Display Spatial Characteristics

One of the contributing factors to the degradation of visual quality in general 3D display setups is the varying spatial resolution with depth. In case of holographic light field displays, the spatial resolution of the display changes with depth, according to:

$$s(z) = s_0 + 2\|z\| \tan(\frac{\Phi}{2}) \tag{2.1}$$

where z is the distance to the screen, and s_0 is the pixel size on the screen surface [16] (see Figure 2.4). Thus, the spatial resolution is higher on the surface of the screen and the diminishes as we move away from screen. Therefore, to optimize the viewing experience on light field displays, the scene center must coincide with display's $Z = 0$ plane, total scene depth should comply with the limited depth of field of the display and the frequency details of the objects in the scene should be adjusted conforming to the displays spatial characteristics.

2.4 Modeling Light Rays for 3D Displaying

To describe the light rays emitted out of the screen, it is necessary to know the transformation that a light ray undergoes in any HoloVizio system. For rendering, multiple-center-of-projection

DOI:10.15774/PPKE.ITK.2015.005

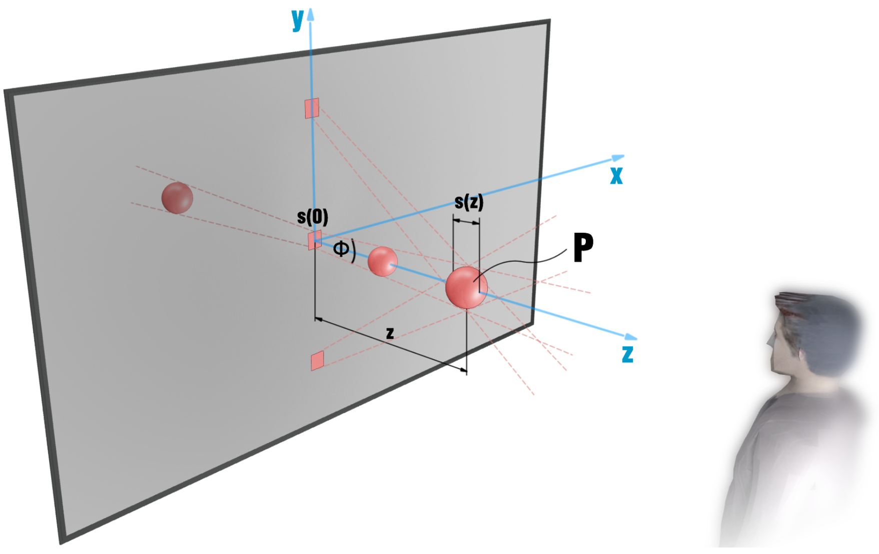


Figure 2.4: The light field display's spatial resolution is depth dependent. The size of smallest displayable feature increases with distance from the screen. Thus, objects rendered on the screen surface appear sharper.

(MCOP) technique [17, 16] is used that helps in preserving the motion parallax cue. Specifically, this perspective rendering approach assumes a viewing line in front of the display at fixed height and depth. In case of holographic light field display, the viewing line is located at depth $z = Obseverdistance$ and at height $y = 0$. Given an observer at $\mathbf{V}$ (see Figure 2.1 right), the ray origin passing through a point $\mathbf{P}$ is determined by

$$\mathbf{O} = ((1-\eta)(V_z) + \eta(E_x + \frac{P_x - E_x}{P_z - Ez}(V_z - E_z), V_y, V_z)) \tag{2.2}$$

where $\mathbf{E}$ is the position of the optical module under consideration (see Figure 2.1 right) and η is a interpolation factor, which allows smoothly transition from standard single view (2D) perspective rendering (with η = 0) to full horizontal parallax (3D) rendering (with η = 1). The ray connecting $\mathbf{O}$ to $\mathbf{P}$ is used as projection direction in order to transform the model into normalized projection coordinates. 3D positions of ray origins corresponding to the viewport pixels of various optical modules are determined using the display geometry calibration information. The 3D rendering equation (with η = 1) is referred to as *holographic transform*.

2.5 General Rendering Setup

Figure 2.5 shows a general rendering setup for processing, rendering and displaying light field content. On the front-end we have an application node. The back-end typically consists of the rendering cluster that drives the optical modules and the display. Front-end and back-end are connected via gigabit Ethernet connection. Note that the number of nodes inside the rendering

DOI:10.15774/PPKE.ITK.2015.005

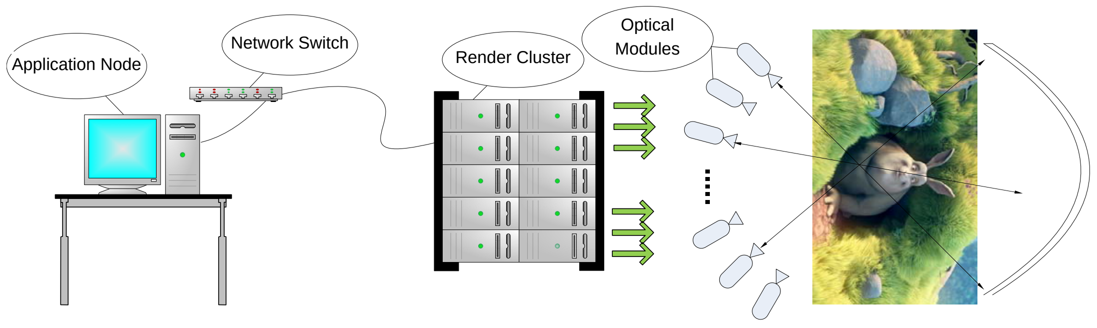


Figure 2.5: General light field rendering hardware setup.

cluster depends on the size of the display and the number of available optical modules. For small scale displays having low resolution, both the front-end and back-end could be a single PC.

The source data for rendering can be a real-world scene in the form of multiview images or a synthetic scene. In both the cases, the application node has access to scene description and related meta-data. This information is streamed over the network to the rendering cluster which is equipped with light field geometry and calibration data. Each node in the cluster adapts and processes the received content taking into account the display geometry. The holographic screen helps in realizing the 3D information in the form of light rays projected by the optical modules.

2.6 Rendering Light Field From Synthetic Scenes

Visualization of synthetic scenes on light field displays requires rendering the given scene from many viewpoints that correspond to the characteristics of the specific light field display. One way to achieve this is using the HoloVizio OpenGL wrapper (see [4]). This wrapper library intercepts all OpenGL calls and sends rendering commands over the network to the backend driving the light field display as well as modify related data (such as textures, vertex arrays, VBOs, shaders etc.) on the fly to suit the specifications of the actual light field display. The wrapper functionality is shown in Figure 2.6. The wrapper is designed in such a way that its operation is completely transparent to the client application producing the scene and it requires no modification of the client application (in the case of third-party applications such modifications are usually not possible).

As we have control over the scene in synthetic environment, it is possible to exploit the additional OpenGL features provided by the wrapper library, through which additional semantic information related to the currently rendered scene can be supplied to adjust visualization in 3D space. An example of this additional information is the distance between the viewpoint and center of the 3D model. What constitutes as center is specific to the model semantics and is not deducible from the OpenGL command stream. When mapping the application's Region Of Interest (ROI) to the light-field display's ROI center of the model is mapped to be slightly behind the display's

DOI:10.15774/PPKE.ITK.2015.005

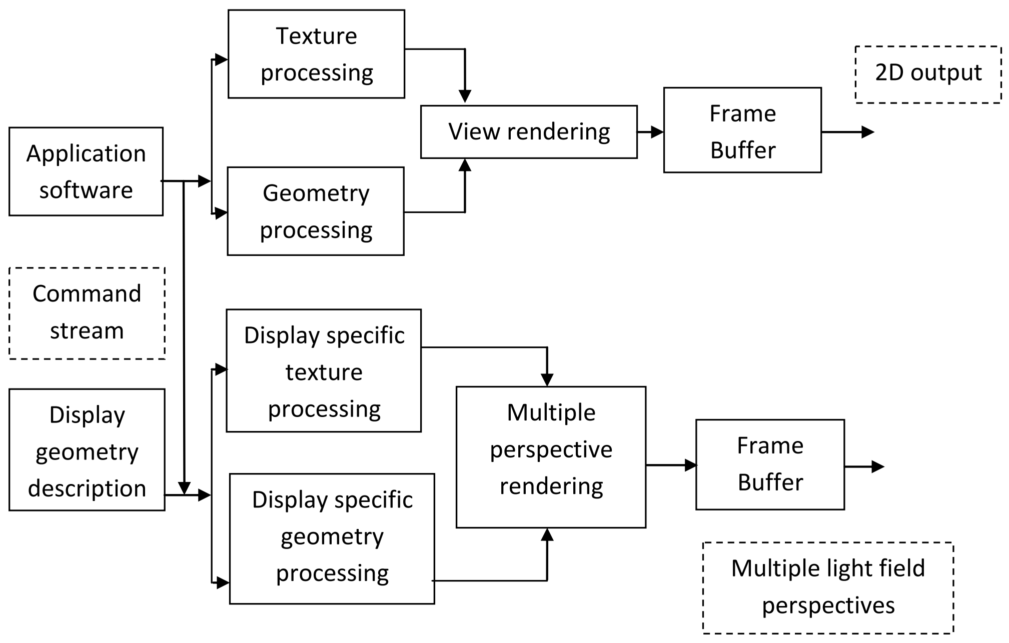

Texture processing
2D output
View rendering
Frame Buffer
Application software
Geometry processing
Command stream
Display specific texture processing
Multiple perspective rendering
Display geometry description
Frame Buffer
Display specific geometry processing
Multiple light field perspectives

Figure 2.6: Light field rendering from OpenGL command stream: the various commands from application software are modified in real-time using the display geometry description. Geometry and texture information is modified and processed to render multi-perspective light field.

screen ensuring that model is in focus. The displacement by which we push the model behind the screen has been experimentally determined for different scale levels, as the same amount does not always work well.

2.7 Rendering Light Field From Real-World Scenes

Rendering real-world scenes captured using multiple camera images on projection-based light field displays is often referred to as *light field conversion*. As an input to this conversion, we need a set of camera images and the geometry information of capturing cameras and target light field display. The result of the conversion is a set of module images for the corresponding light field display. In the following sub-sections the capturing geometry and state-of-art methods for mapping camera and display geometry for creating light field are discussed.

2.7.1 Modeling Capture Geometry

To capture a scene in 3D, we need a set of cameras arranged in certain spatial topology. As the current HoloVizio displays incorporate horizontal only motion parallax, a more useful approach in realizing a 3D scene on HoloVizio is to capture from horizontally arranged/aligned cameras (1D). Two preferred configurations are linear and arc systems as shown below in Figure 2.7

Describing real world's 3D voxels (imaginary) in terms of camera sensor (where the image in

DOI:10.15774/PPKE.ITK.2015.005

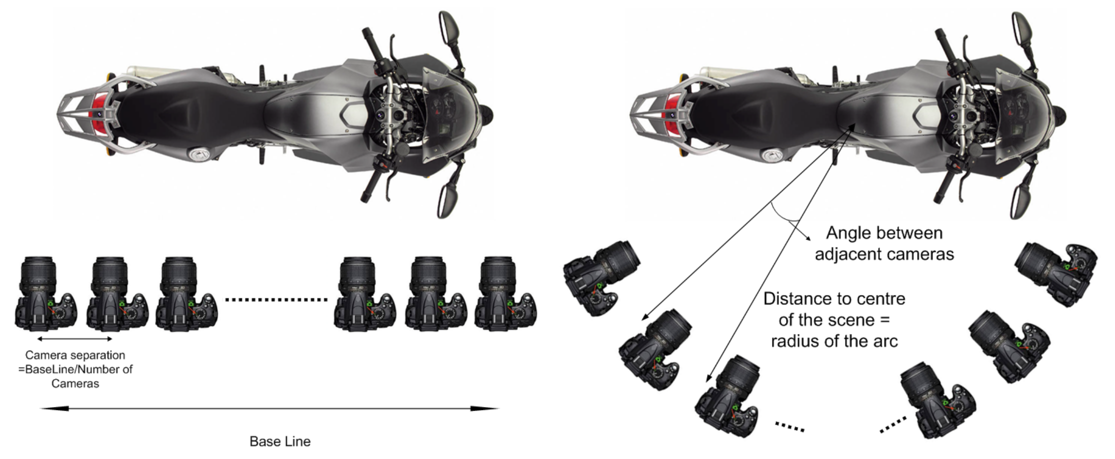

Camera separation =BaseLine/Number of Cameras
Base Line
Angle between adjacent cameras
Distance to centre of the scene = radius of the arc

Figure 2.7: Cameras arranged in linear and arc topologies

pixels is defined, will be referred to as *viewport* hereafter) pixels involves a series transformations. Each transformation can be defined in the form of a matrix. Without the loss of generality, the transformation, T_{WV}, from 3D Cartesian co-ordinate system to a camera viewport is defined as:

$$T_{WV} = V_{pm} \times PV \tag{2.3}$$

Where, T_{WV} - world to viewport transform matrix; V_{pm} - camera viewport matrix; PV - camera projection view matrix. The Projection view matrix (PV) of a camera is defined as follows:

$$PV = P_m \times V_m \tag{2.4}$$

where, P_m - camera projection matrix; V_m - camera view matrix

Camera Viewport Matrix

The viewport matrix of the camera (inside camera) is calculated as:
w, h - width and height of a given camera

$$V_{pm} = \begin{pmatrix} w/2 & 0 & 0 & (w/2) + X_{offset} \\ 0 & -h/2 & 0 & (h/2) + Y_{offset} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.5}$$

Assuming the cameras are identical, once the viewport matrix is calculated, it remains same for all the cameras. X_{offset} and Y_{offset} are the optional pixel shifts. If the complete viewport is utilized, these two values are zeros.

DOI:10.15774/PPKE.ITK.2015.005

Camera Projection Matrix

For a given camera, projection matrix is calculated as:

n, f - distances to near and far clipping planes from the camera optical center

a - camera aspect ratio = V_h/V_w (V_h - viewport height; V_w - viewport width)

e - camera focal length = $1/(tan(FOV_X 2))$ (FOV_X - camera horizontal *FOV*)

$$P_m = \begin{pmatrix} e & 0 & 0 & 0 \\ 0 & e/a & 0 & 0 \\ 0 & 0 & -(f+n)/(f-n) & -2fn/(f-n) \\ 0 & 0 & -1 & 0 \end{pmatrix} \tag{2.6}$$

For a real world camera, it is not practically possible to confine the near clipping plane (C_{NP}) and far clipping plane (C_{FP}) that the camera sees. However, because of practical considerations, there is a limit to depth range that can be provided on the display. As we are trying to model the scene on HoloVizio, the following approximations are valid:

$$C_{NP} = Distancetocenterofthescene - Extents(z)/2 \tag{2.7}$$

$$C_{FP} = Distancetocenterofthescene + Extents(z)/2 \tag{2.8}$$

where, the parameter $Extents(z)$ should be supplied as an input. Extents refer to the region of interest in the scene that we want fit within the depth range of HoloVizio as shown in Figure 2.8. Thus given a scene, the projection matrix of all the cameras remains same assuming identical cameras.

Camera View Matrix

The camera view matrix calculation requires 3D position of the camera. This position information can be calculated by assuming the 3D Cartesian co-ordinate system with origin as the centre of the scene. For example, in a linear camera arrangement, given the number of cameras, baseline distance and distance to the centre of scene, a camera position, C_{CP} can be calculated as follows:

$$C_{SP} = (-B_D/2, 0.0, D_S) \tag{2.9}$$

DOI:10.15774/PPKE.ITK.2015.005

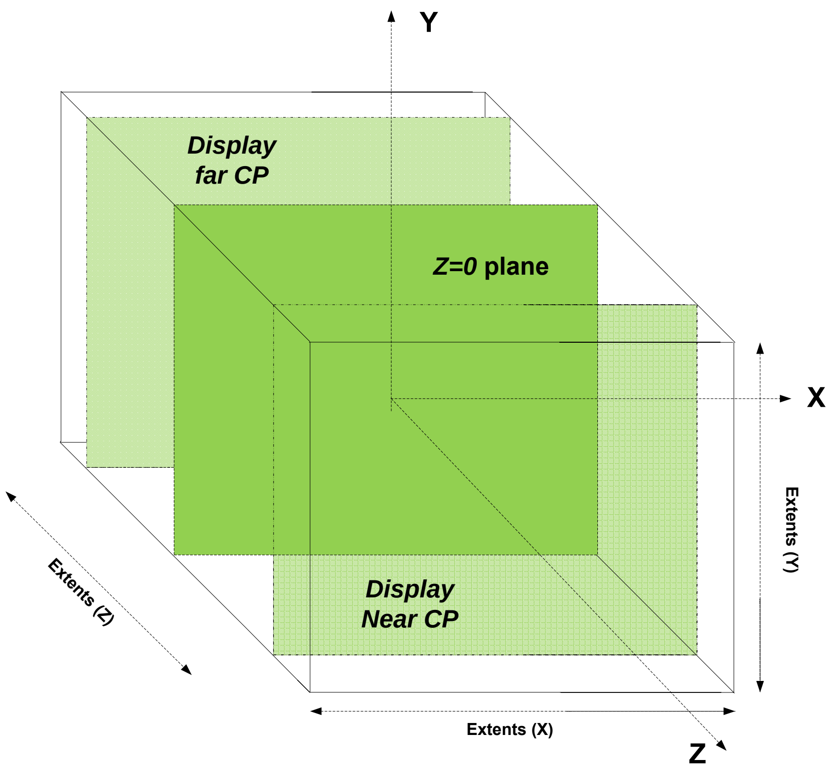


Figure 2.8: Scene region of interest expressed in display coordinates

Where, C_{SP} - starting position of the camera rig; B_D - base line distance; D_S - distance to the center of the scene

$$\delta = (B_D, 0.0, 0.0) \tag{2.10}$$

$$C_{CP} = C_{SP} + \delta * ((C_I)/(N-1)) \tag{2.11}$$

Where, C_{CP} - Current camera position; C_I - Current camera index; N - Total number of cameras The HoloVizio screen has origin at its centre and thus a simple translation matrix with (x, y, z) translation parameters (same as the camera position) would serve as a view matrix. A simple view matrix of a camera is calculated as:

(x,y,z) - displacement of camera with respect to screen center (origin),same as camera position

DOI:10.15774/PPKE.ITK.2015.005

$$V_m = \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.12}$$

The above matrices help us to transform a point in a given 3D on to a camera at a given position.

2.7.2 Mapping Capture and Display Geometry - Towards Content Rendering

The process of mapping capture and display light rays is referred to as *conversion table generation*. It represents generating lookup coordinates for an array of camera images per displayed pixel. For calculating the conversion table entries, a set of pinhole cameras is assumed. Consider a sample camera and display setup as shown in Figure 2.9. For geometry mapping, the cameras are assumed to be located in front of the screen with focus plane (*Cameras Opt.* from Figure2.9) coinciding with the screen of the display, near plane in front of the display screen and far plane behind screen. For generating the conversion tables for a given display optical module, we need an information on display to eye rays for that module *i.e.,* the rays leaving from the viewports of the optical module towards the *observer line*. This can be calculated using the position of the optical module and the *holographic transformation* (see section 2.4). Once we have the current display to eye light ray, the intersection with the camera array can be solved using vector algebra. The intersection point can be used to deduct *closest cameras* in the camera space corresponding to the current display light ray. By suitably sampling color from nearest cameras and using linear interpolation, display light rays are shaded resulting in a light field.

2.7.3 Real-Time Light Field Capture and Display - State-of-the-art

In this section mainly two architectures will be discussed that deal with real-time light field capture and display.

Simple Light Field Rendering

A number of papers showing considerable advances in the areas of real-time 3D video or light field capture and display have been published in the recent years. Most of the approaches are based on pure light field conception and considers the sets of rays captured by the cameras as light field samples. During rendering, captured light field database is re-sampled to produce light rays from a required point of view [6, 7]. These systems do not take scene geometry in to account and thus, in accordance with the plenoptic sampling theory [18], for photo-realistic rendering, one may require very high number of cameras to substantially sample the light field. Estimating the scene geometry helps in producing higher quality views from arbitrary view positions using

DOI:10.15774/PPKE.ITK.2015.005

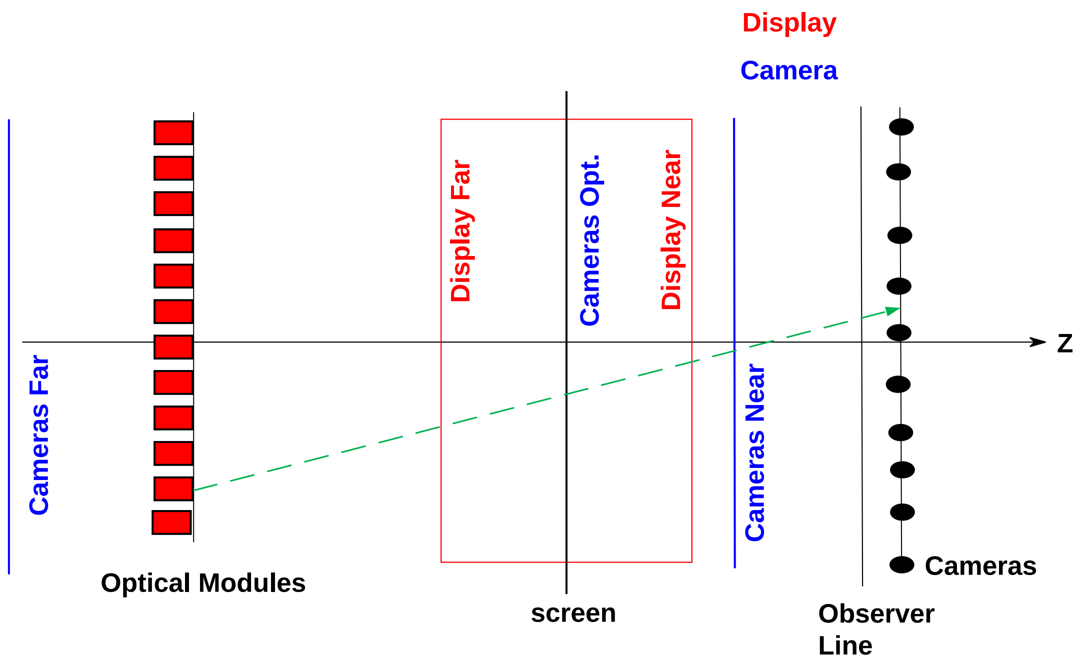


Figure 2.9: Mapping camera and display geometry for rendering.

less cameras [19, 20, 21, 22].

A real-time capture and rendering system on a projection-based light field display with 27 USB cameras is first presented by Balogh *et. al.*,[8]. They assume that the ray origin on the surface of the display screen is voxel position represented by the current ray. This origin is projected on to the nearest camera's viewports. Once we have valid viewport coordinates, we calculate suitable weights for the acquired camera pixels based on their distances. The visual quality of the produced light field in such a setup is highly a function of camera spacing and thus dependent on number of cameras. This is explained in Figure 2.10. If the camera separation increases, the resolution of near and far clipping planes on the screen degrades. For an ideal light field representation, we need to have cameras along all the floating point positions along which the observer line is sampled by the display light rays. In practice this number is varying from one display to another based on the display geometry. It is found empirically that if a display has a horizontal FOV of Φ degrees and an angular resolution of Ω, the required number of cameras, N_{CC} can be approximately given by:

$$N_{CC} = \Phi/\Omega. \tag{2.13}$$

DOI:10.15774/PPKE.ITK.2015.005

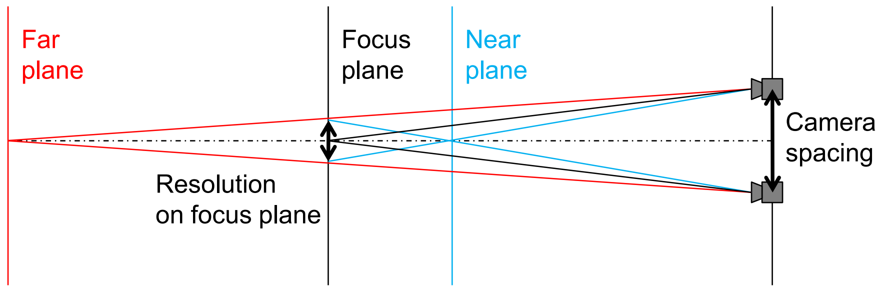


Figure 2.10: Simple light field rendering - dependency on the camera spacing. As the camera spacing decreases, the apparent 3D resolution on the display increases.

Light Field Rendering Through Geometry Estimation

Estimating the scene geometry helps in producing higher quality views from arbitrary view positions using less number of cameras. In general, scene depth estimation can be global or local. On one hand, globally consistent depth estimation is computationally expensive and on the other hand local depth estimation methods are real-time, but prone to local minima resulting in poor quality depth maps. Fabio Marton *et. al.*, developed a multi-resolution approach to estimate scene depth on-the-fly from the perspectives of display optical modules ([9]). They showed that it is possible to achieve an all-in-focus rendering by estimating the depth for display light rays. The method extends a coarse-to-fine stereo-matching method for real-time depth estimation. Using a space-sweeping approach and a fast Census-based area matching. This depth estimation module is adapted to projection-based 3D display imaging geometry for rendering light field. If the depth of a voxel being rendered is known, we can travel along the current ray direction Z steps to reach the voxel in the display space. This position can be transformed into the viewports of the nearby cameras for more accurate light field when using less number of cameras. Figure 2.11 shows the main difference between the simple light field and the geometry based light field renderings. In case of pure light field rendering, the black dot on the surface of the screen is used and geometry based rendering uses the blue dot for reconstructing light field.

DOI:10.15774/PPKE.ITK.2015.005

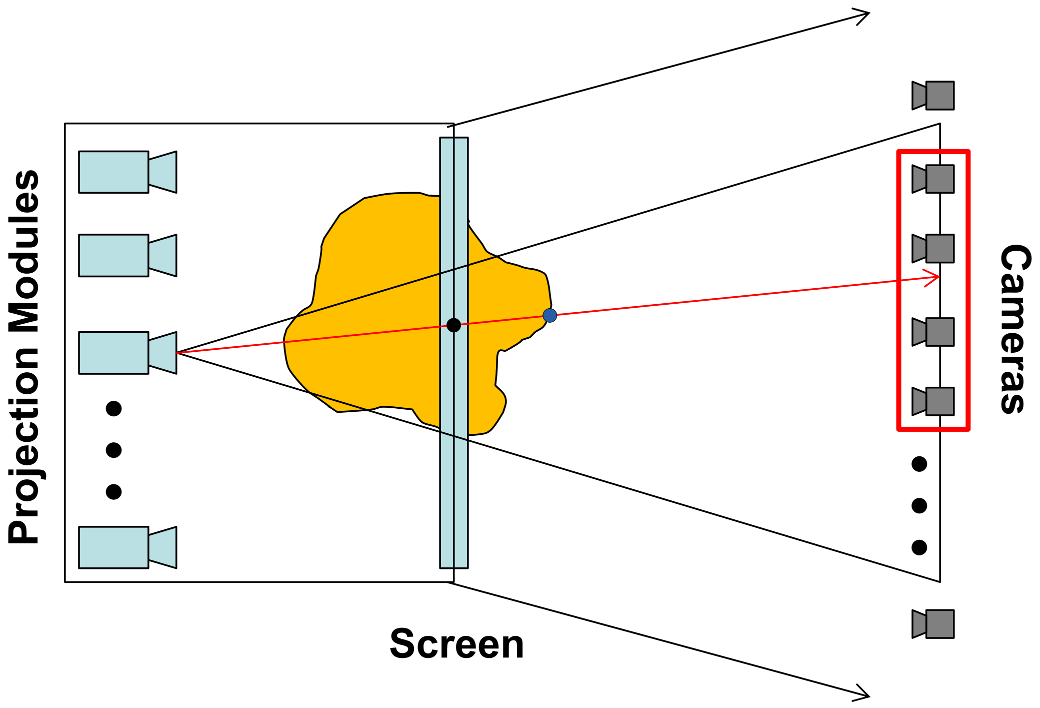


Figure 2.11: Difference between simple light field and geometry based light field rendering - Simple light field rendering considers the intersection point of current light ray (shown in red) emitted by a given optical module and the screen surface and samples the colors captured by the nearest cameras (shown in red rectangle) at the intersection point (black dot in the current example). Geometry based light field rendering attempts to estimate the depth of current light ray and samples the colors captured by the nearest cameras at the estimated depth (blue dot in the current example) in the direction of the light ray.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 3

Determining the Requirements for Representing Holographic Light Field

3.1 Introduction

During the past few years, the demand of remote collaboration systems has increased firmly in the communication world. The introduction of the large /ADDand high-resolution displays in the collaboration space added another appealing dimension; now, the collaboration system is capable of integrating multiple cameras in order to capture and transmit the whole scene of the collaboration space. Projection-based light field displays, used for 3D video display, could be one of such examples of large high-resolution displays. Cutting-edge telepresence systems equipped with multiple cameras for capturing the whole scene of a collaboration space, face the challenge of transmitting huge amount of dynamic data from multiple viewpoints. With the introduction of Light Field Displays into the remote collaboration space, it became possible to produce an impression of 3D virtual presence.

3.2 Light Field Data Transmission Problem

Light field displays in current generation rely on the images obtained from cameras arranged in various spatial configurations. To have a realistic and natural 3D collaboration using light field displays, the data in the form of multiple camera images needs to be transmitted in real time using the available bandwidth. Depending on the *FOV* of the target light field display, we may need up to 100 cameras for good quality light field reconstruction. Parallel acquisition of video data from these many sensors results in huge amount of data at each time instant, which can quickly saturate the available bandwidth of the data link. Thus, for applications involving projection based light field displays, it is required to carefully devise a data representation procedure that optimizes the bit rate and quality of reconstructed light field.

DOI:10.15774/PPKE.ITK.2015.005

Classical compression methods might resolve this issue to a certain level. Inspired by the initial *MPEG (Moving Picture Experts Group)* compression technique, these compression methods tend to find the spatial and temporal similarities to identify the information which can be discarded. However, in many cases the achieved compression level is by far insufficient in the context of projection-based light field displays. Moreover, the available compression schemes do not consider any of the display related attributes.

3.3 Main Contributions

To conceive a suitable light field representation, it is vital to explore multiview image data and the process of multiview to light field conversion. This ensures that the display behavior is incorporated into the representation aspect, which aids in bringing down the size of the data to be transmitted from the acquisition side to the receiver side. The main contributions from the current work are the following:

- Examining the light field conversion process [C2] from camera images acquired from several closely spaced cameras [O1] [J3], I proposed a fast and efficient data reduction approach for light field transmission in multi-camera light field display telepresence environment[C3] [O2] [O3].

- By simulating several projection-based light field displays with different *FOV* and exploring the direct data reduction possibilities (without encoding and decoding), preliminary requirements for a universal light field format have been investigated [C4] [C5].

- Through a practical telepresence case study using 27 cameras, preliminary ideas are presented in a real-time capture and reconstruction setup for holographic light field representation.

3.4 Related Works

A precise representation of the captured data must support convenient processing, transmission and enable successful reconstruction of light field at the receiver side. The methods that will be discussed here are based on inferring the scene through a set of images (*multiview* images). This approach is less complex than the model based representation in the sense that we may neglect the complexity of the scene and the objects involved. Thus the complexity of the scene falls down to pixels.

DOI:10.15774/PPKE.ITK.2015.005

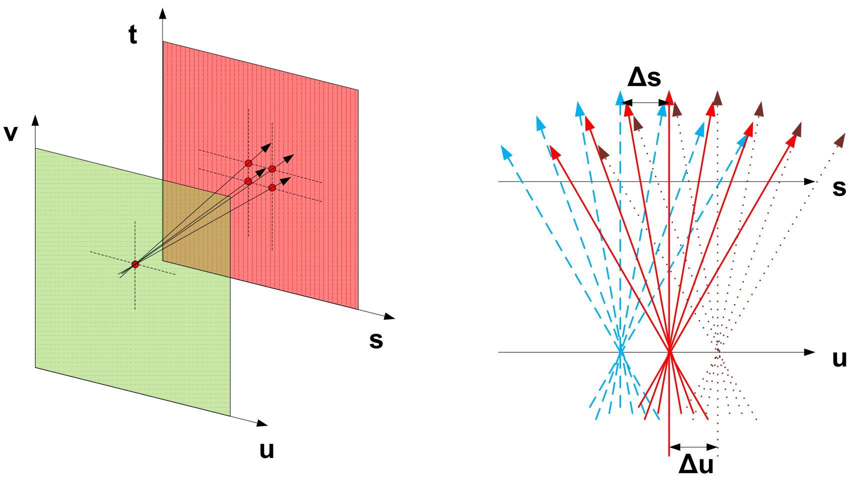


Figure 3.1: Two plane parameterization.

3.4.1 Two Plane Parameterization

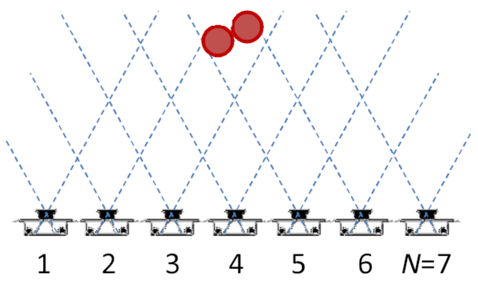
Imagine that we have two planes uv and st separated by a distance as shown in the left part of Figure 3.1. Light rays emerging out from each and every point on uv plane hit the st plane at different locations. Each ray in this representation is parameterized by four co-ordinates, two co-ordinates on the first plane and two co-ordinates on the second. This is simple and highly desirable representation because of its analogy with camera capturing plane and image plane. The right half of Figure 3.1. shows the sampling grids on two planes from top if vertical parallax is ignored.

An advantage of such representation is that it aids in the process of interpolation. This approach brings us the concept of epipolar images proposed by Bolles et.al., in 1987 [23] that supports frequency domain analysis and also enables convenient means for light field representation.

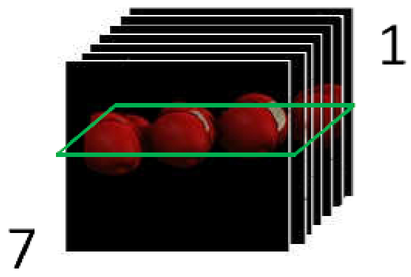
Epipolar Images

Let us assume a scene containing two circular objects as shown in Figure 3.2(a), captured using seven identical cameras. If these seven images are aligned along Z axis and a 2D slice is extracted in the XZ plane, the resulting image is known as epipolar image as illustrated in Figure 3.2(b). The number of rows in the epipolar image is equal to the number of capturing cameras. In this case we have a 7 row epipolar image. The number of columns in the image is equal to the width of camera sensor in pixels. Virtual view interpolation can be interpreted as adding additional rows to the epipolar image in the respective positions of images. If the pixels move exactly by one pixels across all the camera images, an epipolar image contains straight lines. For example if the same scene in Figure 3.2(a) is captured by 600 cameras, the blocks reduce to pixel level forming a line on the epipolar image. Thus each point in the scene appear as a line. The slope

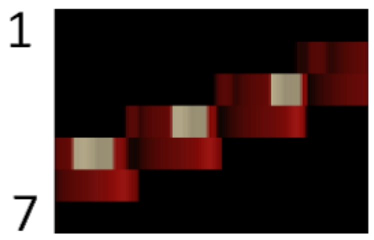
DOI:10.15774/PPKE.ITK.2015.005



(a)



(b)



(c)

Figure 3.2: Epipolar image. (a) shows a sample scene captured by 7 identical cameras. (b) shows the 7 captured camera images arranged in a 3D array; the green rectangle encapsulates a specific row from all the 7 images. (c) shows the epipolar image constructed using the chosen row encapsulates by green rectangle in (b). The width of the epipolar image is equal to the horizontal resolution of the cameras and the height of the epipolar image is equal to the number of cameras.

of the line is directly proportional to the depth of its corresponding point in space. Lines with least slope occlude the lines with larger slope. This is due to the fact that points close to camera occlude the ones farther away from the camera. Thus using an epipolar image, an irregularly structured scene is transformed to a regular and more uniform structure in the form of lines which is expected to make the disparity estimation process simple.

The Fourier transform of an epipolar image is band limited by two lines representing the minimum and maximum scene depth. That is the spectrum of a scene limited within the depth levels d_{min} and d_{max} occupies an area bounded by the lines with corresponding slopes on the epipolar image. The discrete Fourier transform using finite number of cameras with finite number of pixels results in periodic repetition of the spectrum along X and Y directions. The X repetition period is dependent on the camera density (number of cameras) and the vertical repetition period depends on the image resolution. Ideally if we have infinite number of cameras capturing infinite resolution images, then there will be no repetition in the spectrum at all. For successful reconstruction following the sampling theory, the base band signal must be extracted (filtered) without any aliasing. Given a base line, smaller the number of cameras, more will be the spectral aliasing making the signal reconstruction process tedious. Epipolar images enable the derivation of

DOI:10.15774/PPKE.ITK.2015.005

number of cameras we need to reconstruct the scene given a perfect Lambertian scene without occlusions in a given depth range.

A problem with two plane parameterization is that we need to have large number of camera images to render new views. If a horizontal line of an image is considered, the st and uv planes condense down to lines. Given this arrangement, it is hard to extract the rays that should converge in 3D space. In case of a sparse camera arrangement, multiple rays hitting the first row (st plane) at a single point do not contain any information about the 3D converging location. This problem stems back to the epipolar images. A different set of light field representations are proposed to overcome the problems and will be discussed in the following sub-sections.

3.4.2 Lumigraph

This technique is presented by Gortler et. al., (Microsoft Research) [24] and is an extension of the two plane parameterization. It incorporates an approximate geometry for depth correction. The main idea here is to sample and reconstruct a subset of plenoptic function called Lumigraph which has only four dimensions. A scene is captured along six capture planes in the pattern of cube faces using a hand-held camera. Using pre-known markers in the image, the camera position and pose are estimated which aid in designing an approximate scene geometry. Analogous to the computer generated models where the multiple ray casts can be integrated to form a light field, the captured image pixels act as a sample of plenoptic function. The re-sampled light field data is then used to generate arbitrary views.

3.4.3 Layered Depth Images (LDI)

The LDI representation was proposed by Jonathan Shade et.al., in 1998 [25]. The main contribution of their work lies in unifying the depth information obtained from several locations to a single view location by warping. This approach comes handy when dealing with occlusions. An LDI can be constructed by warping n depth images into a common camera view. Consider a case where we have three acquisition cameras, as shown in Figure 3.3. The depth images of cameras C2 and C3 are warped to the camera location C1. Thus referencing the camera C1, there can be more than one depth information along a single line of sight. During the warping process if two are more pixels are warped to the same co-ordinate of the reference location (in this case C1), their respective depth values (Z) are compared. If the difference in the depth is more than a pre-defined threshold, a new depth layer is added to the same pixel location in the reference image. If the difference in depth is less than a threshold, the average depth value is calculated and assigned to the current pixel. After successfully constructing the unified depth information, new views are generated at desired viewpoints.

Note that LDI construction does not need to be done every time we create a new view, which helps in obtaining an interactive camera motion. But an inherent problem with this approach is

DOI:10.15774/PPKE.ITK.2015.005

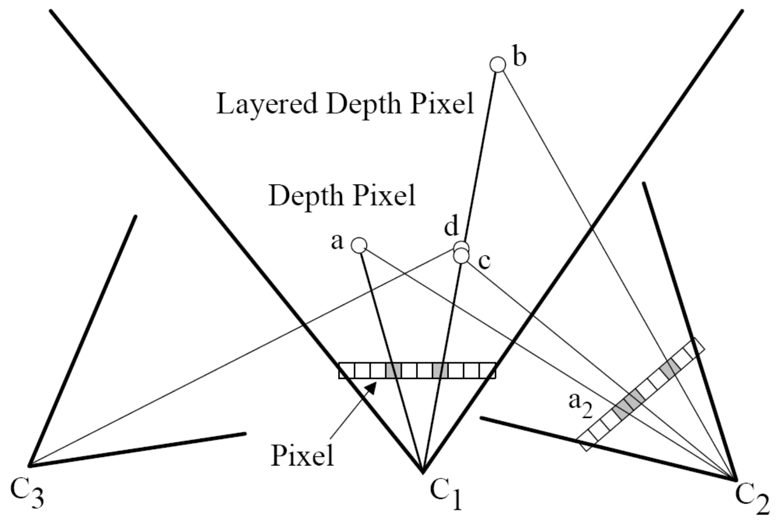


Figure 3.3: Layered Depth Image construction.

the error propagation. If there is a problem by estimating depth from one view location, it shows artifact in the unified depth data, which propagates at a later stage through the whole chain of generating a LDI.

3.4.4 Layered Lumigraph with Level Of Detail (LOD) Control

This is an extension to the Lumigraph and LDI representations [26]. Here instead of calculating the multiple depth layers from one reference location, they proposed doing the same from multiple reference points which increases the amount of data to be handled but is expected to improve the interpolation quality. They also incorporate the approximate geometry estimation following the Lumigraph approach. This approach is capable of producing very good interpolation results if a precise depth information is available.

3.4.5 Dynamically Re-Parameterized LFs

This work is aimed to address the focusing problems in under-sampled light fields. They used multiple cameras organized in to a grid forming a camera surface plane (C) [27]. A dynamic focal surface F represents the plane where the converging light rays from all the cameras are rendered. Each light ray is parameterized by four variables representing the ray origin on the camera surface and the destination on the Focal surface. Given a ray they try to find the rays from all the cameras on the camera surface which intersects the focal plane at the same location as the initial ray. This allows the dynamic focal plane selection to render images at different depth of fields.

DOI:10.15774/PPKE.ITK.2015.005

3.4.6 Unstructured light field (Lumigraph)

The idea of unstructured light field representation is to allow the user to capture images from a handheld camera along horizontal and vertical directions from various viewpoints to generate the light field [28]. The main contribution is that the problem of achieving dense coverage to reconstruct good quality light field is addressed. They use a criteria called re-projection error in all the four directions $(+X, +Y, -X, -Y)$ of the Cartesian coordinate system to estimate the new positions and in turn provide feedback to the user in real time, the information on the next location to capture. In addition, a triangulation approach is considered for obtaining smooth reconstruction. A problem with this approach is that it does not involve any depth computation and it is not possible to render sharp images. There will be always blurring artifacts and we may need infinitely many camera images to render sharp images.

3.4.7 Epipolar Plane Depth Images (EPDI)

This work presents an approach to generate views for free view point television. The main idea is to use *MultiView plus Depth* (MVD) images and extract the disparity information precisely from the epipolar images depending on the available approximate depth and intensities [29]. Although this approach is simple, it suffers from the proxy depth artifacts (for example if we have similar colored objects at slightly different depth levels).

3.4.8 Surface Light Field

This work mainly addresses the problem of rendering shiny surfaces at virtual view locations under complex lighting conditions [30]. They consider images acquired from multiple cameras and in addition they also get the surface texture information using a laser scanner. This approach generates an enormous amount of data and they present an approach for simultaneous compression of data after acquisition and before saving it in to the memory using generalized vector quantization and principal component analysis. They assign different intensities/colours depending on the scene to the light rays emerging out of single point on the recorded surface. They also present an approach for interactive rendering and editing of surface light fields. Editing phase involves simple filtering operations.

3.4.9 Layer Based Sparse Representation

The layer based sparse representation was proposed by Andiry Gelman et. al., in 2012 [31]. It relies on the segmentation in image domain on multiview images simultaneously. They take in to account the camera setup and occlusion constraints when doing the segmentation. The redundant data which is subset in all the segmented images is filtered out using the corresponding epipolar images. These layers when aligned along Z, produce a 3D model from multiview images. They

DOI:10.15774/PPKE.ITK.2015.005

also presented wavelet based compression technique to efficiently compress the source layer and the corresponding layers containing the meta data.

3.4.10 Light Field Data Acquired From Circular Camera Arrangement

Arranging multiple cameras around the scene in the form of a circle [32] or sphere is a slightly different representation compared to the two plane parameterization. Circular arrangement of cameras reduces the edge distortions as more number of cameras capture a given edge in contrast to the linear camera arrangement. For each of the camera images, the depth information is calculated and new views are rendered.

3.4.11 State-of-the-art Overview

The aforementioned state-of-the-art techniques address the problem of light field representation by analyzing the captured scene assuming a camera setup. The research trends show that approaches that consider scene depth explicitly to derive a representation are more practical. Although solutions such as Lumigraph or Layered Depth Images support depth based representation, they cannot provide an optimal means of representation for the use case of projection-based light field displays, as the reconstruction of light field is also dependent on the target display geometry. Thus it is important to take into account the light field conversion process that defines how the captured multiview light field is utilized for reconstruction.

3.5 Representing Holographic Light Field Content

3.5.1 Dependency on Given Display Architecture

Although the discussion is on HoloVizio light-field displays, the approach is directly applicable to any LF display that is driven by a distributed projection and rendering system. Considering the gap between pixel / light ray counts and the rendering capacity available in a single computer / GPU, using a distributed rendering system for these systems is a necessity today and in the foreseeable future. Therefore LF displays are typically driven by multiple processing nodes. Light rays leaving the screen spread in multiple directions, as if they were emitted from points of 3D objects at fixed spatial locations. However, the most important characteristic of this distributed projection architecture is that the individual projection modules do not correspond to discrete perspective views, in the way views are defined in a typical multiview setting. What the projection modules require on their input depends on the exact layout of the light field display, but in general, a single projection module is responsible for light rays emitted at different screen positions, and in different directions at all those positions. The whole image projected by a single projection module cannot be seen from a single viewing position. As such, one projection module represents

DOI:10.15774/PPKE.ITK.2015.005

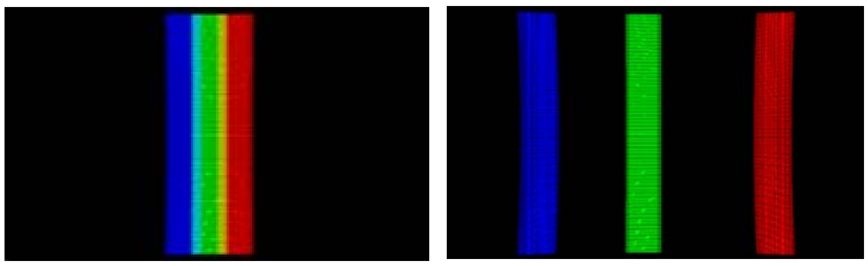


Figure 3.4: Left: Pixels required by processing nodes 4, 5, 6 (Red, Green and Blue channels). Right: Pixels required by processing nodes 0, 5, 9 (Red, Green and Blue channels).

a *light field slice*, which is composed of many image fragments that will be perceived from different viewing positions.

Although these LF slices can be composed based on the known geometry of a multi-camera setup and the geometry of the LF display, this mapping is nonlinear and typically requires accessing light rays from a large number of views, even when generating the image for a single projection module. The layout of the typical rendering cluster, made up of processing nodes (nodes for short), is such that a single computer is attached to multiple projection modules (2, 4, 8 or more), and as such, a single computer is responsible for generating adjacent LF slices. During LF conversion, individual nodes do not require all the views, nor all the pixels from these views. Although there is some overlap between the camera pixels required by nodes, those that are responsible for distant parts of the overall light-field require a disjoint set of pixels from the camera images.

To demonstrate this arrangement visually, Figure 3.4 shows which parts of the input perspective views are actually required for generating specific LF slices. A simulation has been run on a 45^0 large-scale light-field display with 80 projection modules, which has 10 processing nodes for generating the light-field. The display has been fed with 91-view input. What we can see is that adjacent processing nodes use adjacent, somewhat overlapping parts of the views, while processing nodes that are further away in the sense of LF slices will require completely different parts of the same view to synthesize the light field. These results are shown for the central camera, the pattern for other views is similar.

Two general use cases are defined to evaluate the applicability of specific 3D video coding tools, as the requirements imposed by these use cases are substantially different. The use cases identified by MPEG can be classified into one of these cases, depending on whether the content is stored / transmitted in a display-specific or display-independent format. In both use cases, the requirement for real-time playback (as seen by the viewers) is above all other requirements.

The first and least demanding use case is playback of the pre-processed LF content. In this case content has been prepared for a specific LF display model in advance, and must be played back in real time. In this setting the content is stored in display specific LF format. Display specific LF means the light rays are stored in a way that the individual slices of the full LF already correspond

DOI:10.15774/PPKE.ITK.2015.005

to the physical layout (projection modules) of the display on which the content should be played back. In other words, the LF in this case has already gone through the ray interpolation step that transforms it from camera space to display space. The implication is that the LF slices correspond to the layout of the distributed system driving the LF display, and as such, no ray interpolation is needed during playback, and no image data needs to be exchanged between nodes. As an example, in case of an 80-channel LF display, we may consider this data to be 80 separate images or videos making up a 3D image or video, for example 80 times WXGA (approx. 78 MPixels).

The second use case we consider is broadcast LF video transmission, with the possibility to target different LF displays. 3D LF displays can differ in multiple properties, but spatial resolution and FOV have the most substantial effect on the content. The goal is to support different LF displays with the same video stream in a scalable way. In order to support different displays, we need to use display independent LF, which is not parameterized by display terms, but using some other terms (for example capture cameras), which is subsequently processed on the display side during playback. In this paper we consider this display independent LF to be a set of perspective images representing a scene from a number of viewpoints. Please note there are many other device-independent LF representations which lay between these two, however these two are the closest to practical hardware setups (camera rigs and LF displays). The analysis that follows focuses on the decoder / display side, and does not consider encoder complexity.

3.5.2 Processing Display-Specific Light fields

In this case, as LF preprocessing is performed offline, the encoding process is not time critical, i.e. there is no real-time requirement for the encoder. Visual quality should be maximized wrt. bitrate, to be able to store the largest amount of LF video. On the decoding side, the goal is to be able to decompress separately the LF slices that correspond to the individual projection engines contained in the display, in real-time. The simplest solution to this problem is simulcoding all the LF slices independently using a 2D video codec (ie. H.264), and distribute the decoding task to the processing nodes corresponding to the mapping between processing nodes and projection engines. Take 80 optical engines and 10 nodes as an example: if all nodes are able to decompress 8 videos in real-time, simultaneously, we have a working solution (provided we can maintain synchronized playback). The complexity of H.264 decoding typically allows running several decoders on a high-end PC, and 25 FPS can be achieved. This solution is currently used in production LF displays. However, in this case we do not exploit similarities between the LF slice images which have similar features, like multiview imagery. On the other extreme, compressing all 80 LF streams with MVC would require that a single processing node can decompress all of them simultaneously in real-time, which is typically prohibitive. The complexity of MVC decoding is expected to increase linearly with the number of views in terms of computing power. Furthermore it also requires a larger Decoded Picture Buffer (DPB) depending on the number of views. Assuming that having enough RAM for the DPB is not an issue, decoding a 80-view

DOI:10.15774/PPKE.ITK.2015.005

MV stream on a single node in real-time is still an issue, especially as there is no real-time implementation available that can perform this task. Even considering parallelization techniques [33], decoding all views in real-time on a single node is out of reach. A reasonable tradeoff is to compress as many LF module images that are mapped to a single processing element, and do this as many times as necessary to contain all the views. As an example, we may use 10 separate MVC streams, each having 8 LF slices inside. We can increase the number of views contained in one MVC stream as long as a single processing node can maintain real-time decoding speed.

3.5.3 Processing Display-Independent Light Fields

As discussed earlier, not all views are required for interpolating a specific LF slice, and even from these views, only parts are required to generate the desired LF slice - some regions of the camera images might even be left unused.

Table 3.1: Number of views used overall for LF synthesis when targeting LF displays with different FOV.

FOV (degrees)	27	38	48	59	69	79	89
No. views used	42	44	46	48	50	52	54

To find out how much we can bound the number of views and pixels to be compressed, we may determine the images and image regions which are actually used during the LF interpolation process, and compress only those selected regions for the targeted display. However, assuming receivers with displays with different viewing capabilities makes such an approach impractical, and requires scalability in terms of spatial resolution and FOV. Difference in spatial resolution might be effectively handled by SVC, and is not discussed further here. The differences in FOV however have not been addressed, as studies on the effect of display FOV on the source data used for LF conversion have not been performed so far. We have performed simulations to see how the FOV of the receiver's LF display affects the way the available captured views are used. We have modeled 7 hypothetical LF displays, with the FOV ranging between 27^0 and 89^0. Source data with 180 cameras, in a 180^0 arc setup, with 1 degree angular resolution has been used. Using the tool from [C3] and analyzing the pixel usage patterns, we have analyzed how the display's FOV affects the number of views required for synthesizing the whole LF image. This analysis has shown that depending on the FOV of the display, the LF conversion requires 42 to 54 views as input for these sample displays, as seen in Table 1. Please note the actual number depends on the source camera layout (number and FOV of cameras), but the trend is clearly visible. Looking at the images representing the pixels read from each view also reveals that for most views, only small portions of the view are used, which is especially true for side views. This can be intuitively seen if we consider a 3D display with a wide viewing angle, looking at the screen from a steep angle. In this case, we can only see a narrow image under a small viewing angle - this is also what we need to capture and transmit. This observation suggests that any coding scheme targeting multiview video on LF displays should be capable of encoding multiple views with different

DOI:10.15774/PPKE.ITK.2015.005

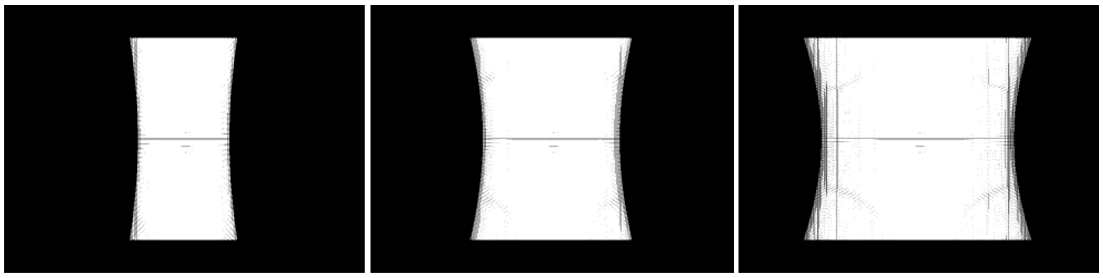


Figure 3.5: Image regions used from the central camera, by the 27^0 (left), 59^0(center) and 89^0 (right) LF displays.

resolution. In case of HPO LF displays, only the horizontal resolution changes. In full parallax setups, both horizontal and vertical resolutions change. Such flexibility is not supported by MVC.

Due to the fact that distributed processing nodes are responsible for different parts of the overall LF, these units require different parts of the incoming views. Thus we may expect that the number of views necessary for one node is lower than for the whole display. Further analyzing pixel usage patterns and separating the parts required by distinct nodes, we can see that this number is indeed lower, however not significantly lower. For example, in case of the 89^0 FOV display, instead of the 54 views required for the whole LF, one node requires access to 38 views on average, which is still high - decompressing these many full views is a challenge. As seen previously, not all pixels from these views are necessary to construct the LF. If we look at the patterns showing which regions of the views captured by the cameras are used for the LF conversion process when targeting LF displays with different FOVs, we can see that the area is pointing to the scene center, and is widening with the increased FOV, see Figure 3.5. This property may be used to decrease the computational complexity of decoding many views, by decoding only regions of interest for the specific display. H.264 supports dividing the image into regions to distinctly decodable regions using slice groups, however this feature is typically targeted to achieve some level of parallelism in the decoding process. By defining individually decodable slice groups that subdivide the image into vertical regions, and decoding only those required, it is possible to decrease the time required to decode the views. Defining several slice groups would give enough granularity to target a wide range of displays with little overhead. On the other hand, by separating views into vertical slices, we lose some coding gain due to motion estimation / compensation not going across slice boundaries. Some of this loss might be recovered by using prediction from the center of views to the sides, however such hierarchies are not supported. Exploiting this possibility is an area of future research (for more details related to display specific and display independent light field processing, please refer: [C4]).

DOI:10.15774/PPKE.ITK.2015.005

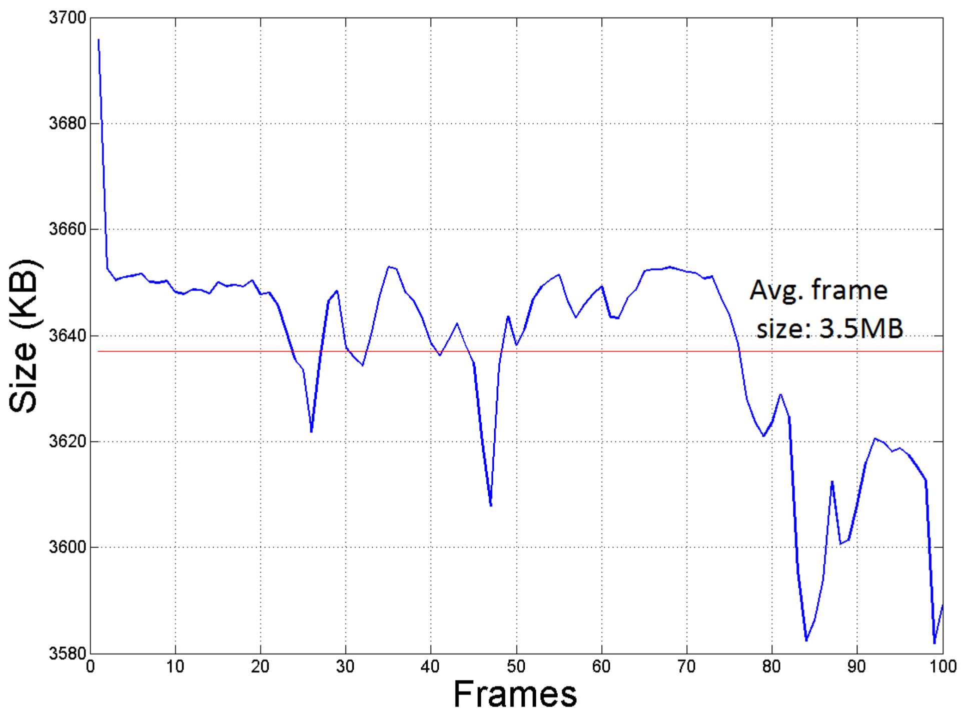


Figure 3.6: Amount of information produced by all the cameras per frame (JPEG compressed) in a sample capture.

3.6 Preliminary Ideas in a Telepresence Scenario

Here, concerning the telepresence scenario, a method is proposed by which we can reduce the data from each of the camera images by discarding unused parts of the images at the acquisition site in a predetermined way using the display model and geometry, as well as the mapping between the captured and displayed light field. The proposed method is simple to implement and can exclude the unnecessary data in an automatic way. While similar methods exist for 2D screens or display walls, this is the first such algorithm for light fields. The experimental results show that an identical light field reconstruction can be achieved with the reduced set of data which we would have got if all the data were transmitted. Real-time light field capture and rendering engine with 27 horizontally aligned cameras is used for experiment (see section 2.7.3). Glasses-free 3D cinema system developed by Holografika is used for rendering (see [C2]). Taking into account a light field display model and the geometry of captured and reconstructed light field, automatic data picking procedure is devised.

Figure 3.6 shows the amount of information generated by all the cameras at different time instances in an experimental capture, hereafter will be referred to as *raw 3D frames*. Each camera generates JPEG compressed RGB images of resolution 960×720 and the average amount of information is generated per raw 3D frame is 3.636 MB. If 25 frames are generated per second, the

DOI:10.15774/PPKE.ITK.2015.005

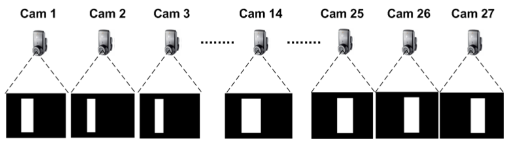
average amount of information to be transmitted in a second is approximately 91 MB. Currently, a single computer is capturing images from all the cameras and the effective 3D frame rate achieved is 10fps. In this case, the amount of information to be transmitted in a second would be 36.36 MB.

Let us consider a telepresence scenario in which the information is captured locally and sent to a remote site on a wired Gigabit Ethernet connection. We have the multi-camera setup together with the acquisition and application node as mentioned earlier, light field conversion inherently refers re-ordering the camera image pixels in order to suit the display 3D projection geometry. Depending the projection module composition group, each node in the render cluster efficiently implements this pixel re-ordering on GPU in real-time. As each rendering node drives a set of projection modules, depending on the location of this projection module group, the node makes use of pixels from many cameras. However, for a single node, it is not necessary to have all the data from all the cameras. As the cameras are horizontally aligned they capture information from slightly different perspectives and thus it is more likely to have redundant information which might not be used during the process of light field reconstruction. The first step in the proposed approach is to locate these unwanted pixels. Given the camera calibration and display geometry pixel re-ordering fashion remains identical temporally. This makes it possible to estimate the unwanted data during light field reconstruction. To facilitate data reduction, the pixel light ray mapping calculations are made locally at the capture side before the start of transmission and we safely store the information on appropriate pixel co-ordinates in the form of masks. Figure 3.8 shows the whole process. At each time instant, from the incoming camera images, we extract the useful information carefully using the pre calculated masks (as shown in Figure 3.7(a)). Figure 3.7(b) shows the amount of information used by individual camera in the experiment. It can be seen on an average, 80% of the data can be reduced using the given camera and display setup. Although this step itself reduces the size of data to be transmitted considerably, it is still possible to achieve significant data reduction by incorporating video compression schemes. The masked pixels can be extracted and stitched into a single 2D frame for transmission.

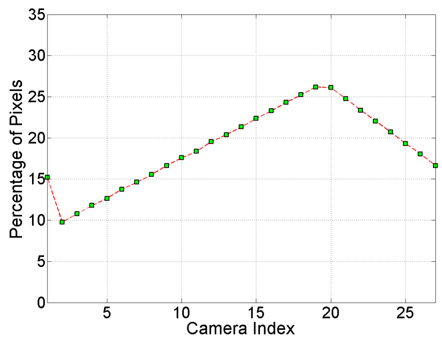
3.7 Summary of the Results and Discussion

This chapter describes how the light field conversion can be explored for a given target display geometry to reduce the amount the data needed for light field transmission and discusses practical approaches towards achieving a universal light field format. The following two subsections brief the requirements for the capturing and encoding systems for future projection-based light field display systems.

DOI:10.15774/PPKE.ITK.2015.005



(a) Sample binary camera masks



(b) Amount of consumed data from individual cameras during light field reconstruction

Figure 3.7: Light field data analysis and sample masks.

3.7.1 Capture Arrangement

The several simulation results obtained by varying the display FOV and checking the utilized camera pixels show that cameras arranged in arc better serve the projection-based light field displays. With the emergence of LF displays with extremely wide FOV, it is more and more apparent that an equidistant linear camera array cannot capture the visual information that is necessary to represent the scene from all around. Also employing *Multiview Video Coding (MVC)* directly should be efficient on arc camera images as the views captured in this manner also bear more similarity than views captured by a linear camera array. However, the kind of pixel-precise inter-view similarity that *MVC* implicitly assumes only exist when using parallel cameras on a linear rig, and assuming Lambertian surfaces. It has been shown that the coding gain from inter-view prediction is significantly less for arc cameras than for linear cameras.

Due to the emergence of wide-FOV 3D displays it is expected that non-linear multiview setups will be more significant in the future. Coding tools to support the efficient coding of views rotating around the scene center should be explored, and the similarities inherent in such views exploited for additional coding gains.

3.7.2 Requirements for a Light Field Encoder

Based on the use cases and processing considerations, we can formulate at least three aspects that need attention and future research when developing compression methods for LFs. First, we shall add the possibility to encode views having different resolution. Secondly, the ability to decode the required number of views should be supported by the ability to decode views partially, starting from the center of the view, thus decreasing the computing workload by restricting the areas of interest. Third, efficient coding tools for nonlinear (curved) camera setups shall be developed, as we expect to see this kind of acquisition format more in the future.

DOI:10.15774/PPKE.ITK.2015.005

3.7.3 Evaluation of the Preliminary Ideas Using H.264 Codec

To evaluate the light field data representation and reduction ideas for telepresence practical use case discussed in section 3.6, let us consider a simple and straight forward approach of encoding the utilized camera pixel data using well known H.264 Codec. The reason behind choosing this codec is that it is an open source software that supports zero latency encoding and decoding and is currently one of the most commonly used formats for the recording, compression, and distribution of high definition video.

For encoding, I integrated the extracted patches from several camera images which are useful in the light field reconstruction process and make a single 2D frame constituting multiple camera image patches (hereafter will be referred to as raw 3D frame). To speed up the memory access (rows of an image are stored in consecutive memory locations) while creating a raw 3D frame, these images are first transposed and integrated in to a raw 3D frame vertically. The idea behind the integration of patches is to explore and make use of the potential of legacy 2D video compression standards in transmitting raw 3D frames. This also allows fair comparison as the originally available camera data is also encoded using *MJPEG*, which is also a 2D compression scheme embedded into the hardware of the capturing cameras. For transmission, the raw 3D frames at each time instant are packetized using H.264 codec and sent to the receiver side.

The encoded packets are broadcasted to all the nodes in rendering cluster on the receiving side. Each node decodes the packets it receives using corresponding decoder and uses the decoded raw 3D frame to produce a set of projection module images that it drives. Note that the bounding box information on useful pixels from each camera image is made available at the remote site beforehand to pick out the individual camera images.

Figure 3.9 presents the data sizes after the initial (reduced data after analying light field reconstruction) and final stages (encoding the reduced data using H.264) of the presented approach. The curve in blue shows the size per raw 3D frame sent in the form of multiple streams on multiple ports. Comparing this with Figure 3.6, it can be clearly stated that there is approximately 70% of bandwidth saving at the first step. With patch integration and single stream H.264 encoding, the average size per frame is further reduced. The codec settings were adjusted after trial and error to preserve the quality of decoded image. We found that the compression ratio of 9.81:1 would be enough for the codec to maintain the visual quality of light field rendering. Using this compression ratio, the average size of a raw 3D frame is calculated to be 42 KB and thus bandwidth saving is further increased by approximately 28%. Note that the codec settings were adjusted to emit one intra frame for every 25 frames which is the reason for the periodic peaks in the H.264 single stream curve. Results show that using the presented approach, 25fps raw 3D video can be comfortably streamed on a 10 Mbps line. The average connected speed needed is calculated to be 8.2Mbps.

Figure 3.10 shows the PSNR in dB of the decoded raw 3D frames. Using the presented approach we could achieve an average PSNR of 36.38 dB., which proves the maintenance of significant

DOI:10.15774/PPKE.ITK.2015.005

visual quality.

3.7.4 Applicability of the Proposed Ideas to Other Display Systems

As mentioned earlier, the proposed data reduction ideas explicitly take into account the display geometry and the light field conversion process. The bit-rate reduction ideas are mainly drawn based on the observation that holographic light field reconstruction, depending on the capture and display geometry, may not need/utilize the complete captured information. In case of lens base autostereoscopic displays, these approaches and assumptions may not be valid as these displays project multiple 2D perspectives. However, for more advanced displays that are based on spatial and time multiplexing, such as tensor displays, these ideas can be interesting as those display systems tend to reconstruct actual light field rather than projecting the captured multiview images.

DOI:10.15774/PPKE.ITK.2015.005

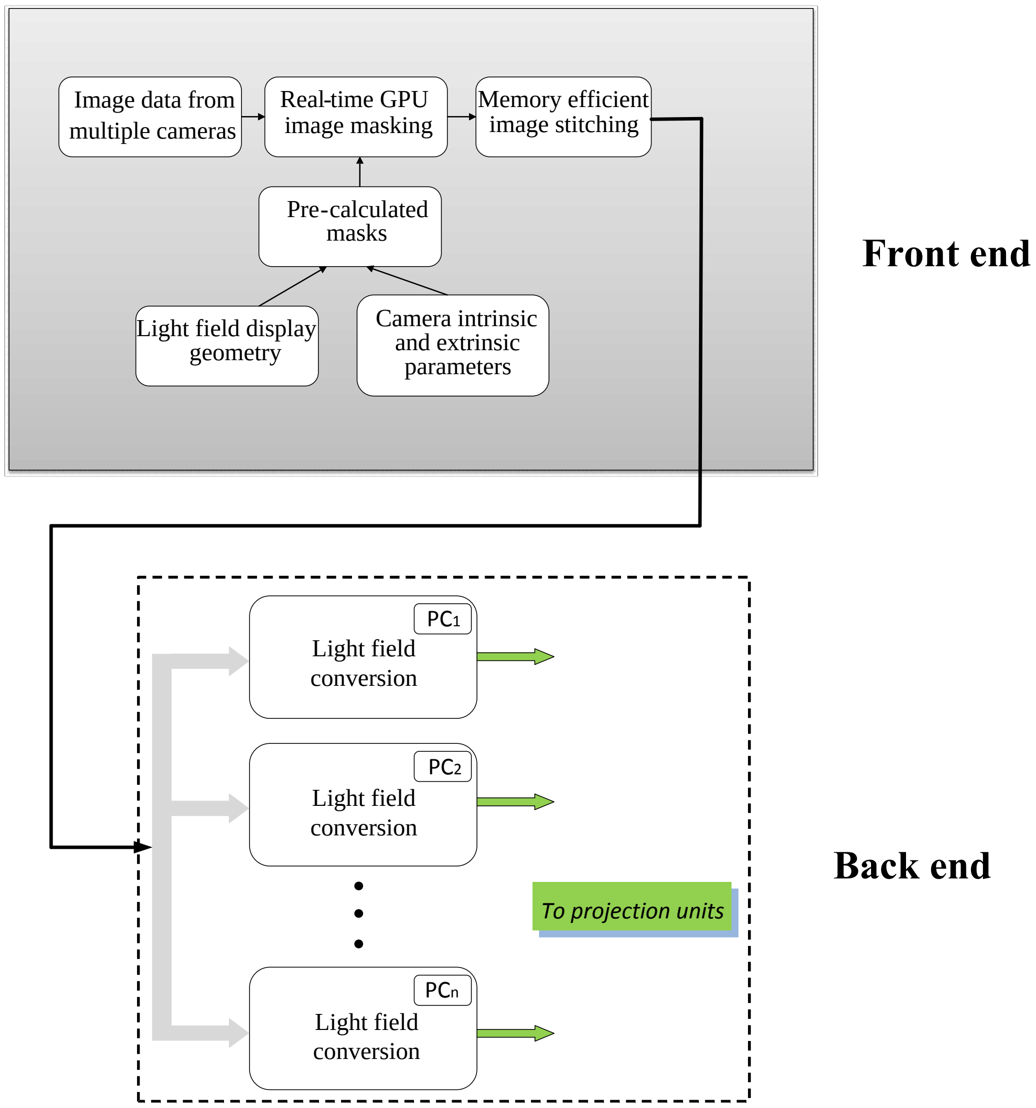


Image data from multiple cameras
Real-time GPU image masking
Memory efficient image stitching
Pre-calculated masks
Light field display geometry
Camera intrinsic and extrinsic parameters
Front end
PC1
Light field conversion
PC2
Light field conversion
To projection units
PCn
Light field conversion
Back end

Figure 3.8: Light field data reduction procedure system overview.

DOI:10.15774/PPKE.ITK.2015.005

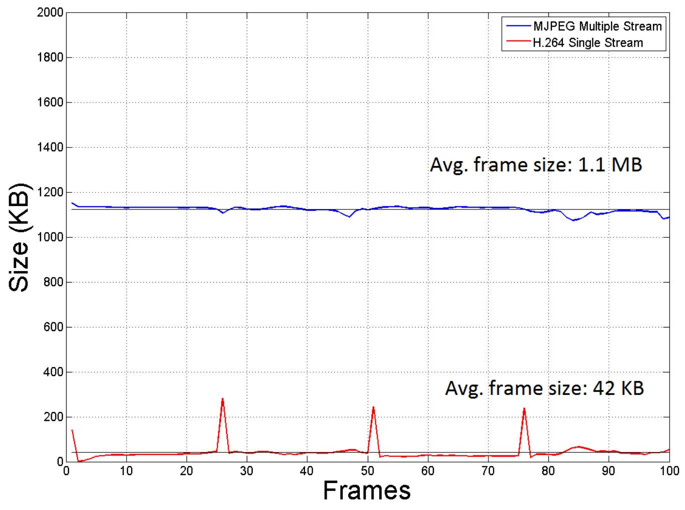


Figure 3.9: Data reduction results - Discarding the unwanted data that is not utilized in light field reconstruction before transmission resulted in approximately 70% of bandwidth saving using the current experimental setup (blue curve). Additionally, by using H.264, the average size of a single raw 3D frame comes down to 42KB (red curve).

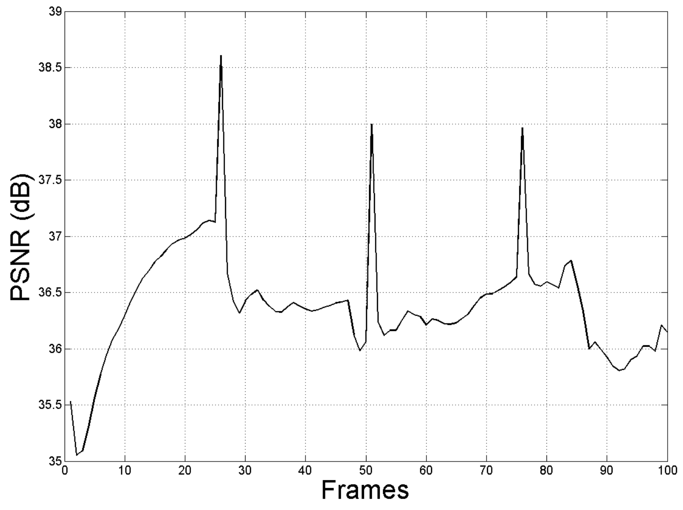


Figure 3.10: PSNR of H.264 decoded frames compared with the uncompressed frames. An average of 36.38 dB PSNR can be achieved using the proposed pipeline.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 4

Depth Retargeted Light Field Rendering

4.1 Introduction

As mentioned earlier, taking inspiration from the real-world, a light field display emits light rays from multiple perspectives using a set of optical modules. The various emitted light rays hit a holographic screen which performs the necessary optical modulation for reconstructing a 3D scene. Various scene points are described by intersecting light rays at corresponding depths. Even though these displays provide continuous views and improve over traditional automultiscopic solutions, the extent of practically displayable depth with reasonable 3D quality is still limited due to finite number of light generating optical modules. Scene points rendered outside this range are subjected to poor sampling and suffer from aliasing, which typically lead to excessive blurring in regions. This blurring makes it difficult to perceive details of very far objects from the screen, and leads to visual discomfort.

4.2 Depth Retargeting

By matching the depth extent of scene to that of display by applying a process of *depth retargeting*, it is possible to greatly reduce the blurring artifacts, achieving all-in-focus rendering. An important consideration while retargeting the light field depth is that any depth compression results in flattening of objects and distorting the 3D structure of the scene. Thus, in order to provide compelling results, depth compression must be non-linear and content-adaptive. In the current work, this problem of *depth retargeting* is addressed by proposing a low-complexity real-time solution to adaptively map the scene depth to display depth by taking into account the perspective effects of a light field display and the saliency of the scene contents. The proposed retargeting module, which strives to reduce distortions in salient areas, is integrated into a real-time light field

DOI:10.15774/PPKE.ITK.2015.005

rendering pipeline that can be fed with a live multi-view video stream captured from multiple cameras.

An architecture is proposed coupling the geometry estimation and retargeting processes to achieve the real-time performance. While rendering the light field, the renderer estimates input scene geometry as seen from the positions of various display optical modules, using only multiview color input. The estimated depth is used for focusing the light field and is the basis for adaptive depth retargeting. In order to compute an optimal scene deformation, a convex optimization problem is formulated and solved by discretizing the depth range into regions, and using saliency information from the scene to preserve the 3D appearance of salient regions of the scene in retargeted space. Scene saliency is computed by analyzing the objects distribution in the scene depth space and weighting this distribution with appropriate texture gradient magnitudes. During retargeting, scene points are subjected to a perspective transformation using the computed non-linear mapping which changes depths and accordingly scales x-y positions. The quality and performance of this retargeting approach is demonstrated in an end-to-end system for real-time capture and all-in-focus display that achieves real-time performance using 18 cameras and 72 projection modules. More details about the implementation details and the retargeting results are presented in the following paragraphs.

In particular, the improvements with respect to the state-of-the-art are the following:

- A perspective depth contraction method for live light field video stream that preserves the 3D appearance of salient regions of a scene. The deformation is globally monotonic in depth, and avoids depth inversion problems.
- A real-time plane sweeping algorithm which concurrently estimates and retargets scene depth. The method can be used for all-in-focus rendering of light field displays.
- An end-to-end system capable of real-time capturing and displaying with full horizontal parallax high-quality 3D video contents on a cluster-driven multiprojector light field display with full horizontal parallax.
- An evaluation of the objective quality of the proposed depth retargeting method.

The proposed method for depth retargeting is content-adaptive and computationally light. It is general enough to be employed both for 3D graphics rendering on light field display and for real-time capture-and-display applications. The content-adaptive nature of the method makes it possible to employ a number of different measures to determine which depth intervals should be preserved most. The method currently do not attempt to provide a model of the behavior of the human visual system to drive the optimization, and rather use a simple saliency estimator based on geometry and image gradients. The approach is general enough, however, to replace saliency estimation with more elaborate and domain-specific modules (e.g., face recognition in 3D video-conferencing applications).

DOI:10.15774/PPKE.ITK.2015.005

4.3 Related Works

The end-to-end system enhances and integrates several state-of-the-art solutions for 3D video capture and rendering in wide technical areas. For comprehensive understanding, the reader is referred to established surveys (e.g., [34, 35]). In the subsequent paragraphs, some of the more relevant works are presented.

4.3.1 Light Field Capture and Display

One of the aims while developing the retargeting alogorithm was to achieve the depth retargeting in real-time. In the current work, I followed the real-time approach of Marton et al. [9], which takes into account light field display characteristics in terms of both geometry and resolution of the reproduced light fields (see 2.7.3 for other real-time methods). In particular, they extend a multiple-center-of-projection technique [17, 16, 36] to map captured images to display space, and estimate depth to focus the light-field using a coarse-to-fine space-sweeping algorithm. In the proposed method, their approach is extended to embed a saliency aware depth retargeting step during depth evaluation to properly place the scene in the correct display range thus avoiding aliasing artifacts while maintaining correct depth for salient scene regions.

4.3.2 Adaptive Depth Retargeting

Content remapping is a well established approach for adapting image characteristics to limited displays, and is routinely used for adapting spatial and temporal resolution, contrast, colors, and aspect ratios of images. For the particular case of depth retargeting, Lang et al. [10] proposed a method for remapping stereoscopic 3D disparities using a non linear operator. The non-linear mapping is generated by sparse disparity estimation and combining the local edge and global texture saliency. The method is based on warping the stereo images independently to achieve depth retargeting. As the method relies on sparse disparities, warping can lead to noticeable artifacts especially near the depth discontinuities and may also distort any straight lines in the scene. Extending this method to remap full-parallax light field content would introduce artifacts because of the increased number of views. Kim et al. [37] extend the approach by proposing a framework for the generation of stereoscopic image pairs with per-pixel control over disparity, based on multi-perspective imaging from light fields. While their method might be extended to multiview images, the associated optimization problem is too costly to be solved in a run-time setting. Masia et al. [38] deal specifically with multiview displays by proposing a method for display-adaptive depth retargeting. They exploit the central view of a light field display to generate a mapping function and use warping to synthesize the rest of the light field. Their work strives to minimize perceived distortion using a model of human perception, but does not achieve real-time performance and does not include depth estimation. Piotr Didyk et al. [39] proposed a model for measuring perceived disparity and a way to automatically detecting the threshold for

DOI:10.15774/PPKE.ITK.2015.005

comfortable viewing. Their method can be used as a component to operate depth retargeting. In the current work, the concentration is instead on overcoming device limitations. Birkbauer et al. [40] handle the more general problem of light field retargeting, using a seam-carving approach. The method supports visualizing on displays with aspect ratios that differ from those of the recording cameras, but does not achieve real-time performance. Content-aware remapping has also been proposed to achieve non-linear rescaling of complex 3D models, e.g. to place them in new scenes. The grid-based approach of Kraevoy et al. [41] has also been employed for image retargeting. Graf et al. [42] proposed an interesting approach for axis-aligned content aware 2D image retargeting, optimized for mobile devices. They rely on the image saliency information to derive an operator that non-linearly scales and crops insignificant regions of the image using a 2D mesh. The proposed method also takes the approach of using a discretized grid to quickly solve an optimization problem. In this case, we can use a one-dimensional discretization of depth, which permits us to avoid depth inversion problems of solutions based on spatial grids.

4.4 Retargeting Model

If a 3D scene and display have the same depth extent no retargeting is required, but in a more general case, a depth remapping step is needed. The aim is to generate an adaptive non-linear transform from scene to display that minimizes the compression of salient regions. A simple retargeting function is a linear remapping from world depths to display depths: this transformation can be composed using scale and translation. This simple approach works fine, but squeezes uniformly all the scene contents. The proposed approach aims to minimize squeezing of salient areas while producing the remapping function. The retargeting function computation is a content aware step, which identifies salient regions and computes a correspondence to maps the 3D scene space to the display space. To extract the scene saliency, depth and color from perspectives of multiple display projection modules are computed and combine this information. To make the process faster, saliency is computed from central and two lateral perspectives and use this information to retarget the light field from all the viewing angles. Depth saliency is estimated using a histogram of the precomputed depth map (please refer to section 4.5 for details on how depths are computed). More specifically, the whole scene range is sweeped, starting from camera near plane to far plane along originally defined steps from the three perspectives and collect the number of scene points located in each step. This information is then combined for extracting depth saliency. To estimate color saliency, Gradient map of the color image associated to the depth map of the current view is computed and dilated to fill holes, as done in [42]. The gradient norm of a pixel represents color saliency.

To avoid any abrupt depth changes, the scene depth range is quantized into different depth clusters and accumulate the depth and color saliency inside each cluster. In real-world, objects far away from the observer appear flatter than the closer objects and thus impact of depth compression on closer objects is more than that of far objects. Taking into account this phenomena, weightes are

DOI:10.15774/PPKE.ITK.2015.005

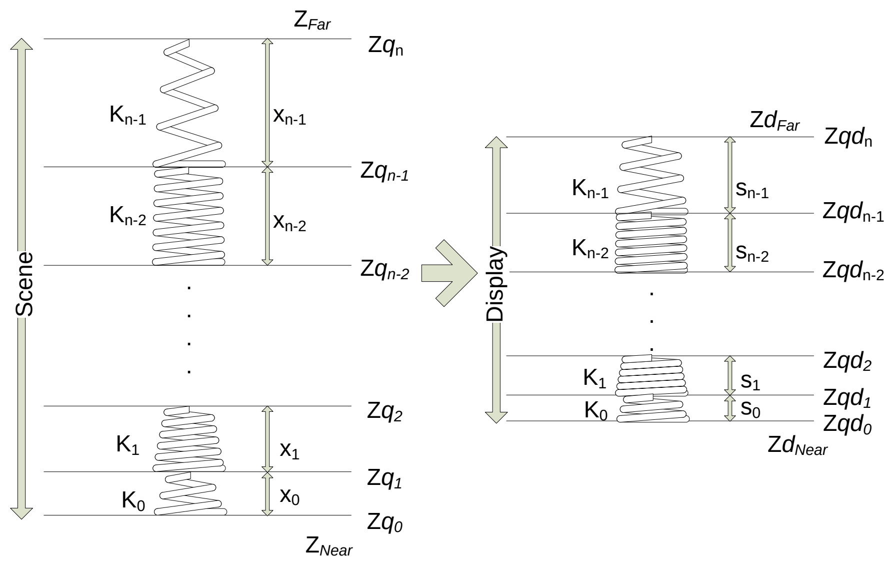


Figure 4.1: Computation of content aware depth retargeting function. Scene depth space is quantized into n clusters and saliency is computed inside each. Cluster sizes in the retargeted display space is computed by solving a convex optimization. Z_{qi} and Z_{qdi} denote i^{th} quantized depth level in scene and display spaces respectively.

also applied to the saliency of each cluster based on the relative position from the observer. Using the length of a cluster and it's saliency, a convex optimization is solved to derive the retargeting function.

4.4.1 Solving Retargeting Equation

The generation of retargeted light field is formulated as a quadratic optimization program. Let's assume that the scene range is quantized into n clusters and a spring is assigned to each cluster as shown in Figure 4.1 left. Let us denote the length and stiffness of a spring as the size and saliency of the representing cluster. Assuming that we compressed the n spring set within scene range to display range as shown in Figure 4.1, the resulting constrained springs define the desired new clusters in the display range which preserve the salient objects. To estimate the size of each constrained cluster, we define an energy function proportional to the difference between the potential energies of original and compressed spring. By minimizing this energy function summed over all the springs, we obtain the quantized spring lengths within the display space. Following the optimization of Graf et al. [42] for 2D image retargeting and adapting it to one-dimensional depth retargeting, The aim is to minimize:

$$\sum_{i=0}^{qn-1} \frac{1}{2} K_i \left(S_i - X_i\right)^2 \tag{4.1}$$

DOI:10.15774/PPKE.ITK.2015.005

subject to:

$\sum_{i=0}^{qn-1} S_i = D_d$

$S_i > {D_{cs}}^{min}, i = 0, 1, ..., n-1$ Where, X_i and K_i are the length and stiffness of i^{th} cluster spring, D_d is the total depth of field of the display and ${D_{cs}}^{min}$ are the minimum and allowable sizes of the resulting display space clusters. By expanding and eliminating the constant terms that do not contribute to the optimization, equation 4.1 can be re-written in the form of Quadratic Programming (QP) optimization problem as follows.

$$\frac{1}{2}\left(x^T G x + g_0 x\right) \tag{4.2}$$

subject to: $CE^T x + Ce_0 = 0$ & $CI^T x + Ci_0 \geq 0$
Where

$$G = \begin{pmatrix} K_0 & 0 & \cdots & 0 \\ 0 & K_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & K_{n-1} \end{pmatrix}$$

$g_0 = [-2K_0X_0 - 2K_1X_1 \cdots - 2K_{n-1}x_{n-1}]$

$$x = \begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_{n-1} \end{pmatrix}; CE = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}; Ci_0 = \begin{pmatrix} -{D_{cs}}^{min} \\ -{D_{cs}}^{min} \\ \vdots \\ -{D_{cs}}^{min} \end{pmatrix}$$

$CI = I_(nxn); Ce_0 = -D_d$

CE and Ci_0 are $(nX1)$ vectors.

For each point in the scene, a new point in the display $z_{display} = f(z_{scene})$ is computed using piecewise linear interpolation. It is important to note that while adapting the scene to display, displacement of depth planes parallel to $XY = 0$ plane results in XY cropping of the scene background. Thus, in order to preserve the scene structure, a perspective retargeting approach is followed, i.e., along with z update XY position proportional to $\frac{1}{\delta Z}$ is also updated, as it is done in a perspective projection (see Figure 4.2). Thus in the retargeted space, the physical size of the background objects is less than the actual size. However, a user looking from the central viewing position perceives no change in the apparent size of the objects as the scene points are adjusted in

DOI:10.15774/PPKE.ITK.2015.005

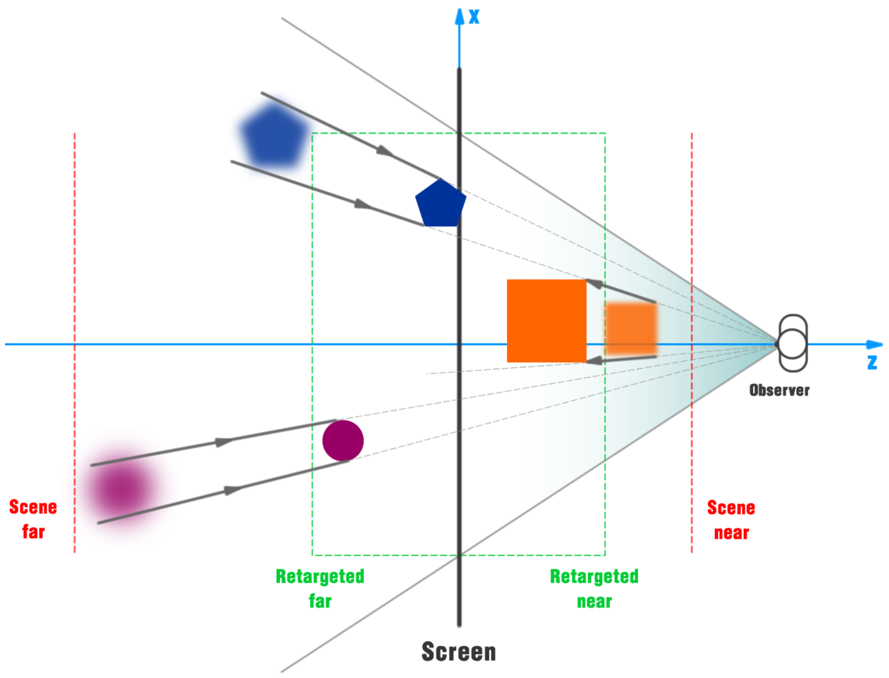


Figure 4.2: Perspective Content adaptive retargeting - Objects are replaced within the display space with minimum distortion, trying to compress empty or not important depth ranges. Objects are not simply moved along z, but the xy coordinates are modified by a quantity proportional to $\frac{1}{\delta Z}$.

the direction of viewing rays.

4.4.2 Calculating Scene Rays from Display Rays Using Retargeting Function

During retargeting, we modify the scene structure, which causes that the unified camera and display coordinate system that is used for rendering is no longer valid. Due to the adaptivity of the proposed algorithm to the scene depth structure, the retargeting function is different for every new frame. Thus it is not possible to assume a fixed transformation between scene and display rays. Also depending on the saliency, within the same frame that is being rendered, the transformation from display rays to scene rays is not uniform allover the 3D space. This section presents discusses more details on how camera rays are calculated from display rays.

Let us consider a sample scene (in light blue color) as shown in Figure 4.3 which we want to adaptively retarget to confine the scene depth to displayalbe depth (in light red color). Note that the camera attributes in Figure 4.3 are shown in blue and display attributes are shown in red. The pseudocode to calculate the scene rays from display rays is given in Algorithm 1. For a given display optical module, depending on the position in current viewport, we obtain the current display ray origin and direction based on the display geometry calibration and holographic transformation. For the display light ray, using all-in-focus rendering approach, depth value at which the color must be sampled from nearest cameras is calculated (depth calculation is

DOI:10.15774/PPKE.ITK.2015.005

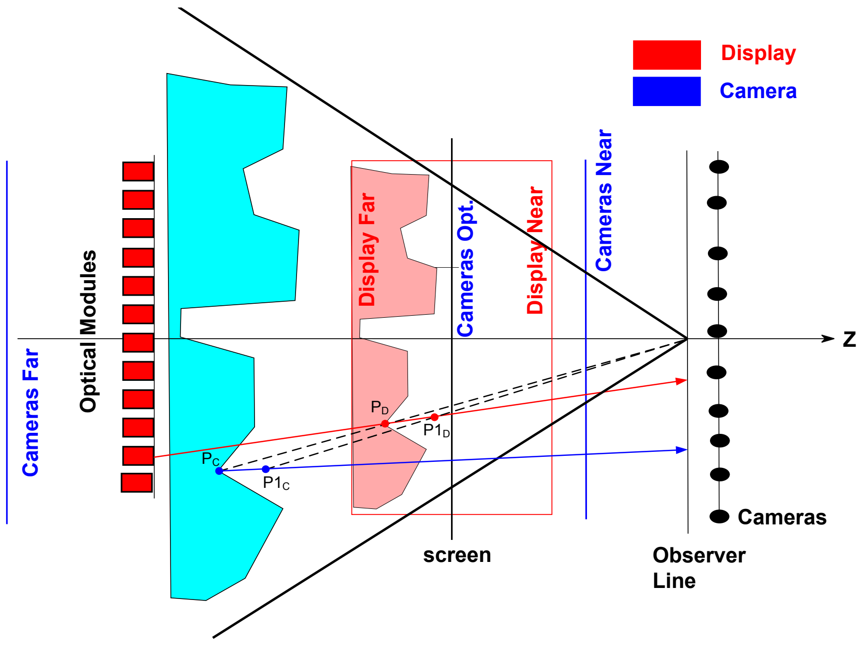


Figure 4.3: obtaining camera rays from display rays. After computing depth of a current display ray, the depth value is transformed to camera space using inverse retargeting operator. Later a consecutive depth value in display along the display ray is also transformed to camera space. The ray joining the two transformed points in camera space is used for calculating the color that should be emitted by the current display ray.

discussed more in the next section). Using the inverse retargeting operator, the calculated depth value in the display space is transformed into the camera space. During depth calculation, the display space is non-linearly subdivided into number of discrete depth steps. Considering the immediate depth plane towards the observer after the depth plane of the current display ray, we obtain a new 3D coordinate along the current display ray at the consecutive depth level. The new coordinate is transformed into the camera space from display space in a similary way using the inverse retargeting function. The ray connecting the two scene space points is used as the camera ray to interpolate the color information from the set of nearest cameras.

4.5 End-to-end Capture and Display System Implementation

The retargeting method is simple and efficient enough to be incorporated in a demanding real-time application. In the proposed method, the optimization is solved on CPU. On a Intel Core i7 processor with 8GB internal memory, the optimization can be solved at 60 *fps* to generate a non-linear mapping to display. While this use is straightforward in a 3D graphics setting, where retargeting can be implemented by direct geometric deformation of the rendered models, in this section, the first real-time multiview capture and light field display rendering system incorporating the adaptive depth retargeting method is introduced. The system seeks to obtain a video stream

DOI:10.15774/PPKE.ITK.2015.005

Algorithm 1 Calculate scene (camera) rays from display rays

```
1: $RT \leftarrow$ retargeting operator from scene to display from spring optimization
2: $P \leftarrow$ total number of display optical modules
3: $Vh \leftarrow$ height of the viewport
4: $Vw \leftarrow$ width of the viewport
5: for <displayModule ← 1 to P> do
6:     for <viewPortCordinateX ← 1 to $Vh$> do
7:         for <viewPortCordinateY ← 1 to $Vw$> do
8:             $R_O \leftarrow currentDisplayRayOrigin$
9:             $R_D \leftarrow currentDisplayRayDirection$
10:
11:            $R_{DP} \leftarrow currentDisplayRayDepth$
12:            $P_D \leftarrow R_O + R_D \times R_{DP}$
13:            $P_C \leftarrow INV(RT) * P_D$
14:
15:            $R1_{DP} \leftarrow displayRayDepthAtNextDepthStep$
16:            $P1_D \leftarrow R_O + R_D \times R1_{DP}$
17:            $P1_C \leftarrow INV(RT) * P1_D$
18:
19:            $rayOrgInCamera \leftarrow P_C$
20:            $rayDirInCamera \leftarrow P1_C - P_C$
21:        end for
22:    end for
23: end for
```

as a sequence of multiview images and render an all-in-focus retargeted light field in real-time on a full horizontal light field display. The input multiview video data is acquired from a calibrated camera rig made of several identical off the shelf USB cameras. The baseline length of the camera array is sufficiently chosen to meet the display FOV requirements. The captured multiview data is sent to a cluster of computers which drive the display optical modules.

Each node in the cluster drives more than one optical module. Using the display geometry and input camera calibration data, each node estimates depth and color for corresponding light rays. As mentioned before, to maintain the real-time performance, the depth estimation and retargeting processes are combined. The overall system architecture is shown in Figure 4.4. The implementation details are elaborated in the following paragraphs.

4.5.1 Front End

The front end consists of a master PC and the capturing cameras. The master PC acquires video data from multiple software-synchronized cameras and streams it to several light field clients. The data is acquired in JPEG format with VGA resolution (640 X 480) at 15Hz over a USB 2.0 connection. At a given time stamp, the several captured multiview images are packed into a single multiview frame and sent to the backend over a Gigabit Ethernet connection. Following

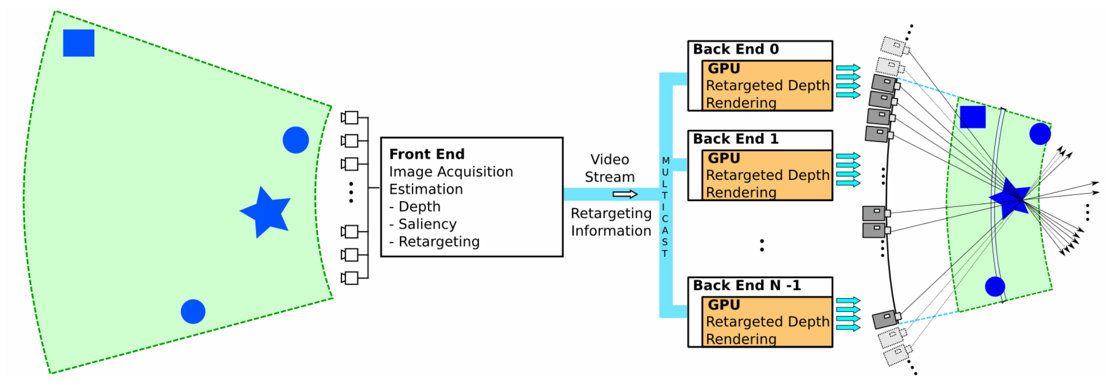


Figure 4.4: End-to-end system overview. The front-end performs capture and adaptively computes retargeting parameters, while the back-end performs all-in-focus rendering

the approach in [9], a reliable UDP multicast protocol is incorporated to distribute the data in parallel to all the clients. Apart from multiview capture and streaming, the master PC also runs in parallel, the rendering application instances for the display central and two lateral optical modules to compute the required light field retargeting function. This function describes the mapping between a set of quantized scene depth plane positions and their corresponding constrained and optimized positions in the retargeted display space. While scene depth plane positions are computed independently by all the clients, the quantized retargeted display plane positions are sent as metadata along with the current multiview frame to the backend.

4.5.2 Back End

The back end constitutes the rendering cluster and the light field display. All the clients in the rendering cluster work independently of each other and produce a set of optical module images. Each client decodes the received multiview images and uploads the RGB channel data as a 3D array to the GPU. For a given display projection module, the depth information for various viewport pixels is computed extending the space sweeping approach of [9] to perform simultaneous estimation and retargeting. The method follows a coarse to fine approach. For each of the camera textures, a Gaussian RGBA pyramid is pre-computed, constructed with a 2D separable convolution of a filter of width 5 and factor of two sub-sampling. In parallel, we also generate and store a descriptor pyramid for pixels of each level which will be used for depth computations. The descriptors are defined following the census representation [43]. Depth values are calculated iteratively by up-scaling the current optical module viewport from coarse to fine resolution with each iteration followed by median and min filtering to remove high frequency noise.
To estimate depth for a given display light ray, space sweeping is performed in display space using a coarse-to-fine stereo-matching method described in [9]. Note that during stereo matching, the matching costs should be evaluated in the camera space where original scene points are located.

DOI:10.15774/PPKE.ITK.2015.005

Thus, while computing the matching cost at a particular depth level, we perspectively transform the candidate point position in display space to camera space using the inverse retargeting function. As mentioned earlier, a light ray may not be emitted by the same display optical module before and after retargeting. Thus, the scene ray corresponding to current display ray can have a different direction, which makes it necessary to re-compute the closest camera set (for matching cost evaluation) at every depth step of space sweeping. For a display-to-scene mapped voxel, ray direction is computed by finite differences, *i.e.*, at a given display depth step, we transform another display voxel position at a consecutive depth step to camera space using the same inverse retargeting function. The required ray direction in scene space at current display depth step is along the ray joining the two transformed points in camera space. Using this direction, we compute a set of four closest cameras over which the matching cost is evaluated and summed. The display depth step with best matching cost will be chosen as input candidate for the next iteration in the coarse-to-fine stereo matching method. As described in [9], matching cost is a function of two factors: the luminance difference that helps in tracking local depth variations and hamming distance between the census descriptors which helps in tracking texture areas and depth boundaries. After pre-defined number of iterations, we will have the depth map computed at finest resolution for all light rays of a display optical module. We then use this computed depth information to calculate the color to be emitted from individual view port pixels of a given display optical module. Specifically, for a display light ray under consideration, we calculate a position in display space and along the display ray that falls at the computed depth level and transform this position to camera space using the inverse retargeting function. The final color for the display light ray is weighted average of the colors sampled at the transformed position from the four nearest cameras.

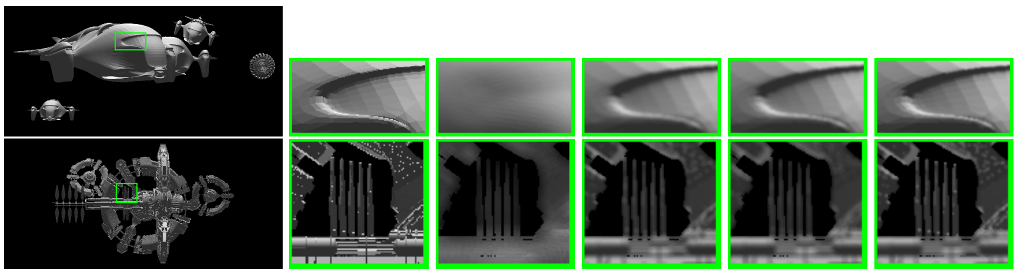


Figure 4.5: Original and retargeted simulation results. Top row: `Sungliders` scene. Bottom row: `Zenith` scene. Left to right: ground truth central view and close-ups: ground truth, without retargeting, with linear, logarithmic and adaptive retargeting. Note that, as we present the content from display center viewing position, viewport content is not distorted in X-Y.

4.6 Results

The end-to-end capture and display pipeline is implemented in Linux. On-the-fly light field retargeting and rendering is implemented on GPU using CUDA. The results of the proposed

DOI:10.15774/PPKE.ITK.2015.005

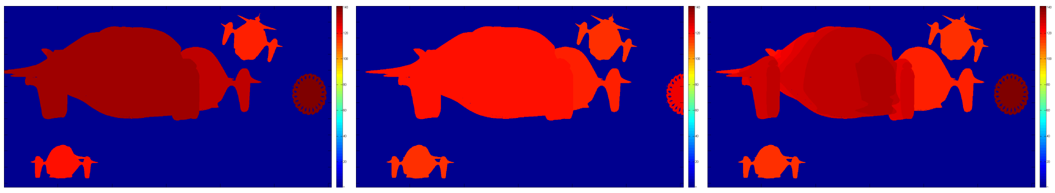


Figure 4.6: Simulated retargeted display side view depth maps of the sequence - `Sungliders`. Left: linear retargeting, middle: logarithmic retargeting and right: adaptive retargeting. The depth variations are better preserved for adaptive retargeting, thus producing increased parallax effect on light field display.

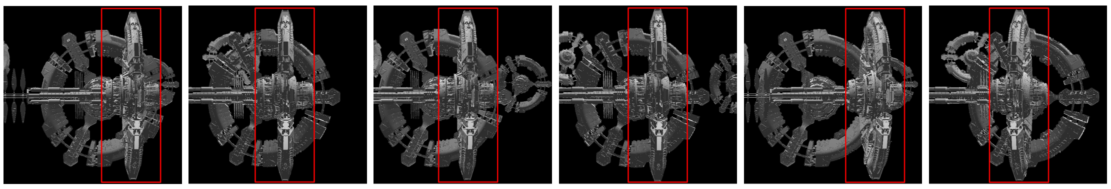


Figure 4.7: Simulated retargeted display side views of the sequence - `Zenith`. Left to right - left view from linear retargeting, right view from linear retargeting, left view from logarithmic retargeting, right view from logarithmic retargeting, left view from adaptive retargeting and right view from adaptive retargeting. Observe better parallax effect of adaptive retargeting due to improved depth preservation of 3D objects.

content aware retargeting are teseted on a Holografika 72in light field display that supports 50° horizontal Field Of View (FOV) with an angular resolution of 0.8°. The aspect ratio of the display is 16:9 with single view 2D-equivalent resolution of 1066×600 pixels. The display has 72 SVGA 800x600 LED projection modules which are pre-calibrated using an automatic multiprojector calibration procedure [14]. The front end is an Intel Core-i7 PC with an Nvidia GTX680 4GB, which captures multiview images at 15 fps in VGA resolution using 18 calibrated Logitech Portable Web cameras. The camera rig covers a base-line of about 1.5m and is sufficient to cover the FOV of light field display. In the back end, we have 18 AMD Dual Core Athlon 64 X2 5000+ PCs running Linux and each equipped with two Nvidia GTX560 1 GB graphics boards. Each node renders images for four optical modules. Front-end and back-end communicate over a Gigabit Ethernet connection. In the following sub-sections, retargeting results using synthetic and real world light field content are presented.

4.6.1 Retargeting Synthetic Light Field Content

Synthetic scenes are employed to evaluate the results and compare them with alternative approaches. As the aim is to retarget the light field content in real-time, objective quality evaluation of the proposed method is limited with ground truth and other real-time methods (in particular, linear and logarithmic remapping [10]). The two synthetic scenes are `Sungliders` and `Zenith`. The ground truth central view and close-ups from the central views generated without

DOI:10.15774/PPKE.ITK.2015.005

Table 4.1: Central view SSIM and RMSE values obtained by comparison with ground truth image for `Sungliders` (S) and `Zenith` (Z) data sets. SSIM=1 means no difference to the original, RMSE=0 means no difference to the original

	Without	Linear	Logarithmic	Adaptive
SSIM-S	0.9362	0.9733	0.9739	0.9778
SSIM-Z	0.8920	0.9245	0.9186	0.9290
RMSE-S	3.6118	2.0814	2.0964	1.9723
RMSE-Z	3.6700	2.8132	2.8910	2.7882

retargeting, with linear, logarithmic and content adaptive retageting are shown in Figure 4.5. The original depths of the scenes are 10.2m and 7.7m, that is remapped to a depth of 1m to match the depth range of the display. Similarly to Masia et al. [38], images are generated by simulating the display behavior, as given by equation 4.3 (also discussed in the beginning of Chapter 2) and the display parameters.

$$s(z) = s_0 + 2\|z\| \tan(\frac{\Phi}{2}) \tag{4.3}$$

Figure 4.5 shows the simulation results: ground truth central view and close-ups from the central views are generated without retargeting, with linear, logarithmic and content adaptive retageting respectively. To generate the results for logarithmic retargeting, a function is used of the form $y = a + b * log(c + x)$, where y and x are the output and input depths. The parameters a, b & c are chosen to map the near and far clipping planes of the scene to the comfortable viewing limits of the display. When the original scene is presented on the display, voxels that are very close to the user appear more blurry. Note that in all the three retargeting methods, after retargeting, the rendered scene is less blurry. The adaptive approach better preserves the object dephts, avoiding to flatten them. This is more evident for frontal objects between the screen and display near plane, which are almost flattened by the linear and logarithmic approaches and the blurry effect is still perceivable. We can see it from insets of Figure 4.5, where near objects drawn with linear and logarithmic retargeting are less sharper than corresponding adaptive retargeted objects. Table 4.1 shows *Structural SIMilariy* (SSIM) index and *Root Mean Square Error* (RMSE) values of various renderings from the two experimental sequences when compared to ground truth. The metrics show that the content adaptive retargeting performs better than linear, logarithmic and no retargeting. The flattening of objects in case of linear and logarithmic retargeting is clearly perceivable as we move away from central viewing position. Figure 4.6 presents the color coded side view depth maps from the scene `Sungliders` for the three test cases. The global compression in linear retargeting results in the loss of depth resolution in the retargeted space. The non-linear logarithmic mapping leads large depth errors unless the objects are located very close to the display near plane. Adaptive retargeting approach produces continuous and better depth variations and thus preserves the 3D shape of objects. The flattening of objects manifests in the form of reduced motion parallax as shown in Figure 4.7.

DOI:10.15774/PPKE.ITK.2015.005

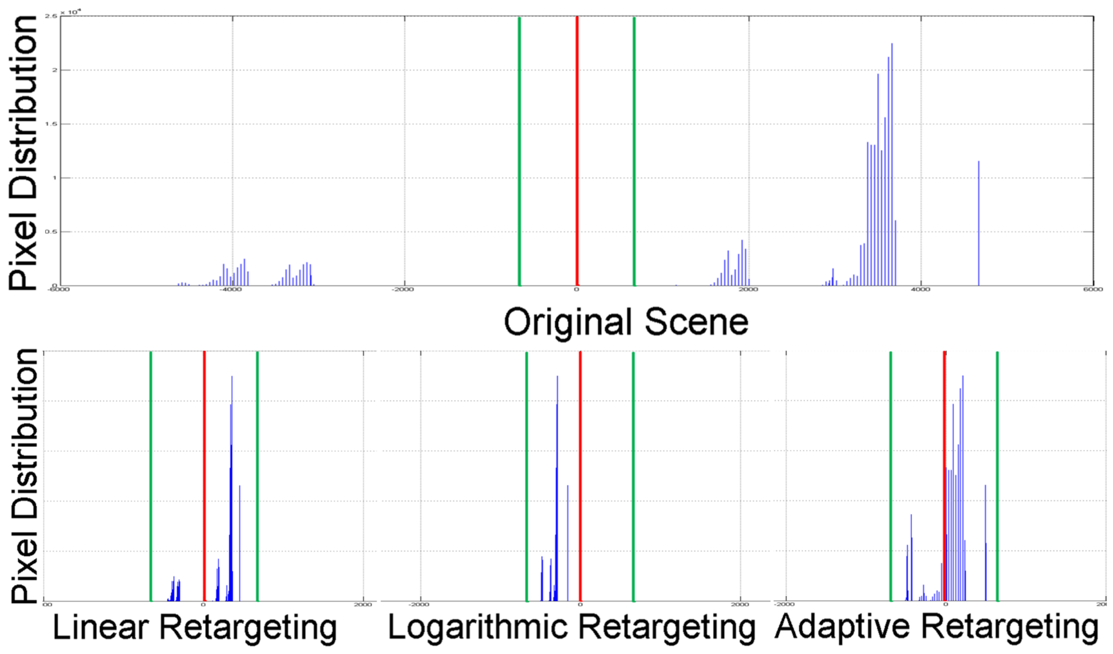


Figure 4.8: Sunglider: linear, logarithmic and adaptive retargeting behavior explained using depth histograms. Top row: Original scene, bottom row : left to right - retargeted scene using linear, logarithmic and adaptive retargeting.

The performance of the proposed method can be better explained from the original and retargeted depth histograms. In Figure 4.8, red lines represent the screen plane and the two green lines before and after a red line correspond to negative and positive comfortably displayable depth limits of the light field display. Linear retargeting compresses the depth space occupied by scene objects and the empty spaces in the same way, logarithmic retargeting is highly dependent on object positions and results in large depth errors after retargeting. In contrast, content aware approach best preserves the depth space occupied by objects and instead, compresses the less significant regions, thus maintains the 3D appearance of objects in the scene.

4.6.2 Retargeting Live Multiview Feeds

To demonstrate the results of the proposed method on real-world scenes, using a simple hand-held camera, the processes of live multiview capturing and real-time retargeted rendering are recorded. It should be noted that the 3D impression of the results on the light field display can not be fully captured by a physical camera. In Figure 4.9, screen shots of the light field display are presented with various renderings at a single time instance of a multiview footage. For fair comparison, images are captured from the same point of view to show the perceivable differences between plain rendering, linear retargeting and adaptive retargeting. Experiments show that the results from the real-world scenes conform with the simulation results on the synthetic scenes. By following direct all-in-focus light field rendering, areas of the scene outside the displayable

DOI:10.15774/PPKE.ITK.2015.005

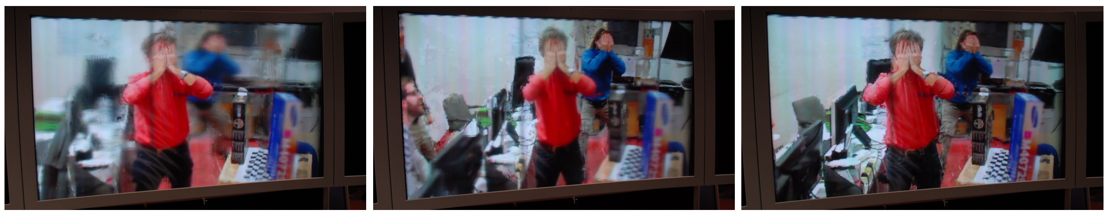


Figure 4.9: Real-time light-field capture and retargeting results. From left to right: without retargeting, with linear retargeting, with adaptive retargeting.

range are subjected to blurring. Linear retargeting achieves sharp light field rendering at the cost of flattened scene. Content aware depth retargeting is capable of achieving sharp light field rendering and also preserves the 3D appearance of the objects at the same time. The front end frame rate is limited at of 15fps by the camera acquisition speed. The back end hardware used in the current work supports an average frame rate of 11fps. However, experiments showed that Nvidia GTX680 GPU is able to support 40fps. In the back end application the GPU workload is subdivided in this way: 30% to upsample depth values, 20% for census computation, 15% jpeg decoding, 13% to extract color from the depth map, other minor kernels occupy the remaining time. Retargeting is embedded in the upsampling and color extraction procedures.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 5

Light Field Interaction

Light field displays create immersive interactive environments with increased depth perception, and can accommodate multiple users. Their unique properties allow accurate visualization of 3D models without posing any constraints on the user's position. The usability of the system will be increased with further investigation into the optimal means to interact with the light field models. In the current work, I designed and implemented two interaction setups for 3D model manipulation on a light field display using Leap Motion Controller. The gesture-based object interaction enables manipulation of 3D objects with 7DOFs by leveraging natural and familiar gestures. To the best of my knowledge, this is the first work involving Leap Motion based interaction with projection-based light field displays. Microsoft Kinect can be used to track user hands. Kinect works fine starting from a given distance from the sensor and is mainly used to detect big gestures. Although it is possible to detect the hand gestures by precisely positioning the sensor and carefully considering the acquired information, tracking minute hand movements accurately is quite imprecise and error prone

5.1 Interaction Devices - Related Works

The devices that enable the interaction with 3D content are generally categorized into two groups which correspond to wearable and hands-free input devices. The devices from the first group need to be physically worn or held in hands, while, on the other hand, no physical contact between the equipment and the user is needed when using hands-free devices.

5.1.1 Wearable Devices

One of the recent commercially successful representatives of the wearable devices was the Nintendo WiiMote controller serving as the input device to the Wii console, released in 2006. The device enables multimodal interaction through vocal and haptic channels but it also enables

DOI:10.15774/PPKE.ITK.2015.005

gesture tracking. The device was used for the 3D interaction in many cases, especially, to track the orientation of individual body parts. On the other hand it is less appropriate for precise object manipulation due to its lower accuracy and relatively large physical dimensions.

5.1.2 Marker-Based Optical Tracking Systems

As wearable devices in general (including data gloves) impede the use of hands when performing real world activities, hand movement may also be tracked visually using special markers attached to the tracked body parts. Optical tracking systems, for example, operate by emitting infra-red (IR) light to the calibrated space. The IR light is then reflected from highly-reflective markers back to the cameras. The captured images are used to compute the locations of the individual markers in order to determine the position and orientation of tracked body parts. The advantage of the approach is a relatively large interaction volume covered by the system; while the disadvantage is represented by the fact that the user still has to wear markers in order to be tracked. The optical tracking system was, for example, used as the tracking device when touching the objects rendered with a stereoscopic display [44]. The results of the study demonstrated the 2D touch technique as more efficient when touching objects close to the display, whereas for targets further away from the display, 3D selection was more efficient. Another study on the interaction with a 3D display is presented in [45]. The authors used optical tracking system to track the positions of markers placed on the user's fingers for a direct gestural interaction with the virtual objects, displayed through a hemispherical 3D display.

5.1.3 Hands-Free Tracking

Optical tracking can also be used for marker-less hands-free tracking. In this case, the light is reflected back from the body surface and the users do not need to wear markers. However, as body surface reflects less light compared to highly-reflective markers this usually results in a much smaller interaction volume. Although a number of studies with hands-free tracking for 3D interaction have been performed with various input setups (e.g. [46], [47]) Microsoft Kinect sensor represents an important milestone in commercially accessible hands-free tracking devices. The device was introduced in late 2012 as an add-on for the Xbox 360 console. Beside visual and auditory inputs, the Kinect includes a depth-sensing camera which can be used to acquire and recognize body gestures for multiple users simultaneously [48]. The device proved to be mostly appropriate for tracking whole body parts (i.e. skeletal tracking), e.g. arms and legs while it is less appropriate for finger and hand tracking. A variety of studies using Kinect and other camera-based approaches has been conducted including studies on the interaction with a 3D display (e.g. [49], [50] [51]). A study similar to the one presented in this paper was conducted by Chan et. al [52] where users had to perform selecting tasks by touching the images rendered on an intangible display. The touch detection was implemented using stereo vision technique with two IR cameras. The display used in the study was based on projection of a flat LCD screen to a

DOI:10.15774/PPKE.ITK.2015.005

fixed virtual plane in front of the user. Consequentially, only 2D planar images were displayed resulting in limited range of viewing angles and no true depth perception as it is provided by volumetric displays.

5.1.4 Leap Motion Controller

Leap Motion Controller is a motion sensing device introduced by Leap Motion Inc. which fits very well with the interaction needs. The device is proven to be relatively inexpensive, precise and is useful in tracking the hands and finger inputs than any existing free hands interaction devices. The device has a high frame rate and comes with a USB interface. The device provides a virtual interaction space of about one sq. meter with almost 1/100th millimeter accuracy. The Leap Motion Control SDK provides access to abstract data such as number of hands and fingers sensed, their location, stabilized palm position etc. A direct access to few set of gestures such as circle, swipe, key tap and screen tap is also provided. Using the SDK [13], it is more convenient to design and program any user defined gestures.

The Leap Motion Controller can be categorized into optical tracking systems based on stereo vision. The device uses three LED emitters to illuminate the surrounding space with IR light which is reflected back from the nearby objects and captured by two IR cameras. The device's software analyzes the captured images in real-time, determines the positions of objects and performs the recognition of user's hands and fingers. The discrete positions of recognized hands, fingers and other objects as well as detected gestures can then be obtained through APIs (Application Programming Interfaces). The device and the coordinate system used to describe positions in the device's sensory space are shown in Figure 5.1.

A study on the Controller's performance [53] revealed that the device's FOV is an inverted pyramid centered on the device. The effective range of the Controller extends from approximately 3 to 30 centimeters above the device (y axis), approximately 0 to 20 cm behind the device (negative z axis) and 20 cm in each direction along the device (x axis). Standard deviation of the measured position of a static object was shown to be less than 0.5 mm.

5.2 Light Field Interaction - Hardware Setup

For the interaction experiment, I used a small-scale light field display of the size comparable to the FOV volume of Leap Motion Controller. The light field display hardware used for this work was developed by Holografika. As mentioned before, we assume that the screen is located at z = 0 with center as the origin of the display coordinate system. y-axis is in the vertical direction, the x-axis pointing to the right and the z-axis pointing out of the screen. The display coordinate system is shown in Figure 5.2 and experimental setup is shown in Figure 5.3. Leap Motion Controller is placed in front of the display with x, y and z-axis parallel to display x, y and z-axis.

All the rendering and interaction functionality is implemented on single computer. The pattern for

DOI:10.15774/PPKE.ITK.2015.005

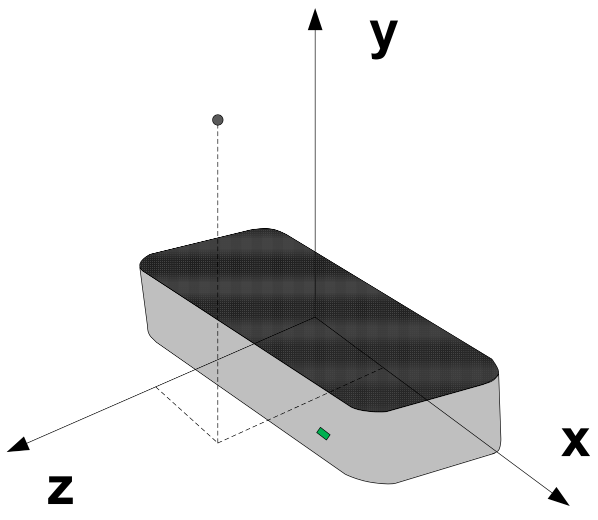


Figure 5.1: Leap Motion Controller and the coordinate system used to describe positions in its sensory space.

interaction is implemented in C++ using OpenGL. The application generates random patterns of tiles in run time and rendered at a given depth. In parallel, the application receives interaction data from Leap Motion Controller, processes and updates the renderer in real-time. The controlling PC runs GL wrapper and feeds the resulting visual data to optical modules. Hence we can see the same application running on a LCD monitor in 2D and on light field display in 3D.

5.3 Interaction with Light Field Displays Using Leap Motion Controller-Implementation

I designed and implemented HoloLeap - a system for interacting with *light field displays* (LFD) using hand gestures tracked by Leap Motion Controller. It is important to note that, although this section describes the implementation of gesture suit using the Leap SDK, the most important novelty of the work is based on the hardware used: the realistic and interactive rendering of scenes on light field display, the intuitive and accurate freehand interaction enabled by the Leap Motion Controller and the efficient and accurate coupling of the input and output. Also the choice of Leap Motion Controller for light field interaction is not based on the simplicity/complexity of using the SDK to derive interaction gestures. For setting up the platform for initial testing, I designed a set of gestures for manipulating virtual objects on a light field display. The system supports the basic six degrees of freedom object manipulation tasks: translation and rotation. I also extended the gesture suite to accommodate basic presentation features: scaling and spinning. Having reviewed previous work, I decided to design a tailor-made gesture suite, as the increased depth perception

DOI:10.15774/PPKE.ITK.2015.005

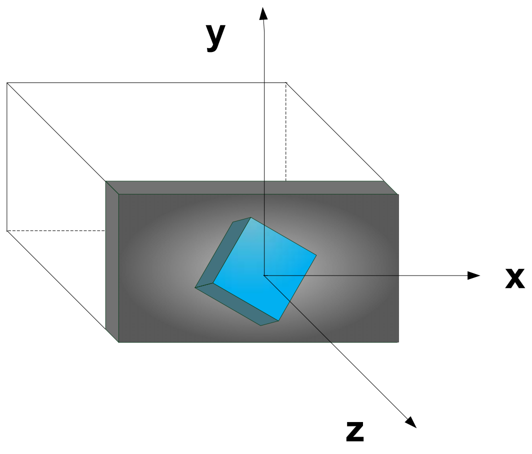


Figure 5.2: Small scale light field display system prototype used for interaction experiment.

of a light field displays affects how objects are manipulated. The various interaction gestures are shown in Figure 5.4 and are briefly explained in the following subsections.

Rotation Around Datum Axes

Simple rotation is performed with a single hand. The user rotates their wrist in the desired direction to rotate the object. This allows for fast correction in all the rotation degrees of freedom, and even combining rotations in a single gesture.

Translation

Translating an object requires the use of both hands. Moving two hands simultaneously without increasing the distance between them translates the object

Presentation features

HoloLeap enables zooming in and out by increasing and decreasing the distance between palms. The method is provided to facilitate investigating details of the model and to easily provide a comprehensive overview.

In order to facilitate presenting 3D objects to audiences, spinning feature is also implemented. The spinning gesture involves closing the palm of one hand and rotating the wrist of the other hand. The speed of the object's spin is directly proportional to the speed of the gesture. The object is set to continuous rotation and the state can be changed by invoking any of the single hand gestures.

DOI:10.15774/PPKE.ITK.2015.005

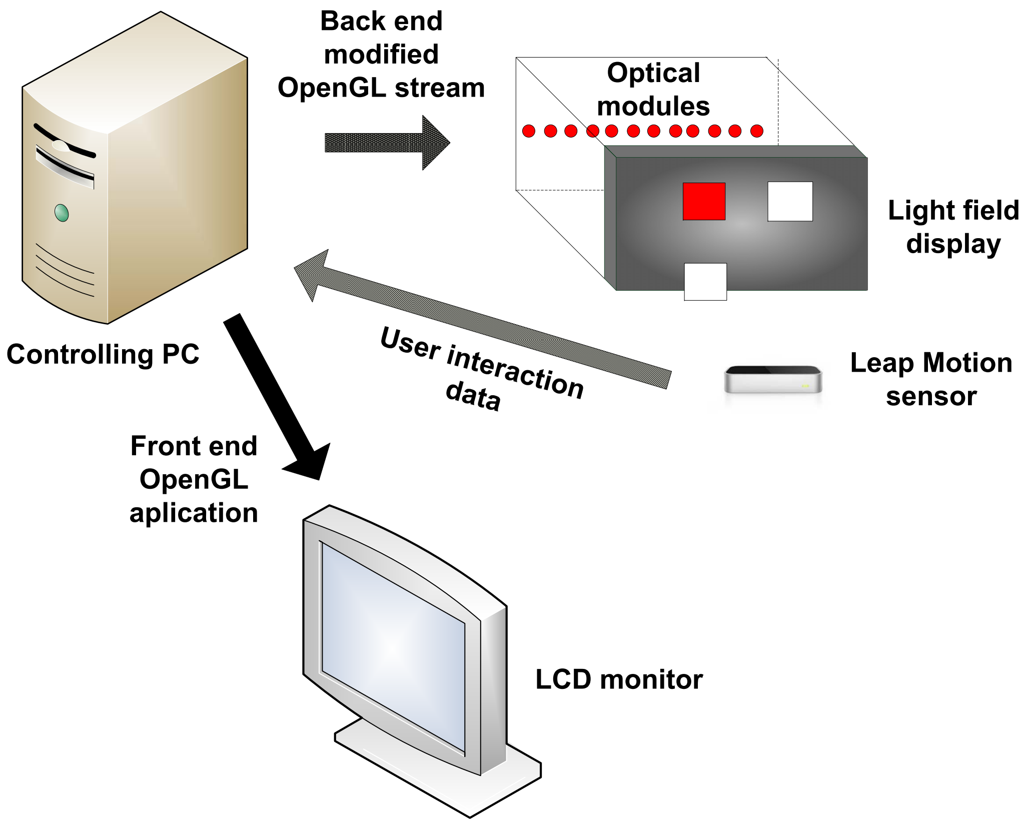


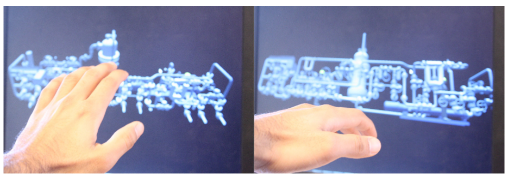
Figure 5.3: Experimental setup: The controlling PC runs two applications: main OpenGL frontend rendering application for 2D LCD display and backend wrapper application that tracks the commands in current instance of OpenGL(front end application) and generates modified stream for light field rendering. The front end rendering application also receives and processes user interaction commands from Leap Motion Controller in real-time.

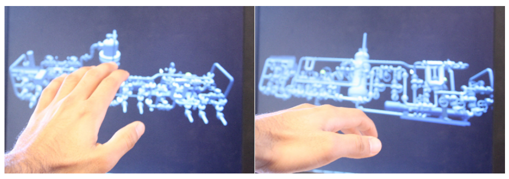
5.3.1 Interactive Visualization on Light Field Display

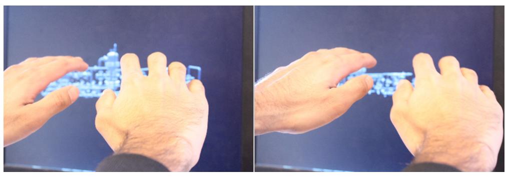
Real-time visualization on light field displays requires rendering the given scene from many viewpoints that correspond to the characteristics of the specific light field display. In case of synthetic scenes, it is easy to derive the required visual information for light field rendering based on the scene geometry. For interaction purpose, as there is an offset between the coordinate space of the Leap Motion Controller and the light field display, to have a realistic and intuitive interaction without causing user distraction, it is important to maintain the scaling ratio between the two coordinate spaces. This ensures a balance between the parameters such as the model volume Vs user hand size, hand motion velocity Vs rendered model motion velocity etc.,. Note that the scene coordinate space may be also different from the display coordinate space and thus it is important to balance the scene-display-leap coordinate systems for providing a user-friendly interaction.

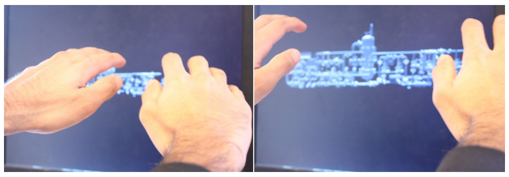
For interactive visualization procedure, we need to know the positions of real-world vertices of

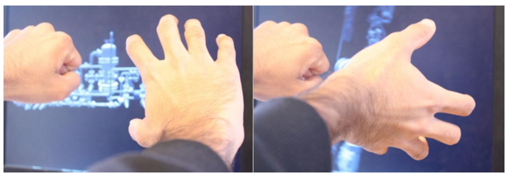
DOI:10.15774/PPKE.ITK.2015.005


(a) Single hand object rotation along x axis


(b) Single hand object rotation along z axis


(c) Double hand object translation


(d) Double hand object zoom in and zoom out


(e) Double hand object spin along z axis

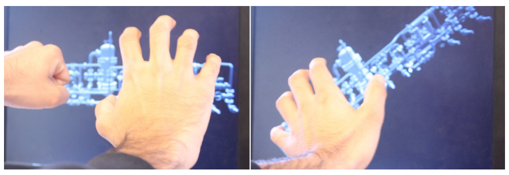

(f) Double hand object spin along x axis

Figure 5.4: Sample gestures for interaction

scene objects in the screen coordinate space. For this purpose, we draw a scene of volume exactly the same as the displayable volume. When mapping the visualization application's Region of Interest (ROI) to the light field display's ROI, we add an additional constraint to map the central plane of the scene to the screen plane. This ensures correct mapping of scene coordinates to display coordinates. This also ensures that the important scene objects are centered at the origin of the display cooridnate system, where we have high 3D resolution. For rendering on light field displays similar to the real-world, perspective projection scheme is followed, i.e., objects close to the user appear larger than the objects far away. However, the Leap Motion Controller supports orthogonal coordinate space while dispensing the interaction data. Thus, we should also make sure that the orthogonal behavior of Leap is carefully adapted to the perspective nature of the display.

Special cases arise when a user hand surpasses the valid region of Leap Motion Controller. This may happen in the occasions such as translating or zooming the model. Whenever they happen, to be more consistent from the rendering aspect, the rendered model is made to bound to the extents box of the display (bounded by the displayable front, back, left, right, top and bottom planes). Soon after if the user hand appears back on the operating range of Leap Motion Controller, the final known position of the interaction model is always retained to avoid any abrupt and annoying movements. This approach provides a new offset value between the coordinate systems and ensures natural interaction as if as user is picking up an object where it was left.

DOI:10.15774/PPKE.ITK.2015.005

5.3.2 Light Field Interaction Prototype For Use Case - Freehand Interaction with Large-Scale 3D Map Data

Exploring and extending the 3D model interaction framework, a practical use case of such a system is also implemented. The use case is an interactive visualization of large-scale 3D map on a light field display. Finding optimal methods of interacting with geographical data is an established research problem within Human-Computer Interaction (HCI). While map datasets are becoming more and more detailed and novel scanning methods enable creating extensive models of urban spaces, enabling effective ways of accessing geographical data is of utmost importance. To the best of my knowledge, this is the first report on freehand interaction with light field display. Light field display is a very interesting and sophisticated piece of modern technology and will play an important role in the future of displaying technologies (of 3D content). This section offers a fundamental explanation of the visualization technology on these displays and explores an interaction scenario using the Leap Motion Controller. The proposed approach cannot be directly (objectively) compared to any of the related works since this is the first study on the interaction with a light field display. However, a slightly different interaction setup and it's evaluation with 2D counterpart is presented in the subsequent section.

3D Map

The software that allows real-time streaming and rendering 3D map data on variety of 2D devices, as well as sample 3D map data have been developed and made available for research by myVR software [54]. The myVR mMap SDK is using a client-server model for downloading map data over the Internet. The C++ interface of the API provides direct access to high level functions such as: querying the depth from the virtual camera at a particular pixel location, getting the position of virtual camera in latitude, longitude and height etc., and also allows real-time streaming and rendering of 3D model of a map. Inside the API, most of the communication is carried out using JSON queries.

The mMap SDK uses composites for rendering the 3D map, with each composite consisting of several layers (*e.g.* we can have a layer that will render vector data, a layer that renders aerial data, and a layer that renders any Point Of Interests (POI's)). A typical map contains several composites and each composite and its corresponding layers are enabled to receive JSON queries and return information to the calling application. Each layer is assigned a priority when created and the order of rendering layers is based on the assigned priority. If two layers have the same priority, the layer that is first created gets the highest priority. The SDK is optimized to eliminate unnecessary redrawing. I have built on a sample map viewer application which allows the exploration of potentially infinite map data, streams the multiresolution map data and displays it using Level Of Detail techniques. This application, which supported only mouse interaction, have been amended with the described interaction techniques and 3D-display specific optimizations.

DOI:10.15774/PPKE.ITK.2015.005

With a 24 Mbps download speed internet connection, the map data is streamed and rendered at 75 frames per second (FPS). The cameras of Leap Motion generate almost 300 FPS of the raw data from which the information on the user hand(s) position is extracted. The frame rate supported by Leap Motion is much higher than the required frame rate for the application leaving us sufficient space to further processing and filtering the gestures before dispatching the final interaction commands. Kinect acquires depth and color stream at only 30FPS and hence limits the interaction speed.

Map Interaction Design

Designing interaction involves defining a set of interface controls to navigate through the map. As the data is visualized on a light field display with a smooth and continuous horizontal parallax, the spatial relation of the objects in the scene e.g., buildings in the map, are properly maintained similar to the real world (see Figure 5.5).

As described before, the streaming and light field visualization is done on the fly without using any pre-rendered animations or images. Thus the interaction process should be fast enough to manipulate heavy light field data. Once the interaction control messages are acquired, rendering is performed in real-time using OpenGL wrapper library. Designing interaction gestures in a way that is obvious for untrained users can be a very complicated task. The main factor of concern is the complexity and familiarity of a gesture, as it directly effects the learning time. On one hand easily detectable gestures such as open hand, closed hand may not be very intuitive and require additional attention from the user. On the other hand more intuitive gestures used to interact with real world objects e.g., lifting/grabbing, could be more complex and often cannot be precisely defined within a given group of users. For us the main challenge is to bring the best trade-off between the complexity of the gesture and its intuitiveness i.e., the gestures should be very easy to learn and also should be precisely detected within a given amount of time to support real-time interaction.

A contributing factor to the degree of intuitiveness is the increasing usage of mobile smart phones. These devices usually contain maps and the touch interaction for navigating through the map is very well in practice nowadays. Designing a set of similar gestures for interaction can make the learning phase shorter, or eliminate it completely, based on prior experience. Generally interaction includes pan, rotate and zoom. The respective gestures are defined in the following sub-sections. All the gestures are active within the valid Field Of View (FOV) of Leap Motion device and are shown in Figure 5.6. Note that as there is no direct relation between the display and Leap Motion Controller's *FOV*, the identical gesture suit can be applicable for interactive visualization of models on any light field display.

A. Pan

Panning in horizontal and vertical directions can be done by translating the virtual camera in the opposite direction by a given amount. Panning is achieved using one hand (either left or

DOI:10.15774/PPKE.ITK.2015.005

right). Leap Motion driver runs a separate thread providing the information about the number of hands and the stabilized palm position continuously. As the display, scene and Leap Motion coordinate systems are different, first the axes are normalized and then the relative changes as seen by the Leap are used to update the scene camera. The Leap device faces upside, the direction vectors of the device and the light field display point towards the same direction. Information on the previous frame from the Leap is always stored until the next frame. Upon receiving a new frame, the difference in the hand position is calculated and a panning message with respective parameters is dispatched to translate the virtual camera.

B. Rotate

The rotation gesture is implemented using one hand. The virtual camera is made to mimic the hand rotation. As translation also depends on single hand gesturing, it is needed to isolate the rotate and pan actions carefully for a smooth interaction. A simple approach is to detect the change in the position of a hand in a given time. If the stabilized palm position does not change above a fixed threshold in a given amount of time, rotation event is recorded and the subsequent steps involve assessing the amount of hand rotation in degrees. Similar to translation we rely on the previous frame information to acquire the rotation amount and direction.

C. Zoom

Zooming mode is activated if two hands are detected within the FOV of Leap Motion. Bringing two hands closer is the gesture for zooming out and taking the hands apart results in zooming in to the map as shown in Figure 4. As the entire application is real-time, it is important to preserve the states of various modes (for e.g., translation at a given zoom, zoom at a given rotation and translation). From the rendering side, the view matrix takes care of the virtual camera positioning and it is also important to check the state of a given mode from the interaction point of view as we only need to update the current state. This is done by storing the last known active time of various modes and checking the time lapse between current and previous active states.

Similar to panning and rotation, zooming commands are dispatched considering the current and previous frames from the Leap Motion. Experiments showed that depending on the previous frame provides sufficient and accurate information to detect the change in states and also meets real-time requirements.

5.4 Direct Touch Interaction - Prototype Implementation

In the aforementioned interaction setup, interaction and display spaces remain isolated and such approaches do not explicitly take into account the display characteristics in terms of both geometry and resolution of the reproduced light fields for interaction.

In the section, a framework is presented to explore direct interaction with virtual objects on a light field display. We couple the interaction and display spaces to provide an illusion of touching virtual objects. To the best of my knowledge, this is the first study involving direct interaction with

DOI:10.15774/PPKE.ITK.2015.005

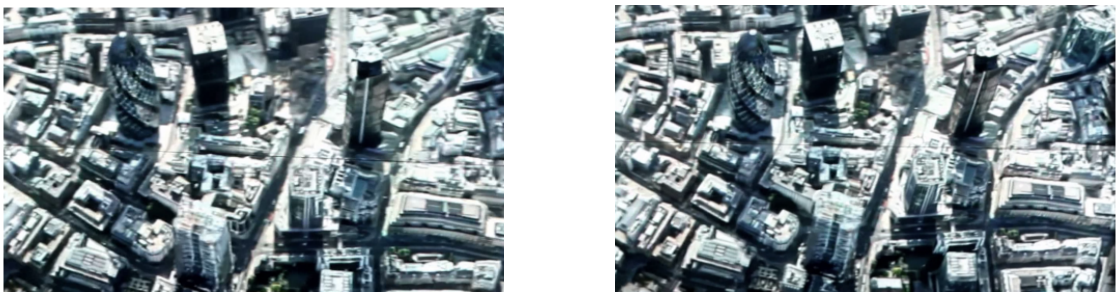


Figure 5.5: Sample 3D map on a light field display. (a) & (b) shows the identical light field as seen from different viewing positions

virtual objects on a light field display using Leap Motion Controller. The proposed interaction setup is very general and is applicable to any 3D display without glasses. In the current work 3D interaction framework is explored for holographic light field display with full horizontal parallax use. However, the method is easily extendable to 3D displays with vertical parallax as well. In addition, the method is scalable, and the interaction space can easily be extended by integrating multiple Leap Motion Controllers.

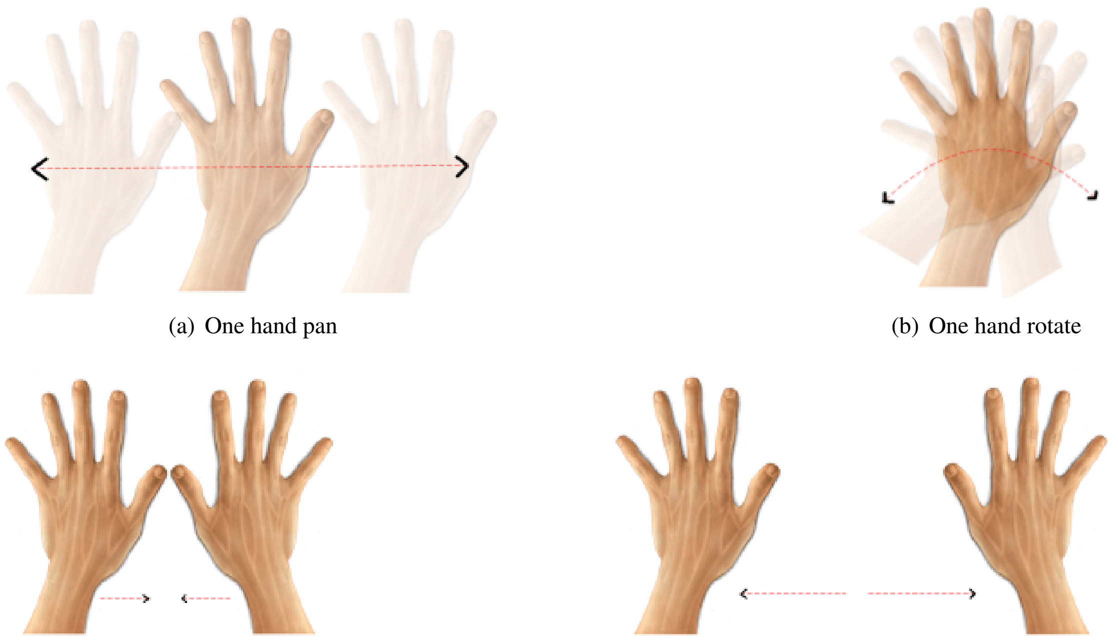
5.4.1 Calibrating Light Field Display to Leap Motion Controller

With the advance of scientific fields such as computer vision, robitics and augmented reality, there are increasingly more and more solutions available for problems related to 3D calibration based on point cloud data. These calibration algorithms are used in several applications such as 3D object scanning, 3D reconstruction and 3D localization etc., The more popular approaches that are still in practice in the state-of-the art methods are *Singular Value Decomposition* (SVD) or *Principal Component Analysis* (PCA) or more complex iterative approaches such as *Iterative Closest Point* (ICP) algorithm. For more details about these algorithms and their evaluation, the readers are referred to [55].

To preserve the motion parallax cue while rendering content on light field displays, perspective rendering is followed. Perspective rendering on light field displays involves assuming a specific observer distance from the screen. As a previous step to rendering, all the light emitting optical modules are calibrated using the observer distance from the screen. After the display calibration is performed, due to the non-linear optical properties of the screen, the resulting coordinate system for rendering is a sort of skewed and is not Cartesian. Thus the regular methods for 3D registration can not be applicable directly for calibrating the light field display and leap motion controller. Further, inside the display coordinate system, the volume of the 3D points are varying according to the spatial resolution which should be also taken into account during the calibration for more realistic interaction.

For calibration, we assume that the display is at a fixed position with Leap Motion Controller

DOI:10.15774/PPKE.ITK.2015.005



(a) One hand pan (b) One hand rotate

(c) Two hands zoom out (d) Two hands zoom in

Figure 5.6: Sample interaction gestures for 3D map interaction

placed anywhere in front of it. The exact position of the Controller in the display coordinates is not known. To ensure uniformity, we assume that both the display and Controller coordinates are in real world millimeters. In practice, when using Leap Motion Controller, hand-positioning data can be more accurately acquired at heights greater than 100 mm. To meet this requirement, we place the Controller at a height less than h_{max}, the maximum allowed height from the display center, where h_{max} is given by the equation 5.1 and D_h is the height of the screen in millimeters.

$$h_{max} = (\frac{D_h}{2}) - 100mm \tag{5.1}$$

As it is not possible physically to reach the zones of the display where depth values (on the z-axis) are negative, we only consider the depth planes on and above the surface of the display for interaction. In the current work, an approach based on sparse markers for performing the calibration is followed. A set of spherical markers centered at various known positions in display coordinates are rendered on the light field display and user has to point to the centers of the projected spheres one after another sequentially with index finger. The positions of the pointed centers as seen by the user (the fingertip positions) are recorded in Leap Motion Controller coordinate system. This information serves as an input for calculating the transfer function between the two coordinate systems.

As mentioned, the apparent size of the calibration marker will not be the same when projected on the surface of the screen and elsewhere. Also, the display geometry calibration data is calculated based on minimizing the projection error on the surface of the screen. Thus, similar to spatial

DOI:10.15774/PPKE.ITK.2015.005

resolution, the calibration accuracy will not be the same all over and is spatially varying. One of the outcomes of reduced calibration accuracy is blurring. Although a blurred background far from the user is acceptable, excessive blur near the user leads to discomfort. Also, Leap Motion Controller has relatively high frame rates and can track minor finger/hand movements. A minute finger shaking during the calibration can substantially reduce the calibration accuracy. Hence, given the size of the display and the precision of the Controller (up to 1/100 of an mm) we should take all the aforementioned effects in to account to obtain accurate calibration data.

To minimize the error resulting from the non-uniform spatial resolution and display geometry calibration accuracy, the total available depth range of the display is limited for this experiment. We define two boundary planes within the total displayable depth range where the apparent spatial resolution and calibration accuracy is almost the same as on the screen (see Figure 5.7). This is done by measuring the size of a single pixel at various depths and comparing it with the size of the pixel on the screen plane (for information on pixel size calculation, please refer to section 2.3). Markers are drawn on the surfaces of the screen plane and on the physically accessible boundary plane and their positions in the display space and Leap Motion Controller space are recorded simultaneously. The calibration should produce a transform Ω between the two coordinate systems that minimizes the sum of Euclidean distances between the original and projected points when the set of all 3D points in one system is transformed to another.

Let the $P_i^{disp} \in \Re^3$ be the position of i^{th} voxel in the display rendering coordinate system (after the holographic transform), let the $P_i^{leap} \in \Re^3$ be the position of i^{th} voxel in the Leap Motion Controller coordinate system and let the $P_i^{projleap} \in \Re^3$ be the position of i^{th} voxel in the Leap Motion Controller space projected into the display space, where i is the index of the current voxel. Then $P_i^{projleap}$ and P_i^{leap} are related as following:

$$P_i^{projleap} = \Omega * P_i^{leap} \tag{5.2}$$

where $\Omega \in \Re^{4\times4}$ is the required transform between two coordinates system. Thus, Ω should minimize

$$\sum_{i=0}^{n-1} \left(\mu_i * Euclideandist\left(P_i^{disp}, P_i^{projleap}\right)\right) \tag{5.3}$$

where n is the number of discrete voxels within the comfortable viewing range outside the display and the constant μ_i is given by the following equation:

$$\mu_i = \begin{cases} 1 & \text{if } i^{th} \text{ display voxel is used for calibration} \\ 0 & \text{if } i^{th} \text{ display voxel is not used for calibration} \end{cases} \tag{5.4}$$

Thus using homogeneous coordinates, any coordinate $((x^{leap}, y^{leap}, z^{leap}))$ in the Leap Motion Controller space can be transformed (based on equation 5.2) to the display spaces coordinates

DOI:10.15774/PPKE.ITK.2015.005

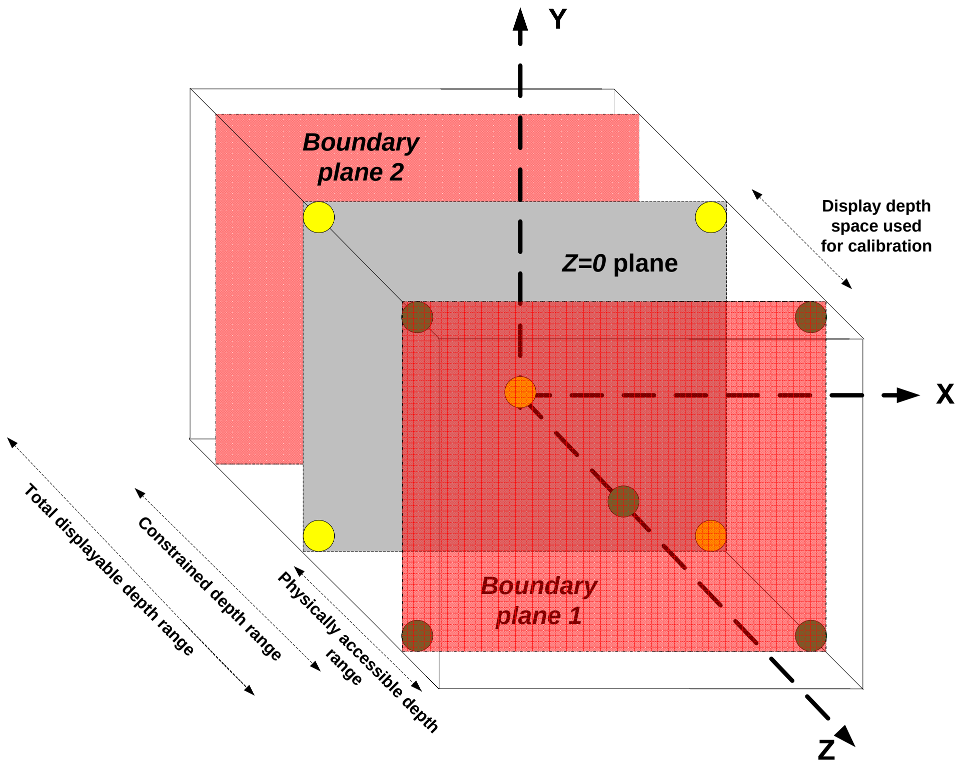

Y
X
Z
Boundary plane 2
Z=0 plane
Boundary plane 1
Display depth space used for calibration
Total displayable depth range
Constrained depth range
Physically accessible depth range

Figure 5.7: Light field display and Leap Motion Controller calibration: Depth volume bounded by the screen plane and physically accessible constrained boundary plane is calibrated to a comparable sized volume of Leap Motion Controller. Yellow circles show the markers drawn on the screen plane and green circles show markers drawn on boundary plane 1 in the figure. When the markers are rendered on the display, the radius of the markers vary depending on the spatial resolution at the center of the marker.

$((x^{disp}, y^{disp}, z^{disp}))$:

$$\begin{pmatrix} x^{disp} \\ y^{disp} \\ z^{disp} \\ 1 \end{pmatrix} = \Omega^{4\times4} \begin{pmatrix} x^{leap} \\ y^{leap} \\ z^{leap} \\ 1 \end{pmatrix} \tag{5.5}$$

Substituting equation 5.2 in 5.3 the required affine transformation matrix should minimize the following energy function:

$$\sum_{i=0}^{n-1} \left(\mu_i * Euclideandist \left(P_i^{disp}, \Omega \times P_i^{leap} \right) \right) \tag{5.6}$$

subject to $P_j^d isp = P_j^l eap, j = 0, 1, 2, 3., m1$

DOI:10.15774/PPKE.ITK.2015.005

where m is the number of markers used for calibration. OpenCV library [56] is used to solve the above optimization problem, which also eliminates any possible outliers in the calibration process. As both the display and the Leap Motion Controller volumes are finite and bounded, it is enough to render markers along the corners of interaction volume bounding box. Although eight corner markers are enough for acquiring a good calibration (total error less than 1 μ m), it is observed that using ten markers improves the accuracy even further. The additional two markers are placed at the centroids of the two z-bounding planes in display space (see Figure 5.7). Increasing the number of markers beyond ten has no considerable effect on calibration accuracy. The calibration process here is further customized here as a display specific optimization problem with constraints governed by the displayable depth and varying spatial resolution. In addition to just restricting the depth space for interaction we also formulate the *sphere of confusion* (SoC) within the interaction boundaries which makes the calibration method more accurate. In our case, the spatial resolution of the display changes with depth, according to the equation 2.1. During interaction, to account for the varying depth resolution within the defined boundary planes, a *Sphere of Confusion* is formulated and the registration of user's finger position is allowed anywhere within the sphere centered at the current 3D position. The radius of SoC is a function of depth from the surface of the screen.

In order to quantify the calibration results accuracy, the Leap Motion Controller space uniformly sampled with 20 samples along each dimension (8000 samples in total). The distance between adjacent samples is 9 mm in the horizontal direction and 5 mm in the vertical direction. These samples are projected individually on to the display space using the calculated transform Ω and record the Euclidean distance between the original and projected point. Figure 5.8 shows the projection errors made at various positions on a uniform grid by a sample calibration process. As shown in the figure, the calibration error is less than 1 μ m in most of the places. This is negligible compared to human finger tremor (the order of magnitude of a millimeter) or even Controller's accuracy.

5.4.2 Direct Touch Interaction System Evaluation

Evaluation Design

The proposed freehand interaction with the light field display was evaluated through a simple within-subject user study with 12 participants. Three tiles of the same size were displayed simultaneously and the participants were asked to point (touch) the surface of the red tile as perceived in space (Figure 5.9). The positions of the tiles varied from trial to trial to cover the entire FOV of the display. 3D and 2D display modes were used representing two different experimental conditions:

- In 2D mode, the displayed objects were distributed on a plane in close proximity of the display surface; and

DOI:10.15774/PPKE.ITK.2015.005

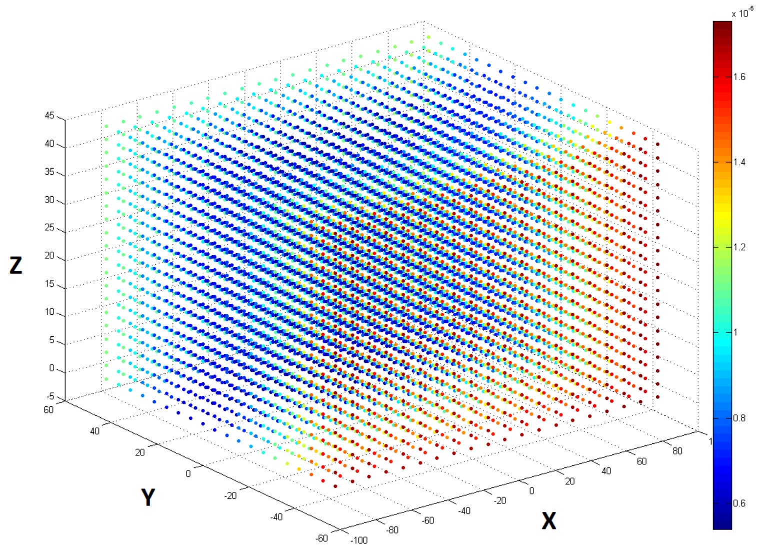


Figure 5.8: Calibration errors on a uniformly sampled grid in Leap Motion Controller space after projecting to display space.

- In 3D mode, the objects were distributed in a space with the distance varying from 0 to 7 cm from the display.

The 2D mode provided a control environment, which was used to evaluate the specifics of this particular interaction design: the performance and properties of the input device, display dimensions, specific interaction scenario (*e.g.*, touching the objects), *etc*. Each participant was asked to perform 11 trials within each of the two conditions. The sequence of the conditions was randomized across the participants to eliminate the learning effect.

Direct touch interaction system evaluation

I have conducted a user study to evaluate the proposed freehand interaction with the light field display through a simple within-subject user study with 12 participants. Three tiles of the same size were displayed simultaneously and the participants were asked to point (touch) the surface of the red tile as perceived in space (see Figure 5.9). The positions of the tiles varied from trial to trial to cover the entire FOV of the display. 3D and 2D display modes were used representing two different experimental conditions:

- In 2D mode, the displayed objects were distributed on a plane in close proximity of the display surface; and

- In 3D mode, the objects were distributed in a space with the distance varying from 0 to 7 cm from the display.

DOI:10.15774/PPKE.ITK.2015.005

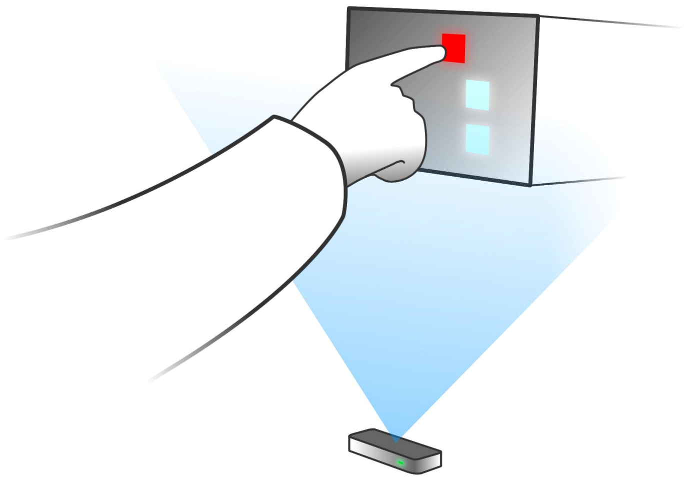


Figure 5.9: Direct touch interaction prototype.

The 2D mode provided a control environment, which was used to evaluate the specifics of this particular interaction design: the performance and properties of the input device, display dimensions, specific interaction scenario (e.g., touching the objects), etc. Each participant was asked to perform 11 trials within each of the two conditions. The sequence of the conditions was randomized across the participants to eliminate the learning effect. The light field display and the interaction design were evaluated from the following aspects:

- Task completion times
- Cognitive workload
- Perceived user experience

The task completion time was measured from the moment when a set of tiles appeared on the display until the moment the user touched the red tile (e.g., hovered over the area where the red tile was displayed within a specific spatial margin of error (15 mm) and for a specific amount of time (0.5 s)). The cognitive workload was measured through the NASA TLX (Task Load Index) questionnaire, which provides a standardized multi-dimensional scale designed to obtain subjective workload estimates [57]. The procedure derives an overall workload score on the basis of a weighted average of ratings on the following six subscales: "Mental Demands", "Physical Demands", "Temporal Demands", "Own Performance", "Effort" and "Frustration". The perceived user experience was measured through UEQ (User Experience Questionnaire) [58]. which is intended to be a user-driven assessment of software quality and usability. It consists of 26 bipolar items, each to be rated on a seven-point Likert scale (1 to 7). The UEQ algorithm derives

DOI:10.15774/PPKE.ITK.2015.005

a quantified experience rated using the six subscales labeled "Attractiveness", "Perspicuity", "Efficiency", "Dependability", "Stimulation" and "Novelty" of the technology evaluated.

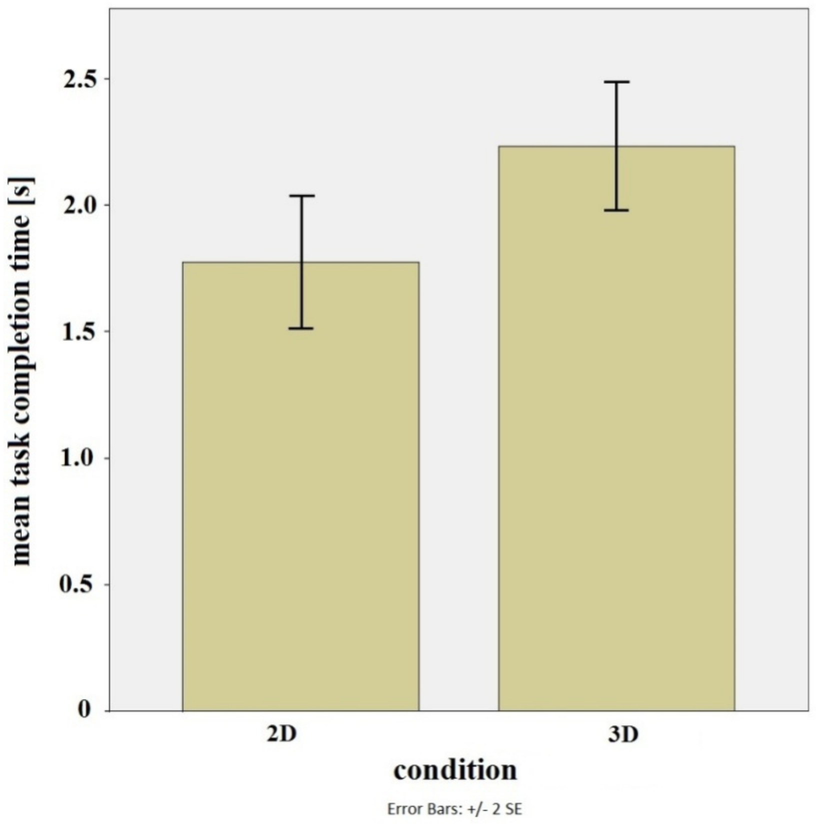
Results

Figure 5.10 (a) shows mean task completion times for both conditions. The results of the T-test showed the interaction in 3D to be significantly slower than the interaction in 2D (t(22) = 2.521, p = 0.019). This result was expected since the additional dimension implies extra time that is needed to, firstly, cognitively process the visual information and, secondly, to physically locate the object in space. Figure 5.10 (b) shows mean workload scores for the subscales of the NASA TLX test as well as the overall workload score. The results of the T-test (t(22) = -0.452, p = 0.655) reveal no significant difference in cognitive workload between the conditions.

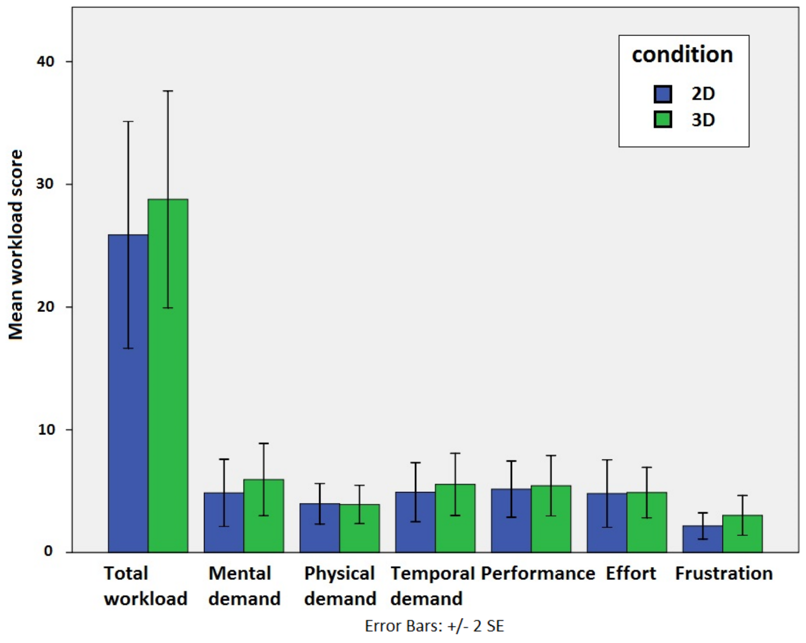
Similarly, the results of the UEQ also did not reveal any significant differences between both conditions in overall user experience score as well as in the majority of the individual subscales of the test. In other words, results show that users did not perceive any significant differences between the conditions in terms of general impression, the easiness to learn how to interact with the content, the efficiency of such interaction, the reliability or the predictability of the interfaces used and the excitement or the motivation for such an interaction. The exception is the novelty subscale where a tendency towards higher preferences for the 3D mode can be observed.

The analysis of the post-study questionnaire revealed that the rendered objects were seen clearly in both experimental conditions. However, the users favored the 3D mode in terms of rendering realism. When asked to choose the easiest mode, the user’s choices were equally distributed between both modes. However, when asked which mode led to more mistakes in locating the exact object position, two-thirds indicated the 3D mode, which is reflected also in longer task completion times in this particular mode. Finally, when asked about their preference, two-thirds of the participants chose the 3D mode as their favorite one.

DOI:10.15774/PPKE.ITK.2015.005



(a)



(b)

Figure 5.10: (a) Mean task completion times for the interaction with the objects in 2D and 3D.& (b)Total workload score and workload scores on the individual subscales of the NASA TLX (Task Load Index) test.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 6

Summary of New Scientific Results

The results of this dissertation can be categorized into three main parts:
results dealing with light field

- Representation
- Retargeted rendering
- Interaction

The respective contributions are briefed in the following thesis groups.

6.1 Thesis Group I - Light Field Representation

Examining the light field conversion process from camera images acquired from several closely spaced cameras, I proposed a fast and efficient data reduction approach for light field transmission in multi-camera light field display telepresence environment.
Relevant publications: [C2] [C3] [C4] [C5] [O1] [J3] [O2] [O3]

6.1.1 Fast and Efficient Data Reduction Approach for Multi-Camera Light Field Display Telepresence System

I proposed an automatic approach that isolates the required areas of the incoming multiview images, which contribute to the light field reconstruction. Considering a real-time light field telepresence scenario, I showed that up to 80% of the bandwidth can be saved during transmission.

- Taking into account a light field display model and the geometry of captured and reconstructed light field, I devised a precise and automatic data picking procedure from multiview camera images for light field reconstruction.

DOI:10.15774/PPKE.ITK.2015.005

- The proposed method does not rely on image/video coding schemes, but rather uses the display projection geometry to exploit and eliminate redundancy.
- Minor changes in the capturing, processing and rendering pipeline have been proposed with an additional processing at the local transmission site that helps in achieving significant data reduction. Furthermore, the additional processing step needs to be done only once before the actual transmission.

6.1.2 Towards Universal Light Field Format

Exploring the direct data reduction possibilities (without encoding and decoding), I presented the preliminary requirements for a universal light field format.

- Simulations were made to see how the *field of view* (FOV) of the receiver's light field display affects the way the available captured views are used. Seven hypothetical light field displays were modeled, with the FOV ranging between 27^0 and 89^0. Source data with 180 cameras, in a 180^0 arc setup, with 1 degree angular resolution has been used.
- Analyzing the pixel usage patterns during the light field conversion, the affect of display's FOV on the number of views required for synthesizing the whole light field image has been realized.
- This analysis has shown that depending on the FOV of the display, the light field conversion requires 42 to 54 views as input for these sample displays. Note the actual number depends on the source camera layout (number and FOV of cameras), but the trend is clearly studied.
- Based on the use cases and processing considerations, three aspects were formulated that need attention and future research when developing compression methods for light fields:
  - The possibility to encode views having different resolution must be added.
  - The ability to decode the required number of views should be supported by the ability to decode views partially, starting from the center of the view, thus decreasing the computing workload by restricting the areas of interest.
  - Third, efficient coding tools for nonlinear (curved) camera setups shall be developed, as it is expected to see this kind of acquisition format more in the future.

6.2 Thesis Group II - Retargeted Light Field Rendering

I presented a prototype of an efficient on-the-fly content aware real-time depth retargeting algorithm for accommodating the captured scene within acceptable depth limits of a display. The discrete nature of light field displays results in aliasing when rendering scene

DOI:10.15774/PPKE.ITK.2015.005

points at depths outside the supported depth of field causing visual discomfort. The existing light field rendering techniques: plain and rendering through geometry estimation, need further adaption to the display characteristics, for increasing quality of visual perception. The prototype addresses the problem of light field depth retargeting. The proposed algorithm is embedded in an end-to-end real-time system capable of capturing and reconstructing light field from multiple calibrated cameras on a full horizontal parallax light field display.
Relevant publications: [C6] [C7] [C8] [J4] [J2] [C9]

6.2.1 Perspective Light Field Depth Retargeting

I proposed and implemented a perspective depth contraction method for live light field video stream that preserves the 3D appearance of salient regions of a scene. The deformation is globally monotonic in depth, and avoids depth inversion problems.

- All-in-focus rendering technique with 18 cameras in the capturing side is considered for implementing the retargeting algorithm and a non-linear transform from scene to display that minimizes the compression of salient regions of a scene is computed.

- To extract the scene saliency, depth and color saliency from perspectives of central and two lateral display projection modules is computed and combined. Depth saliency is estimated using a histogram of the pre-computed depth map and to estimate color saliency, a gradient map of the color image associated to the depth map of the current view is computed and dilated to fill holes. The gradient norm of a pixel represents color saliency.

- To avoid any abrupt depth changes scene depth range is quantized into different depth clusters and the depth and color saliency inside each cluster is accumulated.

- While adapting the scene to display, displacement of depth planes parallel to XY = 0 plane results in XY cropping of the scene background. Thus, in order to preserve the scene structure, perspective retargeting approach is followed i.e., along with z, XY positions are also updated proportional to $\frac{1}{\delta Z}$, as done in a perspective projection. Thus in the retargeted space, the physical size of the background objects is less than the actual size. However, a user looking from the central viewing position perceives no change in the apparent size of the objects as the scene points are adjusted in the direction of viewing rays.

6.2.2 Real-time Adaptive Content Retargeting for Live MultiView Capture and Light Field Display

I presented a real-time plane sweeping algorithm which concurrently estimates and retargets scene depth. The retargeting module is embedded into an end-to-end system capable of real-time

DOI:10.15774/PPKE.ITK.2015.005

capturing and displaying with full horizontal parallax high-quality 3D video contents on a cluster-driven multiprojector light field display with full horizontal parallax.

- While this use is straightforward in a 3D graphics setting, where retargeting can be implemented by direct geometric deformation of the rendered models, the first setup for real-time multiview capture and light field display rendering system incorporating the adaptive depth retargeting method has been proposed and implemented.
- The system seeks to obtain a video stream as a sequence of multiview images and render an all-in-focus retargeted light field in real-time on a full horizontal light field display. The input multiview video data is acquired from a calibrated camera rig made of several identical off the shelf USB cameras.
- The captured multiview data is sent to a cluster of computers which drive the display optical modules. Using the display geometry and input camera calibration data, each node estimates depth and color for corresponding light rays. To maintain the real-time performance, the depth estimation and retargeting steps are coupled.

6.2.3 Adaptive Light Field Depth Retargeting Performance Evaluation

I evaluated the objective quality of the proposed depth retargeing method by comparing it with other real-time models: linear and logarithmic retargeting and presented the analysis for both synthetic and real-world scenes.

- To demonstrate results on synthetic scenes, two sample scenes are considered: Sungliders and Zenith. The ground truth central view and close-ups from the central views generated without retargeting, with linear, logarithmic and content adaptive retageting are shown in Figure 4.5. The original depths of the scenes are 10.2m and 7.7m, that are remapped to a depth of 1m to match the depth range of the display. Retargeted images are generated by simulating the display behavior.
- Figure 4.5 shows the simulation results: ground truth central view and close-ups from the central views generated without retargeting, with linear, logarithmic and content adaptive retageting. To generate the results for logarithmic retargeting, a function of the form $y = a + b * log(c + x)$ is used, where y and x are the output and input depths. The parameters a, b & c are chosen to map the near and far clipping planes of the scene to the comfortable viewing limits of the display.
- Objective evaluation is carried out using two visual metrics: *SSIM and PSNR*. Results show that the adaptive approach performs better and preserves the object dephts, avoiding to flatten them.

DOI:10.15774/PPKE.ITK.2015.005

- To demonstrate the results on real-world scenes, using a simple hand-held camera, the processes of live multiview capturing and real-time retargeted rendering is recorded. It should be noted that the 3D impression of the results on the light field display can not be fully captured by a physical camera.
- In Figure 4.9, the screen shots of the light field display with various renderings at a single time instance of a multiview footage are presented. For fair comparison, images are captured from the same point of view to show the perceivable differences between plain rendering, linear retargeting and adaptive retargeting.
- Experiments show that the results from the real-world scenes conform with the simulation results on the synthetic scenes. By following direct all-in-focus light field rendering, areas of the scene outside the displayable range are subjected to blurring. Linear retargeting achieves sharp light field rendering at the cost of flattened scene. Content aware depth retargeting is capable of achieving sharp light field rendering and also preserves the 3D appearance of the objects at the same time.
- The front end frame rate is limited at of $15fps$ by the camera acquisition speed. The back end hardware used in the current work supports an average frame rate of $11fps$. However, experiments showed that Nvidia GTX680 GPU is able to support $40fps$.
- In the back end application the GPU workload is subdivided in this way: 30% to upsample depth values, 20% for census computation, 15% jpeg decoding, 13% to extract color from the depth map, other minor kernels occupy the remaining time. Retargeting is embedded in the upsampling and color extraction procedures.

6.3 Thesis Group III - Light Field Interaction

I designed and implemented two interaction setups for 3D model manipulation on a light field display. The gesture-based object interaction enables manipulation of 3D objects with 7DOFs by leveraging natural and familiar gestures.
Relevant publications: [C10] [C11] [J1] [C1]

6.3.1 HoloLeap: Towards Efficient 3D Object Manipulation on Light Field Displays

I designed and implemented HoloLeap - a system for interacting with light field displays (LFD) using hand gestures. For tracking user hands, leap motion controller (LMC) is used which is a motion sensing device introduced by Leap Motion Inc. and fits very well with the interaction needs. The device is proven to be relatively inexpensive, precise and is useful in tracking the hands and finger inputs than any existing free hands interaction devices. The device has a high

DOI:10.15774/PPKE.ITK.2015.005

frame rate and comes with a USB interface. The device provides a virtual interaction space of about one sq. meter with almost $1/100^{th}$ millimeter accuracy.

- Manipulation gestures for translation, rotation, and scaling have been implemented. Continuous rotation ("spinning") is also included. I designed a custom gesture set, as the increased depth perception of a light field display may affect object manipulation.
- The goal was to enhance the ad-hoc qualities of mid-air gestures. Compared to handheld devices (e.g., a mouse) gestural interaction allows one to simply walk up and immediately begin manipulating 3D objects. Rotation uses a single hand. The user rotates their wrist in the desired direction to rotate the object. It allows for fast.
- Rotation uses a single hand. The user rotates their wrist in the desired direction to rotate the object. It allows fast correction for each rotational degree of freedom and multiple axes of rotation in a single gesture. Moving two hands at once without increasing the distance between them translates the object. Scaling is activated by increasing and decreasing the distance between palms. HoloLeap does not use zooming as LFDs have a limited depth range. Scaling is provided as an alternative to facilitate model inspection and to easily provide an overview. Continuous rotation (spin) is activated with a double-hand rotation gesture.

Freehand Interaction with Large-Scale 3D Map Data

Extending the 3D model interaction framework, I implemented a gesture based interaction system prototype for the use-case - interaction with large-scale 3D map data visualized on a light field display in real time. 3D map data are streamed over Internet to the display end in real-time based on requests sent by the visualization application. On the user side, data is processed and visualized on a large-scale 3D light field display.

- The streaming and light field visualization is done on the fly without using any pre-rendered animations or images. After acquiring the interaction control messages from the Leap Motion Controller, rendering is performed in real-time on the light-field display's.
- For real-time visualization, HoloVizio OpenGL wrapper library is used which intercepts all OpenGL calls and sends rendering commands over the network as well as modify related data (such as textures, vertex arrays, VBOs, shaders etc.) on the fly to suit the specifications of the actual light field display.
- During interaction, panning in horizontal and vertical directions can be done by translating the virtual camera in the opposite direction by a given amount and is achieved using one hand (either left or right). For rotation, the virtual camera is made to mimic the hand rotation gestures. Zooming is achieved using two hands. Bringing two hands closer is the gesture for zooming out and taking the hands apart results in zooming in.

DOI:10.15774/PPKE.ITK.2015.005

- The presented system is first of its kind exploring the latest advances in 3D visualization and interaction techniques.

6.3.2 Exploring Direct 3D Interaction for Full Horizontal Parallax Light Field Displays Using Leap Motion Controller

I proposed the first framework that provides a realistic direct haptic interaction with virtual 3D objects rendered on a light field display. The solution includes calibration procedure that leverages the available depth of field and the finger tracking accuracy, and a real-time interactive rendering pipeline that modifies and renders light field according to 3D light field geometry and the input gestures captured by the Leap Motion Controller. The implemented interaction framework is evaluated and the results of a first user study on interaction with a light field display are presented. This is a first attempt for direct 3D gesture interaction with a full horizontal parallax light field display.

- The application generates random patterns of tiles in run time and rendered at a given depth. In parallel, the application receives interaction data from Leap Motion Controller, processes and updates the renderer in real-time. The controlling PC runs GL wrapper and feeds the resulting visual data to optical modules. Hence we can see the same application running on a LCD monitor in 2D and on light field display in 3D.
- Real-time visualization is achieved using OpenGL wrapper library. A controlling PC runs two applications: main OpenGL frontend rendering application for 2D LCD display and backend wrapper application that tracks the commands in current instance of OpenGL (front end application) and generates modified stream for light field rendering. The front end rendering application also receives and processes user interaction commands from Leap Motion Controller in real-time.
- The interaction and display spaces are calibrated to provide an illusion of touching virtual objects. To the best of my knowledge, the is a first study involving direct interaction with virtual objects on a light field display using Leap Motion Controller. The proposed interaction setup is very general and is applicable to any 3D display without glasses. The method is scalable, and the interaction space can easily be extended by integrating multiple Leap Motion Controllers.

Direct touch interaction system evaluation

I have conducted a user study to evaluate the proposed freehand interaction with the light field display through a simple within-subject user study with 12 participants.

Three tiles of the same size were displayed simultaneously and the participants were asked to point (touch) the surface of the red tile as perceived in space. The positions of the tiles varied

DOI:10.15774/PPKE.ITK.2015.005

from trial to trial to cover the entire FOV of the display. 3D and 2D display modes were used representing two different experimental conditions:

- In 2D mode, the displayed objects were distributed on a plane in close proximity of the display surface; and
- In 3D mode, the objects were distributed in a space with the distance varying from 0 to 7 cm from the display.

The 2D mode provided a control environment, which was used to evaluate the specifics of this particular interaction design: the performance and properties of the input device, display dimensions, specific interaction scenario (e.g., touching the objects), etc. Each participant was asked to perform 11 trials within each of the two conditions. The sequence of the conditions was randomized across the participants to eliminate the learning effect. The light field display and the interaction design were evaluated from the following aspects: task completion times, cognitive workload and perceived user experience.

Results show that users did not perceive any significant differences between the conditions in terms of general impression, the easiness to learn how to interact with the content, the efficiency of such interaction, the reliability or the predictability of the interfaces used and the excitement or the motivation for such an interaction. The exception is the novelty subscale where a tendency towards higher preferences for the 3D mode can be observed. The analysis of the post-study questionnaire revealed that the rendered objects were seen clearly in both experimental conditions. However, the users favored the 3D mode in terms of rendering realism. When asked to choose the easiest mode, the user's choices were equally distributed between both modes. However, when asked which mode led to more mistakes in locating the exact object position, two-thirds indicated the 3D mode, which is reflected also in longer task completion times in this particular mode. Finally, when asked about their preference, two-thirds of the participants chose the 3D mode as their favorite one.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 7

Applications of the Work

3D video is increasingly gaining prominence as the next major innovation in video technology that greatly enhances the quality of experience. Consequently, the research and development of technologies related to 3D video are increasingly gaining attention [59]. The research and development work done during this thesis work mainly finds applications related to the content transmission and rendering parts.

In recent years, the telepresence systems [60, 61] tend to be equipped with multiple cameras to capture the whole communication space; this integration of multiple cameras causes the generation of huge amount of dynamic camera image data. This large volume of data needs to be processed at the acquisition site, possibly requires aggregation from different network nodes and finally needs to be transmitted to the receiver site. It becomes intensely challenging to transmit this huge amount of data in real time through the available network bandwidth. The use of classical compression methods for reducing this large data might solve the problem to a certain degree, but using the compression algorithms directly on the acquired data may not yield sufficient data reduction. Finding the actual portion of data needed by the display system at receiver site and excluding the redundant image data would be highly beneficial for transmitting the image data in real time. The proposed automatic data reduction approach exploring the target display geometry might be handy in such situations.

The current generation 3D films are mainly based on *Steroscopic 3D*. Every year, more and more movies are produced in 3D and television channels started to launch their broadcast services for events such as sports and concerts in 3D. But despite these technological advances, the practical production of stereoscopic content that results in a natural and comfortable viewing experience in all scenarios is still a challenge [10]. Assuming that the displaying part has no limitations, the basic problem in a stereoscopic setting lies in the human visual perception [62] [63]. With the advances of more natural 3D displaying technologies, the visual performance has greatly increased. However, the problem of depth accommodation has slid to the display side and there is a need for techniques that address the display content adaption. The content adaptive depth retargeting presented in the work can be applied for automatically adapting the scene depth to

DOI:10.15774/PPKE.ITK.2015.005

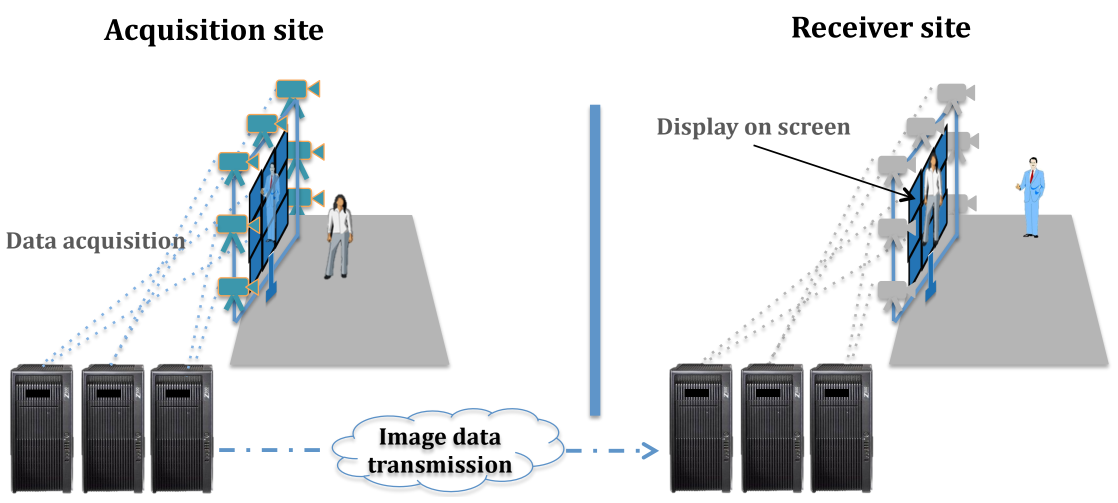


Figure 7.1: Multi-camera telepresence system.

the display depth. The method works in real-time and can be applied for automatic disparity correction and display adaption.

Light field displays offer several advantages over volumetric or autostereoscopic displays such as adjacent view isolation, increased field of view, enhanced depth perception and support for horizontal motion parallax. However, it is unclear how to take full advantage of these benefits, as user interface techniques with such displays have not been explored. Google street view development laid an important milestone for online map services. The prototyped interaction setups can be extended for applying to such navigation purposes.

DOI:10.15774/PPKE.ITK.2015.005

Chapter 8

Conclusions and Future Work

Projection-based light field displays enable us to view three-dimensional imagery without the need for extra equipment, such as glasses. These devices create immersive interactive environments with increased depth perception, and can accommodate multiple users. Their unique properties allow accurate visualization of 3D scenes and models without posing any constraints on the user's position. The usability of the system will be increased with further investigation into the optimal means of rendering and interaction. With fine 3D frame resolution, these displays provide a more sophisticated solution compared to other technologies, meeting all the requirements needed for the realistic display of 3D content.

In this dissertation, I presented valuable insights into the software components of projection-based light field displays. Exploring and extending the two broad philosophies for rendering real-world scenes on these displays, I presented several details that help in understanding the applicability and concerns of these displays for future 3DTV and associated applications. I also presented the first interaction framework using Leap Motion Controller that provides a realistic direct haptic interaction with virtual 3D objects rendered on a light field display. The work can be broadly classified into two categories - rendering and interaction. Few inferences from each group and possible directions for future work are presented in the following sub-sections.

8.1 Light Field Rendering

Two light rendering methods were examined which provide more insights on light field reconstruction from multiview images - plain light field rendering and all-in-focus rendering (simple and geometry based light field rendering). Plain light field rendering involves re-sampling the captured light field data base in the form of multiview images and typically for photo-realistic rendering, one may require very high number of cameras to substantially sample the light field. Given these views, it is possible to produce direction dependent effects for good quality 3D visualization. However, this quality comes at the cost of huge data size and is proportional to

DOI:10.15774/PPKE.ITK.2015.005

the number of available views. All-in-focus rendering requires the scene depth information for computing the light field and the quality of reconstruction is highly dependent on the accuracy of the calculated depth. Given that reasonably accurate depth is available, using this rendering approach it is possible to achieve good quality light field with less number of cameras. The scene geometry estimation helps in producing higher quality views from arbitrary view positions using less cameras. The usability and inherent effects of these two models on the transmission and rendering sides are discussed below.

8.1.1 Transmission Related Constraints

Plain light field rendering

In plain rendering method, as the process only involves re-sampling and the behavior remains temporally consistent, it is possible to establish unnecessary camera rays before hand given that, the display geometry is available. Following this idea, I presented a lossless approach to reduce the data flow in multi-camera telepresence systems using light field displays. The proposed method does not rely on image/video coding schemes, but rather uses the display projection geometry to exploit and eliminate redundancy. I proposed minor changes in the capturing, processing and rendering pipeline with an additional processing at the local transmission site that helps achieving significant data reduction.

In the work, I showed the use of global masks to reduce pixel data selectively. In practice, each rendering node does not need the whole information, even from the extracted pixel subset. Thus, it is possible to customize the masks for each of the render cluster nodes, which can further reduce the data. I made an assumption that the local and remote sites are connected via a low latency, relatively high-bandwidth connection. In general this may not be the case and in order to transmit the light field data over longer distances, it is possible to incorporate multiview coding schemes such as H.264, MVC and HEVC. Also the capturing speed at the acquisition site is a bottleneck in the used camera setup. Using constant exposure time cameras with hardware trigger might further increase the accuracy in the camera synchronization.

With the emergence of LF displays with extremely wide FOV, it is more and more apparent that an equidistant linear camera array cannot capture the visual information necessary to represent the scene from all around. A more suitable setup is an arc of cameras, facing the center of the scene. Compressing such captured information with MVC should also be efficient, as the views captured in this manner also bear more similarity than views captured by a linear camera array. However, the kind of pixel-precise inter-view similarity that MVC implicitly assumes only exist when using parallel cameras on a linear rig, and assuming Lambertian surfaces. It has been shown that the coding gain from inter-view prediction is significantly less for arc cameras than for linear cameras. Due to the emergence of wide-FOV 3D displays it is expected that non-linear multiview setups will be more significant in the future. Coding tools to support the efficient coding of views rotating around the scene center should be explored, and the similarities inherent in such

DOI:10.15774/PPKE.ITK.2015.005

views exploited for additional coding gains. The losless data reduction possibilities show that the candidate compression standard should allow the decoding of views with different resolutions.

All-in-focus light field rendering

The geometry calculation is a part of this rendering method. The camera rays which are used during the light field reconstruction are varying with time and thus direct data reduction exploring the display geometry is not an option. It is important for all the display rendering nodes to have all the captured camera light rays. However, from a broader view, the data requirement for this kind of rendering is considerably less than the plain rendering approach. Using a multi-resolution approach for depth estimation, using synthetic scenes, it is observed that all-in-focus rendering method can be used to produce similar quality light field as plain light field rendering with only 20% of the input data. In such a system, data reduction may be achieved through the encoding/decoding procedures.

8.1.2 Rendering Enhancement - Depth Retargeting

For projection-based light field displays, the spatial resolution is higher on the surface of the screen and the diminishes as we move away from screen. Therefore, to optimize the viewing experience on light field displays, the scene center must coincide with display's $Z = 0$ plane (the screen), total scene depth should comply with the limited depth of field of the display and the frequency details of the objects in the scene should be adjusted conforming to the displays spatial characteristics. While the latter can be addressed by adapting suitable rendering methods, I presented a retargeting method that focuses on addressing the former two goals. The aim of the method is to preserve the 3D appearance of salient objects in a scene.

Plain light field rendering

For depth retargeting in an adaptive way, we need to know the salient regions of the scene. As the plain rendering approach does not involve scene geometry computation, it is necessary to pre-process a set of camera images to obtain a global salience map. For retargeting part, it is necessary to have control over the rendered scene point locations in the display coordinates. A possible solution for this could be the following - we carry out a sparse correspondence matching on the captured images and use this information to pre-warp the captured images confirming the display depth requirements and use the warped images for rendering. This operation could be time consuming depending on the number input cameras.

DOI:10.15774/PPKE.ITK.2015.005

All-in-focus light field rendering

As depth estimation is a part of rendering, it is possible to achieve warped light field in an adaptive way modifying the camera rays. Using this rendering method, I presented a real-time plane sweeping algorithm which concurrently estimates and retargets scene depth. The presented method perceptually enhances the quality of rendering on a projection-based light field display in a real-time capture and rendering framework. To the best of my knowledge, this is the first real-time setup that reconstructs an adaptively retargeted light field on a light field display from a live multiview feed. The method is very general and is applicable to 3D graphics rendering on light field display as well as to real-time capture-and-display applications. I showed that adaptive retargeting preserves the 3D aspects of salient objects in the scenes and achieves better results from all the viewing positions than linear and logarithmic approaches.

In the view of the rendering related discussion, I believe that all-in-focus rendering approach has the capability to emerge as a standard for light field rendering and encoding/decoding routines should be further researched for suitable camera setups with wide baselines, having cameras in arc facing the center of the scene. One of the limitations of the presented real-time end-to-end system with adaptive depth retargeting is the inaccuracy of estimated depth values while retargeting and rendering. In future work, it can be interesting to employ additional active sensors to get an initial depth structure of the scene and use human visual system aware saliency estimation.

8.2 Light Field Interaction

I presented the design and implementation of HoloLeap - a gesture-based system for interacting with light field displays. HoloLeap used the Leap Motion Controller coupled with a HoloVizio screen to create an interactive 3D environment. I presented the design of the system with an array of gestures to enable a variety of object manipulation tasks. Extending the basic interaction setup, I presented the design of a direct touch freehand interaction with a light field display, which included touching (selecting) the objects at different depths in a 3D scene. Again, Leap Motion Controller was used as an input device providing desktop-based user-tracking device. One of the issues addressed in the work is a calibration procedure providing the transformation of 3D points from the Controller's to the display's coordinate system to get the uniform definition of position within the interaction space. This transformation is of vital importance for accurate tracking of users' gestures in the displayed 3D scene enabling interaction with the content and manipulation of virtual objects in real-time. The available interaction space has to be selected and customized based on the limitations of the Controller's effective sensory space as well as the display's non-uniform spatial resolution. The proposed calibration process results in an error less than 1 μm in a large part of interaction space.

The proposed interaction setup was evaluated by comparing the 3D interaction (e.g., pointing and touching) with objects in space to the traditional 2D touch of objects in a plane. The results

DOI:10.15774/PPKE.ITK.2015.005

of the evaluation revealed that more time is needed to select the object in 3D than in 2D. This was expected, since the additional dimension undoubtedly implies extra time that is needed to, firstly, cognitively process the visual information and, secondly, to physically locate the object in space. However, the poor performance of the interaction in 3D may also be contributed to by the intangibility of the 3D objects and the lack of tactile feedback. This is also in accordance with the findings of other similar studies where poor performance in determining the depth of the targets was related to the intangibility of the objects.

Another, perhaps even more interesting finding was a relatively low cognitive demand of interaction in 3D environment, which was comparable to the simplified 2D interaction scenario. This reflects the efficiency and the intuitiveness of the proposed interaction setup and the freehand interaction with 3D content in general. In the 2D environment, the touch screens built in the majority of smart phones and other portable devices also represented such intuitive and simple input devices enabling the adoption of these technologies by users of all ages and different computer skills. They successfully replaced computer mice and other pointing devices (e.g., very effective and widely used in a desktop environment) and their main advantage was the introduction of the p̈oint and select p̈aradigm, which seems to be very natural and intuitive. I believe the proposed freehand interaction setup could represent the next step in this transition enabling such direct selection and manipulation of content also in 3D environment. This assumption was confirmed also by the high preference of the users for the proposed setup expressed in the UEQ questioners.

The Leap Motion Controller sometimes produced anomaly readings, such as reporting identical position although the finger had moved, reporting false positions far away from the actual finger position or suddenly dropping out of recognition. These anomalies, however, were usually short-termed and did not represent a significant impact on the user's performance and the overall results. Nevertheless, as such anomalies were known to happen and therefore expected, the study was designed to cope with them: the conditions were randomized and a large number of trials was used within each condition so the anomalies were uniformly distributed among both conditions.

In the study I observed and evaluated only the process of selection of an object in a 3D scene. In general, more sophisticated actions can be performed while interacting with 3D content, such as advanced manipulation (e.g., changing position and orientation of virtual objects), changing of viewpoint of the scene, etc. In the future it can be interesting to evaluate the proposed interaction setup in a game-like scenario including both hands. Users can be asked to move different objects within the 3D scene and change their orientation or even change their shape.

Another aspect that needs to be evaluated in the future is the interaction with tactile (or/and some other kind of) feedback so the interaction will be similar to touch-sensitive surfaces in 2D. Finally, the comparison in performance between Leap Motion Controller and other, especially classic and user-familiar input devices, like computer mouse, will be also interesting.

DOI:10.15774/PPKE.ITK.2015.005

List of Publications

Journal Publications

[J1] **V. K. Adhikarla**, J. Sodnik, P. Szolgay, and G. Jakus, "Exploring direct 3D interaction for full horizontal parallax light field displays using leap motion controller," *Sensors*, pp. 8642-8663, April, 2015, Impact Factor : 2.245. 81

[J2] **V. K. Adhikarla**, F. Marton, T. Balogh, and E. Gobbetti, "Real-time adaptive content retargeting for live multi-view capture and light field display," *The Visual Computer, Springer Berlin Heidelberg*, vol. 31, May, 2015, Impact Factor : 0.957. 79

[J3] A. Dricot, J. Jung, M. Cagnazzo, B. Pesquet, F. Dufaux, P. T. Kovács, and **V. K. Adhikarla**, "Subjective evaluation of super multi-view compressed contents on high-end light-field 3D displays," *Signal Processing: Image Communication, Elsevier*, 2015, Impact Factor : 1.462. 25, 77

[J4] B. Maris, D. Dallálba, P. Fiorini, A. Ristolainen, L. Li, Y. Gavshin, A. Barsi, and **V. K. Adhikarla**, "A phantom study for the validation of a surgical navigation system based on real-time segmentation and registration methods," *International Journal of Computer Assisted Radiology and Surgery*, vol. 8, pp. 381 - 382, 2013, Impact Factor : 1.707. 79

Conference Publications

[C1] **V. K. Adhikarla**, G. Jakus, and J. Sodnik, "Design and evaluation of freehand gesture interaction for lightfield display," in *Proceedings of International Conference on Human-Computer Interaction*, pp. 54 - 65, 2015. 81

[C2] T. Balogh, Z. Nagy, P. T. Kovács, and **V. K. Adhikarla**, "Natural 3D content on glasses-free light-field 3D cinema," in *Proceedings of the SPIE: Stereoscopic Displays and Applications XXIV*, vol. 8648, Feb., 2013. 25, 36, 77

DOI:10.15774/PPKE.ITK.2015.005

[C3] **V. K. Adhikarla**, A. Tariqul Islam, P. Kovacs, and O. Staadt, "Fast and efficient data reduction approach for multi-camera light field display telepresence systems," in *Proceedings of 3DTV-Conference 2013*, Oct 2013, pp. 1-4. 25, 34, 77

[C4] P. Kovács, Z. Nagy, A. Barsi, **V. K. Adhikarla**, and R. Bregovic, "Overview of the applicability of h.264/mvc for real-time light-field applications," in *Proceedings of 3DTV-Conference 2014*, July 2014, pp. 1-4. 25, 35, 77

[C5] P. Kovacs, K. Lackner, A. Barsi, **V. K. Adhikarla**, R. Bregovic, and A. Gotchev, "Analysis and optimization of pixel usage of light-field conversion from multi-camera setups to 3D light-field displays," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 86-90. 25, 77

[C6] R. Olsson, **V. K. Adhikarla**, S. Schwarz, and M. Sjöström, "Converting conventional stereo pairs to multiview sequences using morphing," in *Proceedings of the SPIE: Stereoscopic Displays and Applications XXIII*, vol. 8288, 22 - 26 January 2012. 79

[C7] **V. K. Adhikarla**, P. T. Kovács, A. Barsi, T. Balogh, and P. Szolgay, "View synthesis for lightfield displays using region based non-linear image warping," in *Proceedings of International Conference on 3D Imaging (IC3D)*, Dec 2012, pp. 1-6. 79

[C8] **V. K. Adhikarla**, A. Barsi, P. T. Kovács, and T. Balogh, "View synthesis for light field displays using segmentation and image warping," in *First ROMEO workshop*, July 2012, pp. 1-5. 79

[C9] **V. K. Adhikarla**, F. Marton, A. Barsi, E. Gobbetti, P. T. Kovacs, and T. Balogh, "Real-time content adaptive depth retargeting for light field displays," in *Proceedings of Eurographics Posters*, 2015. 79

[C10] **V. K. Adhikarla**, P. Wozniak, A. Barsi, D. Singhal, P. Kovács, and T. Balogh, "Freehand interaction with large-scale 3d map data," in *Proceedings of 3DTV-Conference 2014*, July 2014, pp. 1-4. 81

[C11] **V. K. Adhikarla**, P. Wozniak, and R. Teather, "Hololeap: Towards efficient 3D object manipulation on light field displays," in *Proceedings of the 2Nd ACM Symposium on Spatial User Interaction (SUI)*, 2014, pp. 158-158. 81

Other Publications

[O1] P. Kovács, A. Fekete, K. Lackner, **V. K. Adhikarla**, A. Zare, and T. Balogh, "Big buck bunny light-field test sequences," in *ISO/IEC JTC1/SC29/WG11 M35721*, Feb., 2015. 25, 77

DOI:10.15774/PPKE.ITK.2015.005

[O2] **V. K. Adhikarla**, "Content processing for light field displaying," in *Poster session, EU COST Training School on Plenoptic Capture, Processing and Reconstruction*, June, 2013. 25, 77

[O3] **V. K. Adhikarla**, "Data reduction scheme for light field displays," in *Poster session, EU COST Training School on Rich 3D Content: Creation, Perception and Interaction*, July, 2014. 25, 77

DOI:10.15774/PPKE.ITK.2015.005

References

[1] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '96*. : ACM, 1996, pp. 31–42. 1

[2] B. Masia, G. Wetzstein, P. Didyk, and D. Gutierrez, "A survey on computational displays: Pushing the boundaries of optics, computation, and perception," *Computers & Graphics*, vol. 37, no. 8, pp. 1012 – 1038, 2013. 3, 4

[3] T. Agocs, T. Balogh, T. Forgacs, F. Bettio, E. Gobbetti, G. Zanetti, and E. Bouvier, "A large scale interactive holographic display," in *Proceedings of IEEE Virtual Reality Conference (VR 2006)*, , 2006, p. 57. 3

[4] T. Balogh, P. Kovacs, and A. Barsi, "Holovizio 3d display system," in *Proceedings of 3DTV-Conference, 2007*, May 2007, pp. 1–4. 3, 15

[5] T. Balogh, "Method and apparatus for displaying 3d images," Patent US6 999 071 B2, 2006. 3

[6] W. Matusik and H. Pfister, "3D TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 814–824, 2004. 8, 20

[7] R. Yang, X. Huang, S. Li, and C. Jaynes, "Toward the light field display: Autostereoscopic rendering via a cluster of projectors," *IEEE Transactions on Visualization & Computer Graphics*, vol. 14, pp. 84–96, 2008. 8, 20

[8] T. Balogh and P. Kovács, "Real-time 3D light field transmission," in *Proceedings of SPIE*, vol. 7724, 2010, pp. 772 406–772 406–7. 8, 21

[9] F. Marton, E. Gobbetti, F. Bettio, J. Guitian, and R. Pintus, "A real-time coarse-to-fine multiview capture system for all-in-focus rendering on a light-field display," in *Proceedings of 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), 2011*, May 2011, pp. 1–4. 8, 12, 22, 45, 52, 53

DOI:10.15774/PPKE.ITK.2015.005

[10] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross, “Nonlinear disparity mapping for stereoscopic 3d,” *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 75:1–75:10, Jul. 2010. 8, 45, 54, 85

[11] D. Shreiner and T. K. O. A. W. Group, *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 3.0 and 3.1*, 7th ed. Addison-Wesley Professional, 2009. 9

[12] J. Nickolls, I. Buck, M. Garland, and K. Skadron, “Scalable parallel programming with cuda,” *Queue*, vol. 6, no. 2, pp. 40–53, Mar. 2008. 9

[13] Leap Motion Inc. (2012-2014) Leap motion developer portal. [Online]. Available: https://developer.leapmotion.com/ 9, 60

[14] M. Agus, E. Gobbetti, A. Jaspe Villanueva, G. Pintore, and R. Pintus, “Automatic geometric calibration of projector-based light-field displays,” in *Proceedings of Eurovis Short Papers*, June 2013, pp. 1–5. 13, 54

[15] M. Brown, A. Majumder, and R. Yang, “Camera-based calibration techniques for seamless multiprojector displays,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 2, pp. 193–206, Mar. 2005. 13

[16] M. Agus, E. Gobbetti, J. A. Iglesias Guitián, F. Marton, and G. Pintore, “GPU accelerated direct volume rendering on an interactive light field display,” *Computer Graphics Forum*, vol. 27, no. 2, pp. 231–240, 2008. 13, 14, 45

[17] A. Jones, I. McDowall, H. Yamada, M. T. Bolas, and P. E. Debevec, “Rendering for an interactive 360 degree light field display,” *ACM Transactions on Graphics*, vol. 26, no. 3, p. 40, 2007. 14, 45

[18] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, “Plenoptic sampling,” in *Proceedings of SIGGRAPH*, 2000, pp. 307–318. 20

[19] R. Yang, G. Welch, and G. Bishop, “Real-time consensus-based scene reconstruction using commodity graphics hardware,” in *Proceedings of Pacific Graphics*, 2002, pp. 225–. 21

[20] Y. Kunita, M. Ueno, and K. Tanaka, “Layered probability maps: basic framework and prototype system,” in *Proceedings of Virtual Reality Software and Technology*, 2006, pp. 181–188. 21

[21] Y. Taguchi, K. Takahashi, and T. Naemura, “Real-time all-in-focus video-based rendering using a network camera array,” in *Proceedings of 3D Data Processing, Visualization and Transmission*, 2008, pp. 241–244. 21

[22] Y. Taguchi, T. Koike, K. Takahashi, and T. Naemura, “TransCAIP: A live 3D TV system using a camera array and an integral photography display with interactive control of viewing

DOI:10.15774/PPKE.ITK.2015.005

parameters," *IEEE Transactions on Visualization & Computer Graphics*, vol. 15, pp. 841–852, 2009. 21

[23] R. Bolles, H. Baker, and D. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *International Journal of Computer Vision*, vol. 1, no. 1, pp. 7–55, 1987. 26

[24] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96. : ACM, 1996, pp. 43–54. 28

[25] J. Shade, S. Gortler, L.-w. He, and R. Szeliski, "Layered depth images," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. : ACM, 1998, pp. 231–242. 28

[26] X. Tong, J. Chai, and H.-Y. Shum, "Layered lumigraph with lod control," *The Journal of Visualization and Computer Animation*, vol. 13, no. 4, pp. 249–261, 2002. 29

[27] A. Isaksen, L. McMillan, and S. J. Gortler, "Dynamically reparameterized light fields," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. : ACM Press/Addison-Wesley Publishing Co., 2000, pp. 297–306. 29

[28] A. Davis, M. Levoy, and F. Durand, "Unstructured light fields," *Comp. Graph. Forum*, vol. 31, no. 2pt1, pp. 305–314, May 2012. 30

[29] R. Bolles, H. Baker, and D. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *International Journal of Computer Vision*, vol. 1, no. 1, pp. 7–55, 1987. 30

[30] D. N. Wood, D. I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. H. Salesin, and W. Stuetzle, "Surface light fields for 3d photography," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. : ACM Press/Addison-Wesley Publishing Co., 2000, pp. 287–296. 30

[31] A. Gelman, J. Berent, and P. Dragotti, "Layer-based sparse representation of multiview images," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, 2012. 30

[32] P. Rademacher and G. Bishop, "Multiple-center-of-projection images," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. : ACM, 1998, pp. 199–206. 31

[33] Y. Chen, Y.-K. Wang, K. Ugur, M. M. Hannuksela, J. Lainema, and M. Gabbouj, "The emerging mvc standard for 3d video services," *EURASIP J. Appl. Signal Process.*, vol. 2009, pp. 8:1–8:13, Jan. 2008. 34

DOI:10.15774/PPKE.ITK.2015.005

[34] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multiview imaging and 3DTV," *IEEE Signal Processing Magazine*, vol. 24, no. 6, p. 10, 2007. 45

[35] A. Smolic, "3D video and free viewpoint video – from capture to display," *Pattern Recognition*, vol. 44, no. 9, pp. 1958 – 1968, 2011. 45

[36] J. Iglesias Guitián, E. Gobbetti, and F. Marton, "View-dependent exploration of massive volumetric models on large scale light field displays," *The Visual Computer*, vol. 26, no. 6–8, pp. 1037–1047, 2010. 45

[37] C. Kim, A. Hornung, S. Heinzle, W. Matusik, and M. Gross, "Multi-perspective stereoscopy from light fields," *ACM Transactions on Graphics*, vol. 30, no. 6, pp. 190:1–190:10, December 2011. 45

[38] B. Masia, G. Wetzstein, C. Aliaga, R. Raskar, and D. Gutierrez, "Display adaptive 3d content remapping," *Comput. Graph.*, vol. 37, no. 8, pp. 983–996, Dec. 2013. 45, 55

[39] P. Didyk, T. Ritschel, E. Eisemann, K. Myszkowski, and H.-P. Seidel, "A perceptual model for disparity," *ACM Transactions on Graphics*, vol. 30, no. 4, 2011. 45

[40] C. Birklbauer and O. Bimber, "Light-field retargeting," *Computer Graphics Forum*, vol. 31, no. 2pt1, pp. 295–303, 2012. 46

[41] V. Kraevoy, A. Sheffer, A. Shamir, and D. Cohen-Or, "Non-homogeneous resizing of complex models," vol. 27, no. 5, p. 111, 2008. 46

[42] D. Graf, D. Panozzo, and O. Sorkine-Hornung, "Mobile image retargeting," in *Proceedings of Vision, Modeling and Visualization*, 2013. 46, 47

[43] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," in *Proceedings of European Conference on Computer Vision*, 1994, pp. 151–158. 52

[44] C. Wingrave, B. Williamson, P. D. Varcholik, J. Rose, A. Miller, E. Charbonneau, J. Bott, and J. LaViola, "The wiimote and beyond: Spatially convenient devices for 3d user interfaces," *Computer Graphics and Applications, IEEE*, vol. 30, no. 2, pp. 71–85, March 2010. 59

[45] G. Bruder, F. Steinicke, and W. Sturzlinger, "To touch or not to touch?: Comparing 2d touch and 3d mid-air interaction on stereoscopic tabletop surfaces," in *Proceedings of the 1st Symposium on Spatial User Interaction*, ser. SUI '13. : ACM, 2013, pp. 9–16. 59

[46] T. Grossman, D. Wigdor, and R. Balakrishnan, "Multi-finger gestural interaction with 3d volumetric displays," in *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '04. : ACM, 2004, pp. 61–70. 59

DOI:10.15774/PPKE.ITK.2015.005

[47] R. Wang, S. Paris, and J. Popović, “6d hands: Markerless hand-tracking for computer aided design,” in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST ’11. : ACM, 2011, pp. 549–558. 59

[48] A. Butler, O. Hilliges, S. Izadi, S. Hodges, D. Molyneaux, D. Kim, and D. Kong, “Vermeer: Direct interaction with a 360&#176; viewable 3d display,” in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST ’11. : ACM, 2011, pp. 569–576. 59

[49] Z. Zhang, “Microsoft kinect sensor and its effect,” *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, Apr. 2012. 59

[50] D. Vogel and R. Balakrishnan, “Distant freehand pointing and clicking on very large, high resolution displays,” in *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST ’05. : ACM, 2005, pp. 33–42. 59

[51] O. Hilliges, D. Kim, S. Izadi, M. Weiss, and A. Wilson, “Holodesk: Direct 3d interactions with a situated see-through display,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI ’12. : ACM, 2012, pp. 2421–2430. 59

[52] L.-W. Chan, H.-S. Kao, M. Y. Chen, M.-S. Lee, J. Hsu, and Y.-P. Hung, “Touching the void: Direct-touch interaction for intangible displays,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI ’10. : ACM, 2010, pp. 2625–2634. 59

[53] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, “Robust part-based hand gesture recognition using kinect sensor,” *Multimedia, IEEE Transactions on*, vol. 15, no. 5, pp. 1110–1120, Aug 2013. 60

[54] MyVR software. (Accessed: Apr. 14, 2014) mmap sdk. [Online]. Available: http://bit.ly/1hAucMa 65

[55] B. Bellekens, V. Spruyt, and R. B. Maarten Weyn, “A survey of rigid 3d pointcloud registration algorithms,” in *Proceedings of Fourth International Conference on Ambient Computing, Applications, Services and Technologies*, 2014, pp. 8–13. 68

[56] D. G. R. Bradski and A. Kaehler, *Learning Opencv, 1st Edition*, 1st ed. O’Reilly Media, Inc., 2008. 72

[57] G. H. Sandra and E. S. Lowell, “Development of a multi-dimensional workload rating scale: Results of empirical and theoretical research,” *Human Mental Workload*, pp. 139–183, 1988. 74

[58] B. Laugwitz, T. Held, and M. Schrepp, “Construction and evaluation of a user experience questionnaire,” in *Proceedings of the 4th Symposium of the Workgroup Human-Computer*

DOI:10.15774/PPKE.ITK.2015.005

Interaction and Usability Engineering of the Austrian Computer Society on HCI and Usability for Education and Work, ser. USAB '08. : Springer-Verlag, 2008, pp. 63–76. 74

[59] F. Dufaux, B. Pesquet-Popescu, and M. Cagnazzo, *Emerging Technologies for 3D Video: Creation, Coding, Transmission and Rendering*. Wiley, May 2013. 85

[60] A. Maimone and H. Fuchs, "Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras," in *10th IEEE International Symposium on Mixed and Augmented Reality*, October 2011, pp. 137 –146. 85

[61] B. Petit, J.-D. Lesage, C. Menier, J. Allard, J.-S. Franco, B. Raffin, E. Boyer, and F. Faure, "Multicamera real-time 3D modeling for telepresence and remote collaboration," *International Journal of Digital Multimedia Broadcasting*, vol. 2010, pp. 247 108–12, 2009. 85

[62] I. P. Howard and B. J. Rogers, *Seeing in Depth*. Oxford University Press, New York, USA, 2008. 85

[63] D. Hoffman, A. Girshick, K. Akeley, and M. Banks, "Vergence-accommodation conflicts hinder visual performance and cause visual fatigue," *Journal of Vision*, vol. 8, no. 3, p. 33, 2008. 85