USC-SIPI REPORT #392

Stereo Panorama Virtual Environment

by

Chiao Wang

May 2008

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Viterbi School of Engineering
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Suite 400
Los Angeles, CA 90089-2564 U.S.A.

STEREO PANORAMA VIRTUAL ENVIRONMENTS

by

Chiao Wang

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

May 2008

Acknowledgements

I am very blessed to have so many special individuals in my life. Their personal support and practical help made this dissertation possible. Therefore, I would like to express my sincere gratitude to each of them.

First of all, I would like to thank Prof. Alexander A. Sawchuk for not only being my dissertation advisor but also my mentor. Without his thoughtful guidance and encouragement, I would not have been able to finish it. From the beginning, I know he is a great teacher when I first attended his class. No matter what kind questions the students asked, he always had patience and answered questions in a clear way. And he is full of ideas to inspire our passion toward research and learning. Then, through my doctoral work, he allowed me to have the creative freedom in my research in which I developed my independent thinking and research skills. And even though he has many responsibilities as being a department chairman, he still arranged time to discuss research with me or assist me with writing. I could not find a better advisor than him who taught me both academic expertise and lifelong knowledge and discipline.

I am also truly thankful to Prof. B. Keith Jenkins and Prof. Larry Pryor for serving on my dissertation examination committee. Their constructive comments, invaluable time, consideration and kindness are immeasurable assets for my research. I also owe a debt of thanks to Prof. Krishna Nayak, Prof. Roger Zimmermann, and Prof. Margaret McLaughlin, who were on my qualifying examination committee, and Prof. Albert Rizzo. Their extensive knowledge guided me through the early stage of dissertation.

I also appreciate the friendship, assistance, and cooperation from my colleagues: Shih-Ching Yeh, C.Y. Chang, Zahir Y. Alpaslan, Tawei Ho, Nopparit Intharasombat, Stephan

Themis, J. Garagate, and Jaimin Bhise. Working with them was a really wonderful experience. Their helpful discussions made the research work more joyful.

Moreover, I would like to thank all the professors who had taught me at USC. The knowledge I learned became a foundation for my research. And I also thank many EE staff members including Gloria Halfacre, Dr. Alan Weber, Seth Scafani, Tim Boston, Diane Demetras, Alma Hernandez, and Talyia Veal, who provided me with great assistance.

Thanks are extended to my friends, office mates, and research group members: Dr. Keng-Ling Ray, Dr. Min-Shin Chen, Chuping Liu, Bei Wang, Wan-Jen Huang, Pankaj Mishra, Ming-Ying Chou, I-Feng Chang, Hui-Chi Chen, Hsiang-Tai Huang, Yen-Liang Lu, Yunsong Huang, Jing Li, Bo Wu, Anne Burton, Junnita, Chun-Bo Chang, Joyce Yim, Judy Hsiao, Sandy Weng, Jeff Chiu, Wei-Nan Lin, Chih-Ying Lin, Ying-Ying Tang, Ranya Ku, Hsiao-Ping Wu, Yu Zhou, Yi-Chun Ou, Jimmy Liu, etc. They lessened hardships and let good experiences become great experiences.

Most importantly, I would like to express my deepest gratitude to my parents, Jaw-Ming Wang and Yu-Shan Lien, whose unwavering love and support are always unlimited. My sister, Chi Wang, and brother, Ling Wang, have undoubtedly played important roles in my study. I really appreciate them for being my best cheer squad. I also deeply appreciate that my grandparents comforted me with kind and thoughtful words through difficult times. A special thank goes to my parents in law, Byung Ro Lee and Young Sin Park, and their friends. Their understanding and support are important for me.

Finally, I would express my sincere gratitude to my significant other, William Lee, for his endless love, incredible patience, and support. Because of that, I have the source to keep working. His support and encouragements had helped me pass several turbulences to achieve my goals.

Table of Contents

List of Tables

List of Figures

List of Variables

N_v: The view number.

(k,l): The individual sub-pixel coordinates of result image.

k_{offset}: The offset adjustment amount for better image quality.

N_{tot}: The total number of input images or channels.

α_L: The slant angle of the lenticular sheet.

N_{sub}: The number of sub-pixels per lenticule.

$H(x,y,c)$: The sub-pixel coordinates of the output image.

$G_m(x_o,y_o,c)$: The sub-pixel coordinates of image m, where c is the color index.

α: Half of the vergence (convergence) angle which is formed by two viewing axes.

a: The distance between the eye and an object.

B: The distance between two eyes.

f: The focal length of the camera.

F: The fixation point.

O_l: The center of left eye.

O_r: The center of left eye.

f_l, f_r: The images of point F on left/right retina.

p_l, p_r: The images of point P on left/right retina.

q_l, q_r: The images of point Q on left/right retina.

η: The binocular disparity.

a_F: The distance between eye to the point F.

a_Q: The distance between eye to the point Q.

W_X : The width of the captured image.

W_s : The width of a sampled strip.

W_v: The position of a sampled strip.

θ: The rotation angle of the camera.

(X, Y, θ): The coordinate system as shown in Figure 3.3.

$(x_{r1}, y_{r1}, \theta_{r1})$: The location of a particular (right view) pixel included in a strip that is part of the complete right panorama.

O_r: The rotation center.

O_c: The camera center.

O_{cl}: The camera center of the left view.

O_{cr}: The camera center of the right view.

D_{oc}: The radius of the circle forming the camera center.

D_{max}: The radius of the furthest circle which defines the region of interest (ROI).

D_{min}: The radius of the nearest circle which defines the region of interest (ROI).

P_{fox} : The point of fixation.

D_f: The radius of the circle path of the point of fixation.

$\Delta\theta$: The rotation angle between two shots.

$\Delta\theta_{rl}$: The angle between the shots where we start sampling the strip to form the left and right panorama.

θ_w: The half angle of $\Delta\theta_{rl}$.

θ_v: As denoted in Figure 3.5.

D_p: The distance of a scene point to the camera.

Δs_x: The horizontal disparity of a point at distance D_p.

P_A^i: The position of the i th feature point in frame A.

M_{BA}: The fundamental matrix calculated based on frames A and B.

w_k: The weighting factor for frame *k*.

$Y(i,k)$: The Y component of the (Y,I,Q) color representation of the pixel i in strip sampled from frame k.

$V(i,k)$: The local variance of $Y(i,k)$ for each pixel.

$P(i,n)$: The normal probability density we defined for each pixel $Y(i,k)$.

$\tilde{P}_c(i,n)$: The mixture-Gaussian probability density for the pixel *i*.

t: Certain image frame taken time.

δt: Time difference between two image frame .

δx, δy, δz: The movement of the camera in x, y, z world coordinate during certain time frame.

$I(x,y,t)$: The intensity of a pixel at location (x,y,t).

$I_x(x,y,t)$, $I_y(x,y,t)$: The differential of the intensity $I(x,y,t)$ in x direction.

$I_t(x,y,t)$: The differential of the intensity $I(x,y,t)$ in time.

$\lambda_{\min}$: Minimum eigen value of the matrix.

m: The number of edges of minimum spanning tree of a graph in P.39.

n: The number of vertices of minimum spanning tree of that graph.

ω : Size of the window which defined the neighbors of a pixel.

m : The number of neighboring frames used for the corresponding strips in Sec.4.3.2.

y_r : The vertical position of a 2D image point that is a projection of a 3D point in the right view.

y_l : The vertical position of a 2D image point that is a projection of a 3D point in the left view.

Z_r : The vertical position of a 3D scene point to the coordinate of the right camera.

Z_l : The vertical position of a 3D scene point to the coordinate of the left camera.

Δy : The vertical disparity.

s : Image size or the number of pixels in a image.

v : The motion velocity between two frames.

w : The strip width in Figure 5.8.

h : The strip height or the number of pixels in a column.

v_p : The virtual velocity (less than the strip width) between left and right panorama.

P_l , P_r : A vector which lists coordinates of features in one segment in the left or right panorama.

$w \times h$: The size of the user selected region.

(x_c, y_c) : The user selected region center.

$(w + m_w) \times (h + m_h)$: The size of the region (will be processed by mean-shift algorithm).

k : The number of control points denoted for the B-splins.

p : The degree of a basis function.

(x_t, y_t) : The center of matching template.

d_h , d_v : Maximum horizontal/ vertical disparity assumed for the selected region.

H_i : The current histogram of each color component.

H_t : The template histogram.

$d(H_i, H_t)$: The dissimilarity between the current and template histogram.

(X_{pr}, Y_{pr}) : The coordinate system of the right panorama.

(x_{pr}, y_{pr}) : The position of a particular point in the right panorama.

$\Delta s_x'$: The adjusted disparity.

$\Delta\theta'_{rl}$: The angle between the shots where we start sampling the strip to form the left and right panorama in the virtual stereo panorama generation process for adjusting disparity.

O_{cl}': The camera center of the left view panorama in the virtual stereo panorama generation process for adjusting disparity.

θ_l^s : The index of the starting frame of the left panorama.

(x_l, y_l, θ_l) : This index corresponds to a point position within a strip taken from frame indexed by θ_l to generate the left panorama.

(x_l', y_l', θ_l') : The index corresponds to a point position of a pixel, which is used to generate the virtual left panorama for adjusting disparity to fill the taken out the pixel with index (x_l, y_l, θ_l), within a strip taken from frame indexed by θ_l'.

$\mathbf{v}_1 = (x_1, y_1, z_1)$, $\mathbf{v}_2 = (x_2, y_2, z_2)$ and $\mathbf{v}_3 = (x_3, y_3, z_3)$: Three vertices of the triangle.

$\hat{z}(x, y)$: The estimated disparity value.

$z(x, y)$: The disparity value from disparity map.

N : Total number of pixels in a block in Eq. (7.2).

α_{ij}, β_{ij} : Two angles opposite edge (i, j).

$(-B/2, 0, 0)$, $(B/2, 0, 0)$: The location of optical centers of left and right camera.

$(x_l,0)$, $(x_r,0)$, $(x_c,0)$: Positions where a 3D point $(X,0,Z)$ is projected on the image plane of left, right, and center camera.

d_{off}: The disparity denoted for off-axis camera setup.

d_{toe}: The disparity denoted for toe-in camera setup.

Z: The distance which can be calculated from the disparity map and minimum and maximum scene distance.

$s(x,y)$: The disparity of pixel (x,y).

$s'(x,y)$: The disparity after processed with the asymmetric filter.

$g(u,\sigma_v)$, $g(v,\sigma_v)$: The horizontal and vertical Gaussian filter.

t: The maximum depth discontinuity.

d_o: The original disparity.

d_n: The morphed disparity.

d_p: Our final disparity value.

τ: The distance to the edge.

${}^{U}_{W}A$: The homogenous transformation matrix between world and user coordinates.

θ: Yaw (rotation angles relative to Z axis) in Eq. (8.1).

α: Roll (rotation angles relative to X axis) in Eq. (8.2).

β: Pitch (rotation angles relative to Y axis) in Eq. (8.3).

S: The scaling matrix in Eq. (8.4).

T: The matrix used for translation in Eq. (8.5).

Abstract

In this thesis, we describe the creation of virtual environment systems that can capture and display stereo images. The system can use a variety of display formats including auto-stereoscopic (AS) displays, shutter glasses, and head-mounted display (HMDs), and is suitable for virtual tours of outdoor or indoor sites such as a city tour, museums or galleries. We describe techniques for stereo panoramic image capture, present a general image capture model, specify geometrical parameters, and summarize how various physical parameters affect the perceived depth and image quality. We then describe an efficient stitching algorithm that corrects dynamic exposure variation and removes moving objects without manual selection of ground-truth images.

We also present expressions for the horizontal and vertical disparity of the captured image and describe disparity measurement techniques. We review various disparity adjustment algorithms, and develop an object-based horizontal disparity adjusting algorithm that changes the disparities to enhance or reduce the stereo visual effect for a selected object region rather than for columns of the images.

We describe a disparity morphing tool that enables users to specify the region of interest, select the disparity adjusting methods, and see the results directly on a AS display or with shutter glasses. We also describe the results of several subjective tests that compare system performance using different interaction tools and various displays such as HMD, shutter glasses, or AS displays. Future extension may use a large screen partial or fully panoramic display. In these systems, smaller auto-stereoscopic (AS) panels may be merged for a large screen or panoramic effort, or passive glasses-based technology (polarizing or anaglyph) may provide the stereo effect.

Chapter 1 Introduction

1.1 Motivation and Objective

Presence is defined as the subjective experience of being in one place while physically being situated in another. Immersion is a state characterized by perceiving oneself to be enveloped by, included in, and interacting in an environment that provides a continuous stream of stimuli [10]. *Immersive technology* is the creation of a complete aural and visual environment that places a participant or group of participants in a virtual space where they can interact with local or distant information and communicate naturally with others who may be in different physical locations. Virtual Reality (VR) and other immersive systems have been implemented that may fundamentally alter the way people interact with the world and with information [14]. Research in virtual reality and other immersive systems to process images to enable a walkthrough has been an active topic in recent years. Cylindrical panoramas used for virtual reality (VR) and immersive systems have a long history due to their 360 degree viewing capability and simple format. Many panoramic camera systems have been designed [69][46] and panorama-type monoscopic VR systems have been studied [92][70], including single-viewer, multi-viewer, projection-device, or multi-monitor based systems.

Recently, new types of 3D stereo visual displays have been developed, particularly autostereoscopic (AS) displays. AS displays produce a 3D visual sensation to one or more observers without the use of traditional clumsy glasses, goggles, helmets, or head-tracking [4][5]. They have become available commercially from many different companies. A Stanford report indicates that two million 3D display units were shipped in

2003, and suggests that 3D displays may soon become a common part of everyday life [29].

The goal of this research is to design virtual environment systems based on stereo panoramic imaging and various types of display, including AS displays. The system could be used for virtual tours of outdoor sites or indoor sites such as museums or galleries. To achieve this goal, we discuss the stereo panorama generation process and present a detailed and more general image capture model for our imaging system. The image capture system is based on a on a swing panorama camera structure: a single-view camera is mounted on a rotation arm with a rotation axis behind it. With this model, we describe the effects of physical parameters affect on the perceived depth and image quality effects, and determine the sampled strip width and position for the stereo panorama generation process. We also describe the horizontal and vertical disparity and develop a vertical disparity correction algorithm. We then present object-based horizontal disparity-adjusting algorithms that selectively enhance/reduce the stereo visual effect. To extend our research, we designed a disparity manipulation framework for general stereo images and video editing. Finally, we design a methodology to evaluate virtual environments with different user interaction devices and displays.

1.2 Organization

The organization of this thesis is as follows.

The Chapter 2 provides background information on panoramic virtual environment systems, stereoscopic technology, and human depth perception. Also, several important related works are discussed in this chapter.

In Chapter 3, the image capturing system is introduced and analyzed in detail. We give comprehensive mathematical models that pave the ground for the stereo panorama generation and disparity adjusting process in the remaining chapters.

The stereo panorama generation process is introduced in Chapter 4. There are two main processes: image registration and stitching. We give the mathematical details of both of these two processes.

In Chapter 5, the vertical and horizontal disparities are described, while disparity measurement techniques are discussed. We also present the algorithm to correct vertical disparity and perform the horizontal disparity adjustment.

In Chapter 6, we selectively enhance or reduce the perceived sense of depth perception. Details on the so-called on “mean-shift” segmentation algorithm for object selection are also described here. We propose an interactive disparity adjusting tool, which allows the user to select the region not only from manual input using a cursor but also by defining an area with a certain distance range, and with the ability to observe the results in real-time on an auto-stereoscopic or other stereo display.

In Chapter 7, we describe a general framework of disparity morphing and manipulation for stereo images and video.

In Chapter 8, a method to create a virtual environment based on our stereo panorama is described.

In Chapter 9, we present a methodology to evaluate virtual environments that equipped with various interaction and displaying devices.

In Chapter 10, we discuss several possible future studies including stereo panorama video, applications for military training, and present conclusions and a summary.

1.3 Contributions

The contributions of this thesis to panoramic stereoscopic virtual reality systems are summarized as follows.

(1) Mathematical Models for Swing Panorama Capturing Structure and Relevant Parameters for Stereo Panorama Generation Process. We investigate the swing panorama image capturing system, and formulate a mathematical model for the given geometry. This model gives a better understanding of the stereo panorama capturing and generation process.

(2) Mixture Gaussian Image Stitching Algorithm. We propose an image stitching algorithm based on the mixture Gaussian density function. In our results, it shows the ability to balance dynamic luminance and remove moving-objects at the same time without manually selecting ground truth images.

(3) Object-Based Horizontal Disparity Adjusting Algorithm. To further change the stereo sensation for different objects, we develop a disparity-adjusting algorithm employing an object-selecting algorithm based on the mean-shift image segmentation method. The goal is to modify only the disparities in the selected region while those in other areas are unchanged. This method is preferable for stereo image editing.

(4) Disparity Manipulation Framework for Stereo Images and Video. The framework consists of three parts: disparity map generation, disparity map manipulating/editing, and stereo image synthesis. We first discuss disparity map generation techniques for different original input data types. Then, we describe three methods for user manipulation of the disparity map. In the first, the user employs an interactive object-selecting tool by inputting seed points near the desired object boundary. Given the

selected objects, the user defines input-output disparity mapping curves for each object. In the second method, the user arbitrary manipulates a 3D disparity surface and our system calculates the new 3D surface after the user editing. A third method provides conversions between the two common stereo camera capture setups: “toe-in” and “off-axis” (we present their mathematical description). Finally, we describe disparity-based image rendering to synthesize new stereo image pairs from given original stereo image pairs based on a morphed disparity map. The synthesis method includes image warping, data-filling and disparity map smoothing procedures.

(5) Evaluation Methodology for Virtual Environments. We design a general methodology to evaluate virtual environments. We consider different factors that influence the user performance and design a series of questionnaires.

We have published several papers [108][109][110][111][112][113][119] based on this work.

Chapter 2 Background

In this chapter, we provide background information for this thesis proposal. In addition, we also discuss several important works related to our research.

2.1 Panorama-Based Virtual Reality Systems

Virtual Reality (VR) is a simulated environment intended to provide a vivid sense of presence by giving users visual, auditory, tactile or olfactory. VR techniques includes a very wide range of techniques, including different displays, image capturing and generating methods, and various tracking systems [40][82]. The performance of the VR systems is discussed by E. B. Nash *et al* [67]. Most companions of VR systems focus primarily on visual experiences and are roughly classified into the following categories according to the display type:

HMD-based VRs are usually equipped with one or more HMD (head mounted displays). The HMD is usually worn by a single user, and uses a tracking device to measure the head movement and adjust the point of view for the user. Both monoscopic and stereoscopic HMD displays are widely available on the market [105].

Projector-based VRs use multiple monoscopic/stereoscopic projectors to provide an immersive environment covering the entire visual space, and allow multiple participants to experience the virtual environment at the same time. Usually one or two persons can be tracked in the system. Application examples include a virtual classroom or virtual city tour. The CAVE [25] is one of the most famous VR systems, using stereoscopic video projectors to display 3D images on three 3x3 meter screen walls and the floor with participants wearing shutter glasses to see the images.

Multi-Display based VRs use multiple screens such as HDTV or plasma displays to present the images. An advantage of this kind system is lower distortion compared to projector-based VR. The USC Panochamber [81] is an example of Multi-Display based VR systems. However, except for anaglyph techniques, displaying true stereoscopic images for multiple users is a problem for such systems. One reason is that the maximum refresh-rate of the current plasma screens (60Hz without interlace) is lower than the refresh rate required for shutter glasses (96-120Hz without interlace). Other reasons are the relatively small field of view or equivalently, the limits stereo viewing zoon ("sweet spots") of large AS displays that makes it unsuitable for mulit-user virtual environment applications.

Monitor-based VRs usually use monoscopic/stereoscopic desktop monitor with/without head tracking. They are designed for the use by one or two persons, and suitable for personal virtual training or personal games.

Traditional VR systems use computer graphic approaches based on geometric environment models with associated texture mappings on it. Recently, creating the virtual environment from real scenes has attracted increasing interest. Image-based virtual environments that use real pictures to recreate the true reality, are also gaining popularity. Panorama-based VR systems are an example of image-based VR. Panoramic images provide a wide field of view, up 360 degrees, using a simple format. It can be texture mapped to a polygon, cylinder or different extended surfaces, and many researchers have investigated panorama based VR systems in past few decades. Also many different image capturing and rendering techniques for panorama based VR system have been explored. In the following sections, we briefly review these techniques, and list the most famous panorama-based VR systems with their characteristics in Table 2.1.

2.1.1 Omni-directional Image Capturing Devices

Omni-directional image capturing devices usually fall into one of following types:

Still Panoramic Image: In these systems, the display provides users with a single still image with a full field of view. Panoramic images are captured by two approaches. One uses multiple still cameras or one calibrated or un-calibrated moving camera to capture different views of the real scene and then using image mosaic techniques to assemble the pieces into the full image [74][100]. Peleg [121] also discussed the super-resolution panorama mosaicing techniques. Alternative method is to generate the panoramic image directly by equipping the camera with special lens such as a fish-eye lens to project images with a wide field of view on a common surface [3].

Panoramic Video: Several VR systems are designed to capture video sequences with a very wide field of view. Nayar [69], designed catadioptric omni-directional image sensors, and other researchers [3][118] used single video camera with attached convex mirrors or lenses to capture panoramic video. Some limitations of these systems are that the resolution of a single image sensor reduces the quality of the imagery presented to a user, and requiring a special image compensation method. The USC Panochamber [81] uses another image acquiring system, which produces high-resolution panoramic video by employing an array of five video cameras and pentagonal pyramid mirror to view the scene over a combined 360-degrees. Video stitching hardware and software combine the images captured by each video camera appropriately.

Still Stereo Panoramic Images: There are at least two kinds of formats for still image stereo panorama systems. One captures two still panoramic images, one for the left eye and one for the right eye in a directly suitable way for human stereo viewing. Another system captures one panorama and the corresponding depth information. The first system is proposed by Peleg [75]. The system generates left/right stereo panorama pairs by mosaicing strips sampled from images captured by a single camera mounted on a rotation arm and looking outwards. This image capturing structure is usually called a swing panorama structure, and the technique is called the circular-projection multi-perspective panorama method. Based on this system, S.K. Wei *et al*. [114] used a slit camera mounted on a rotation arm instead of sampling the strip from images captured by CCD camera. The second type of systems that captures one panoramic image frame along with the depth information extends the techniques of mono-panoramic cameras by aligning two mono omni-directional cameras on the same vertical axis [36][46] to generate an up/down stereo panorama pair. These data are not useful for human stereo and require a method to calculate disparity to extend depth information or a 3D environment model. An alternate method uses a panoramic camera with a range finder [11]. In addition, several stereo matching, 3D depth estimation and model reconstruction methods [46][11][52][87] have been proposed. Stereo reconstruction from multi-perspective panoramas including concentric panoramas (generated by using slit camera or sampling strips from image taken by cameras mounted on a rotation arm looking at a tangential direction) and swing panorama are also addressed by Szeliski [90].

Stereo Panorama Video: In such systems, users can view any direction with left and right eye video streams. Some systems also extend the omni-directional video sensor techniques with two sensors located one above the other to provide two view points [36]. Another method uses a single camera with a special mirror that produces circular instead of perspective projections. Here, the mirror may limit the FOV of the camera to less than 360 degrees [76]. S. Tzavidas [103] also presented a multi-camera setup that mounts multiple cameras on a rig with fixed geometry. In his system, all cameras look outward from the center of the rig. This system is capable of capturing a 360 degree FOV with horizontal disparity at video rates.

2.1.2 Rendering and Navigation

For monoscopic panoramas, QuickTime VR [18] creates a display of the environment using a panoramic and displays the novel view at any angle around a given point. This method is based on image-based rendering techniques and uses only a two-dimensional plenoptic function (the plenoptic function is usually used in image-based rendering techniques to describe all the radiant energy that can be perceived by the observer at any point in space and time). Later, He [89] *et al* adopted a concentric camera structure and presented a novel 3D plenoptic function that indexes all input image rays naturally in three parameters: radius, rotation angle and vertical elevation. Then, a novel monoscopic view is rendered by combining the appropriate captured rays in an efficient way. Other panoramic rendering methods are found in [38]. Some work has been done on embedding objects into monoscopic panoramic images, so that a viewer can visualize and manipulate the objects directly in the panorama [42][17]. An interactive navigation virtual world through high-resolution cylindrical mono-panoramas based on MPEG-4 and scene description language BIFS has also been presented [27].

Table 2.1 Panorama based VR systems

System	**Content**	**Image Capturing (Generation) Devices**	**Display Type**	**Rendering Method**	**Multi/ Single User**	**Ye-ar**
CAVE [25]	stereo panorama image	computer graphics	stereoscopic projector with shutter glasses	cube projection approximation of a sphere	multi	93
S. Kim Korea [52]	stereo panorama image	two multi-view camera	HMD	IBR/point cloud	single	03
J. Shimamura Japan [87]	stereo panorama video	ominidirectional stereo imaging sensor (twelve cameras and two hexagonal pyramidal mirrors) [114]	CYLINDRA (cylindrical screen with six projectors)	cylindrical 2.5D model with texture mapping	multi	00
Ingo Bauermann Germany [11]	stereo image	one still camera and one laser range finder	anaglyph	interactive image based rendering	single	04
MPEG-4 [27]	still mono panorama	one still camera	none specified	3D cylinder textured with panorama image	multi	04
OmniStereo [75]	still stereo panorama	one still camera (swing panorama)	None specified	polygon approximation	single	01

2.2 Stereoscopic Displays and Techniques

Humans perceive two slightly different views from their two eyes, and the human vision system uses many depth cues to get stereo information and perceive the relative positions between objects in real space. The purpose of stereoscopic imaging techniques is to offer left and right views similar to those obtained in the real world independently to one or multiple viewers. Most current systems simulate and provide the stereo image pair based on human binocular disparity, and some stereoscopic display systems are equipped with head tracking devices to simulate different views according to the head motion and let user experience "looking around". We can classify stereoscopic display systems into one of two categories: Non-autostereoscopic display systems and autostereoscopic display systems [62].

2.2.1 Non-Autostereoscopic Display Systems

Non-Autostereoscopic Display Systems require users to wear special glasses to direct the appropriate images to the correct eye and block unwanted images. Field-sequential techniques and time-parallel techniques are two common methods. In one field-sequential technique, the display alternatively produces the left/right view fields while the user wears synchronized active shutter glasses that alternately block/pass images to the appropriate eyes. Multiple users wearing shutter glasses may also view the stereo images. In time-parallel techniques, the display device presents left and right views at the same time, and uses various optical techniques to direct views to each eye. One example of time-parallel techniques uses anaglyph glasses in which users wear glasses with red/blue color filter to see the anaglyph images. Some disadvantages of this method are the color distortion and the cross talk between left and right images, which users sense as blurred images or double images appearing in the same region. Another technique uses two projectors or displays to show orthogonal polarized right and left images simultaneously. The viewer wears passive polarizing glasses that present the correct right and left image to the eyes.

2.2.2 Flat-Panel Auto-stereoscopic Display Systems

Auto-stereoscopic (AS) displays present users comfortable 3D viewing experiences without the requirement of wearing external goggles or glasses. In this section, we describe the flat-panel AS displays which are used in our research. Other volumetric or AS displays using holographic principle are not discussed here.

Nowadays, most popular and commercial flat-panel AS display are based on parallax-barrier and lenticular techniques [7] [97] [6]:

Parallax-Barrier: A typical parallax-barrier display has an optical grating consisting of an array of open and opaque fine vertical slits located in front of an LCD image panel that displays an interlaced stereo image as shown in Figure 2.1. The vertical slits transmit/block the appropriate image columns and guide the light so that the right and left eye see different images. A variation of this is the back-light or parallax illumination technique used in most 2D/3D switchable parallax barrier displays such as the Sharp AS display. Here, the parallax barrier is made of a series of fine switch-able LCD vertical slits positioned behind the image LCD panel as shown in Figure 2.2. With the display is in the 3D mode, the parallax barrier diverts the light so that the left eye sees the odd columns image on the LCD image panel and the right eye sees the even columns. With the display is in the 2D mode, the viewer sees a fully illuminated flat 2D image. The viewing zone and viewing region are determined by the distance between the image panel to the barrier LCD and the distance between the viewer and the display [65][117]. The advantage of this technique is that user can choose to see 2D content or 3D content from the display by just pushing one button, and this function can allow users read text easily. However, the disadvantage is that the sweet spot is relatively small compared to other AS displays and that the users must keep their head in a specific position. The horizontal resolution of the display in 3D mode is half that of the display in 2D mode.

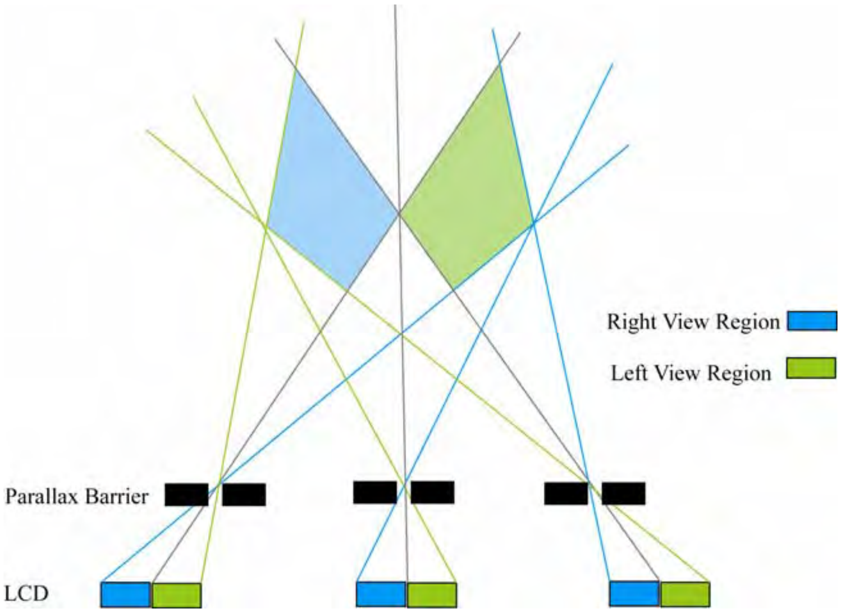


Figure 2.1 Illustration of parallax-barrier AS display.

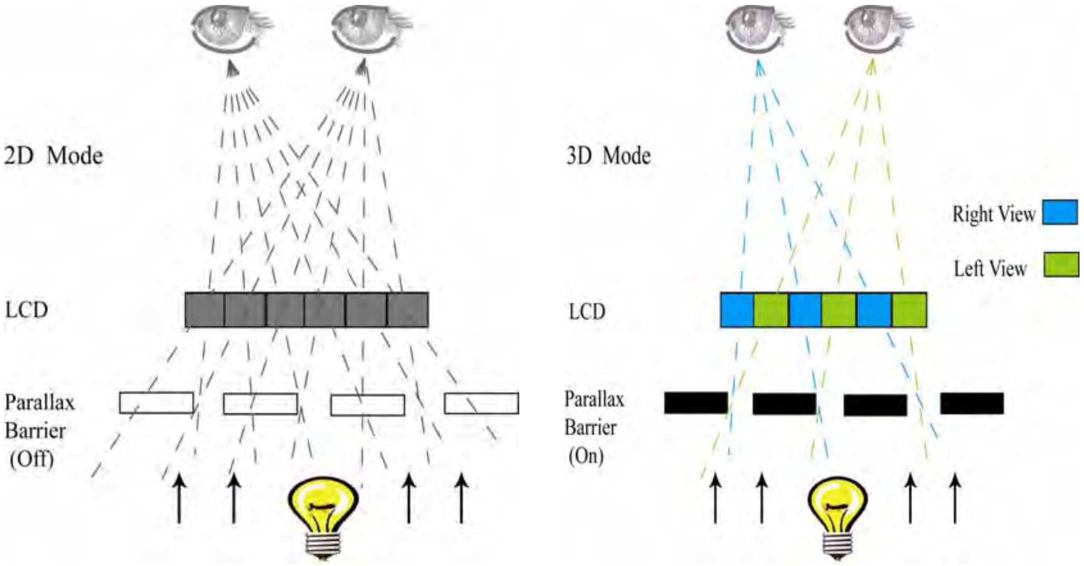


Figure 2.2 Illustration of back-light parallax-barrier AS display.

Lenticular: The lenticular AS display consists a lenticular sheet which is made of series of lenslets, placed on a image LCD panel (Figure 2.3). The lens focuses the light from different sets of strips on the LCD image panel to the two eyes of users. The image quality of the lenticular display is usually superior in brightness than that of the parallax because there is no occlusion as with the parallax barrier systems. In addition, Moire interface pattern effects between periodic patterns in the image are related.

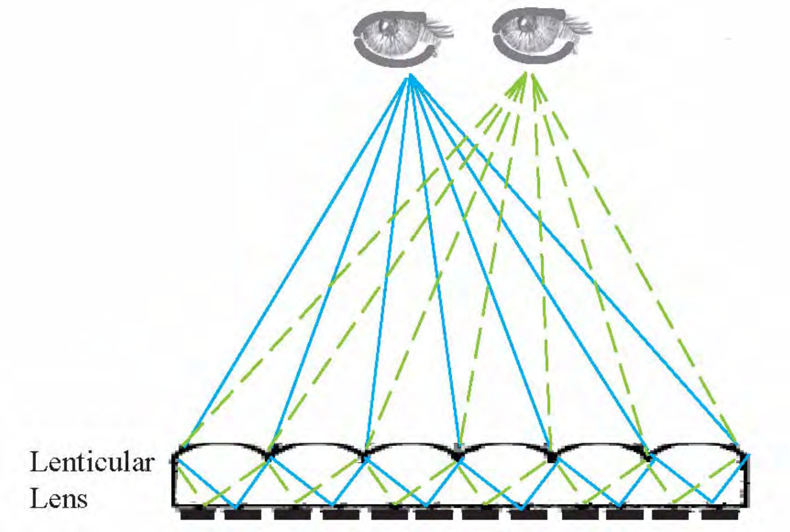


Figure 2.3 Illustration of lenticular AS display

2.2.3 Image Preparation for AS Flat Panel Display

As we describe in the section, an advanced display positions the array of lenticular lenslets at an angle relative to the LCD image panel. The slanted lenticular sheets increase the horizontal resolution because both horizontal and vertical resolutions of the display are used to fill the screen, and remove artifacts such as dark bands in the picture [106]. In addition, it makes the transition between views appear more continuous to let users move around in front of the display, and allow more than one user to watch it. Recently, Philips introduced a seamless 2D/3D switch-able lenticular display, while Seereal launched a prototype system and claiming a larger sweet spot and simultaneous 2D and 3D viewing. The viewing magnification and viewing angle of lenticular AS displays are decided by the lens parameters. Most popular modern AS display techniques are listed in Table 2.2.

Table 2.2 Commercial AS flat panel displays and techniques

AS Display	Techniques	3D Resolution	2D/ 3D	Depth Effect	Application	Cost	Channels	Other Lab
Zebra Imaging	holographic stereograms	full resolution	N	full parallax (h+v)	multi-perspective single viewer	$30,000	n/a	MIT
Sharp	parallax barrier	reduced 1/2 (h) horizontal	Y	image parallax (v)	single view single viewer	$3,000	2	Seereal Elsa
RealD (Stereo-Graphics)	lenticular	reduced 1/3 (h+v)	N	movement and image parallax (v)	single-perspective multi-viewer	$2,000-$20,000	9	Seereal
Dimension Technologies DTI	moving slit parallax barrier	reduced 1/ 2 (h)	Y	image parallax (v)	multi-perspective single viewer	$1,600-$4,000	2	Sanyo Seereal
New-Sight (X3D) (Opticality)	wave length sensitive filter array	n/a	N	movement and image parallax (v)	single-perspective multi-users	n/a	8	Siemens
Philips	lenticular	n/a	Y	movement and image parallax (v)	single-perspective multi-users	$17,500	9	

In this section, we describe the process for displaying images on flat panel AS Displays including the Sharp display and Stereographics display. For either parallax-barrier display or lenticular display, the image LCD panel is composed of sub-pixels. Thus, vertical interlacing of the stereo is the main part of image preparation [13]. For parallax-barrier displays such as the Sharp, the image is created by interlacing the columns from left-eye and right-eye perspective image one by one as shown in Figure 2.4, making the horizontal width double that of one input 2D image. This kind of image is also called a parallax panoramagram.

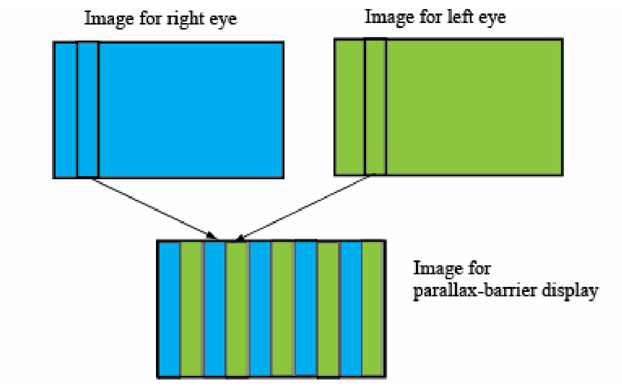


Figure 2.4 Illustration of image generation for parallax barrier AS displays.

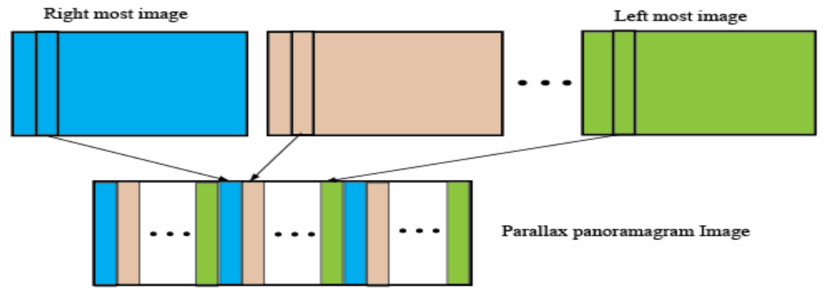


Figure 2.5 Illustration of image generation for lenticular AS displays.

The parallax panoramagram can also be used for vertical lenticular AS diplays. In this case, images are created by interlacing columns from a set of more than two input images as shown in Figure 2.5. However, the interlacing process for the slanted lenticular AS display is much more difficult as shown in Figure 2.6. Berkel [107] developed an interlacing method that improves image quality by choosing only certain pixels useful for creating the best visual effect, and the formula is

$$N_v = \frac{[(k + k_{offset} - 3l \cdot \tan\alpha_L)/N_{sub}]}{N_{sub}} N_{tot} \tag{2.1}$$

where $[]$ is the operator calculates the remainder of a division, N_v is the index of viewing/displaying channel or image, (k,l) are the individual sub-pixel coordinates of the resulting image, k_{offset} is the offset adjustment amount for better image quality. N_{tot} is the total number of input images or channels. α_L is the slant angle of the lenticular sheet and N_{sub} is the number of sub-pixels per lenticule. For the SG202 display, $N_{sub} = 8.5$, $N_{tot} = 9$, $k_{offset} = 0$, $\alpha_L = 20.7$ degrees. Using Eq. (2.1), we can find subpixel (1200, 300) comes from view 8. Thus only 1/9 of input data are used.

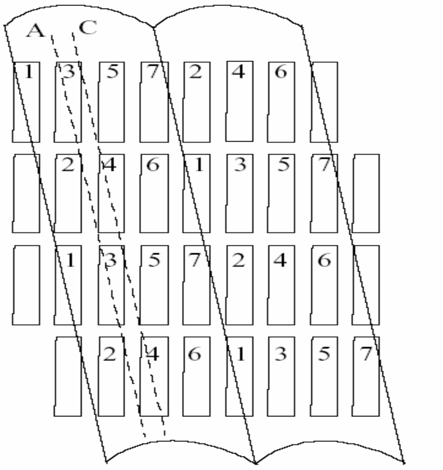


Figure 2.6 Sub-pixel arrangement under two slanted lenticules [107].

A simpler mapping method using lower resolution images with a slight loss of image quality is proposed [26], and so is a hybrid method for slanted barrier display. The formula gives the sub-pixel coordinates of the output image in terms of the sub-pixel coordinate of the input images as:

$$H(x,y,c) = G_m(x_o, y_o, c) \tag{2.2}$$

$$m = 1 + [(3x + c - [y/N_{tot}])/N_{tot}] \tag{2.3}$$

$$x_o = (3x + c - [y/3]) / N_{tot} \tag{2.4}$$

$$y_o = y/3 \tag{2.5}$$

Where [] is the operator calculates the remainder of a division, $H(x,y,c)$ is the sub-pixel coordinates of the output image, $G_m(x_o, y_o, c)$ is the sub-pixel coordinates of image m, where c is the color index ($c = 0$ for red, $c = 1$ for green, $c = 1$ for blue). N_{tot} is the total number of input image.

Some SDK (software development kits) for interlacing input images for several commercial parallax-barrier and lenticular AS display are provided by DDD and RealD [95][96].

2.3 Human Depth Perception and Disparity

The purpose of this section is to review how human perceive depth in order to generate images for a comfortable stereo viewing experience. Humans use many different techniques to perceive the depth of objects. These ways are called depth cues in psychology and can be classified into two groups: monocular cues and binocular cues [71].

2.3.1 Monocular Cues

Human can sense depth from planar images by using monocular cues as follows. Most of them are psychophysical cues.

Relative Size: Humans know the relative depth between positions because the image object size changes when object moves.

Linear Perspective: This depth cue is usually used in painting. The principle is that the parallel lines converge to a single point at infinite distance from the observer.

Areal Perspective: The depth sense of a far away object (such as a mountain) is perceived to be enhanced if the object (mountain) looks blurred. The reason for this is that the light is scattering by the particles in the air.

Occlusion: Objects generally located in the foreground have outlines that appear relatively continuous and obscure than objects located in the background.

Shades and Shadows: An object's position can be inferred even with little knowledge of the location of the light source and shape of the object.

Texture Gradient: When we look at a uniformly textured object such as a gravel road, the texture of the object region in distance has a finer scale.

Motion parallax: If two same objects at same distance move with different speeds, the viewer feels the faster object is closer.

Accommodation: When the viewer's eyes adjust the focal length to see an object within a viewing distance of two meters, a muscular tension of the eyes exists. This effect is called an accommodation depth cue. However, this cue is only works when it is combined with other binocular depth cues.

Although these depth cues are distinct effects [59] in human perception, they are strongly coupling in some ways and the interpreted results of these depth cues and influenced by human prior assumptions [16]. Certain depth cues have strong interactions and influence each other [66].

2.3.2 Binocular Cues

Convergence: When the left and right eyes rotate slightly to focus on a common point, the resulting muscular tension of the eyes gives a depth perception cue called vergence. The vergence (convergence) angle 2α is the angle formed by two viewing axes, and is approximated when α is small by dividing the distance between two eyes, B , by the distance between the eye and object a , as given by

$$2\alpha = B / a \,. \tag{2.6}$$

By differentiating Eq. (2.6), we get

$$d\alpha / da = B / (2a^2) \tag{2.7}$$

From this equation, note that fixing the distance B between two eyes at the typical approximately value of 6.5 cm, the change of the vergence angle α is approximate one degree when a varies from ∞ to 3.58 m and when a varies from 25 to 23.4 cm [71].

Binocular Disparity: Referring to Figure 2.7, when an observer looks at a point F , the viewing axes of the two eyes intersect at that point F . The point F is called a point of fixation, and its images are on the centers of left and right retinas. However, images of a point which is not the fixation point are generally not at the same corresponding positions on the two retinas. The difference between image positions of a point on left/right retina is called binocular disparity. The O_l and O_r are centers of left and right eyes looking at the fixation point F . The O_l , O_r and F define a circle called the Vieth-Müller circle (fixation point circle). The images of F on left/right retina are marked as f_l , f_r . The images of two other points P and Q on left/right retina are denoted as p_l , p_r and q_l , q_r .

Then, the binocular disparity η for point Q is defined as the difference of the two angles formed by the ray to the two eyes and the optical axes of the two eyes as

$$\eta = \theta_2 - \theta_1 = 2(\beta - \alpha) \tag{2.8}$$

By defining the distance between one eye to the point F and Q as a_F and a_Q, respectively, the binocular disparity can be represented as

$$\eta \approx (B / a_Q - B / a_F) \approx B(a_F - a_Q) / a_F^2 \tag{2.9}$$

As we can see, the binocular disparity of a point P on the Vieth-Müller circle is zero. In addition, humans can perceive binocular disparity within the angular range

$$-0.5^\circ < \eta < 0.5^\circ \tag{2.10}$$

according to experiments [75]. A method for calculating stereoscopic camera parameters to produce stereo images that are viewable without discomfort between these depth cues is proposed in [45], which clearly separates the image capture camera/scene space from the image viewing viewer/display space and provides a transformation between these two spaces. This method is implemented as an API extension of OpenGL.

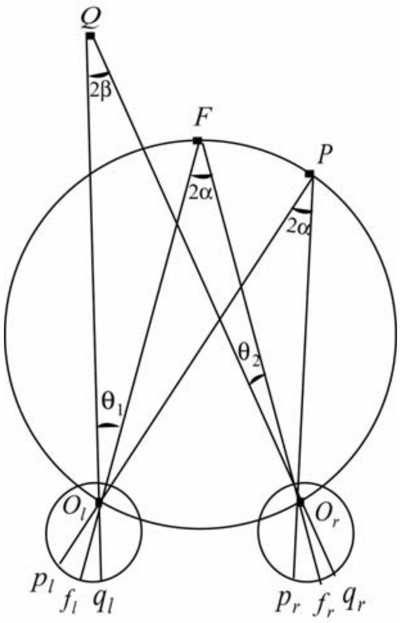


Figure 2.7 Illustration of human binocular disparity.

2.4 Other Related Work

Adjusting the perceived depth or disparity of stereoscopic images within an acceptable range to provide a comfortable stereo viewing experience has become an interesting research topic because humans can only perceive binocular disparity within a small dynamic range, [71], as stated earlier in Eq. (2.10). We generally categorize most previous work into the following: (1) methods applied to stereoscopic images with a known 3D model, such as computer graphic generated images [45][41]; and (2) methods applied to stereo images captured by cameras [75] from the real world, such as stereo photography, videos, and panoramas. The first set of methods control the virtual camera parameters (i.e. the distance between two cameras) and provide a different mapping/rendering method to map the virtual scene depth to the target display. The second method often uses view synthesis, image-based rendering or similar techniques.

Peleg *et al* [75] proposed a method to create a stereo panorama pair by mosaicing sampled strips from images captured with a swing panorama structure [3] having one single-view camera mounted on a rotation arm with a rotation axis behind it. He also described a method to adjust the horizontal disparity by computing horizontal disparities between left/right panoramas using simple correlation windows along each column and changing the strip separation by using a larger distance between strips for distant objects and a close distance for nearby objects. However, this method modifies not only the disparity of certain regions but also the columns that include these regions. We review these ideas in detail in Chapter 5.

In this paper, our goal is to develop stereo panorama image capture and a panoramic virtual environment system to improve the sense of immersion. The system may be equipped with an AS or other stereo display and is suitable for virtual tours of outdoor or indoor sites such as museums or galleries. This is feasible because several new types of 3D autostereoscopic (AS) visual displays (e.g. desktop and laptop, large screen versions) that produce a 3D visual sensation to one or more observers without the use of traditional clumsy glasses have been developed and commercialized.

Our stereo panorama creation method is based on Peleg *et al* [75]. In this work, we address the global image registration problem, and propose a novel stitching algorithm to balance exposure and remove moving objects automatically. We also consider the vertical disparity problems that occur in the stereo panorama causing eyestrain and problems in image fusion (particularly for close objects) for most participants. Thus, we present expressions for the horizontal and vertical disparity for a complete understanding of the process, measure vertical and horizontal disparity simultaneously between panoramas with optical flow techniques rather than Peleg's correlation window method, and develop a vertical disparity control algorithm to improve the rendering of stereo panoramas.

Since large horizontal disparities of far away objects and smaller disparities of closer objects are needed for stereo perception, we develop a human interactive object-based tool to adjust the disparity using similar view synthesis techniques. The method changes the disparities within local 2D selected regions rather than for an entire column as in Peleg's method [75]. In the future, our complete interactive disparity-adjusting tool will allow the user to select the object from manual input using a cursor or by defining an area with a certain distance range, with the ability to observe the results immediately on an AS or other displays.

We also present adaptive vertical and horizontal disparity control algorithms for rendering the stereo panoramas in different viewing directions and for rendering on AS displays in a manner similar to Quick-Time VR [18].

Chapter 3 System Model

We describe the stereo panoramic capturing system in this chapter, and provide a detailed geometrical model. With this model, we can calculate the strip sampling position based on known geometry and camera parameters, and describe how various physical parameters affect the perceived depth and image quality.

3.1 System Overview

Our panoramic virtual environment concept, as shown in Figure 3.1, is based on a swing panorama camera structure: a single-view camera is mounted on a rotation arm with a rotation axis behind it. Sampled strips from the captured images are combined to create the stereo panorama and render it on an AS display.

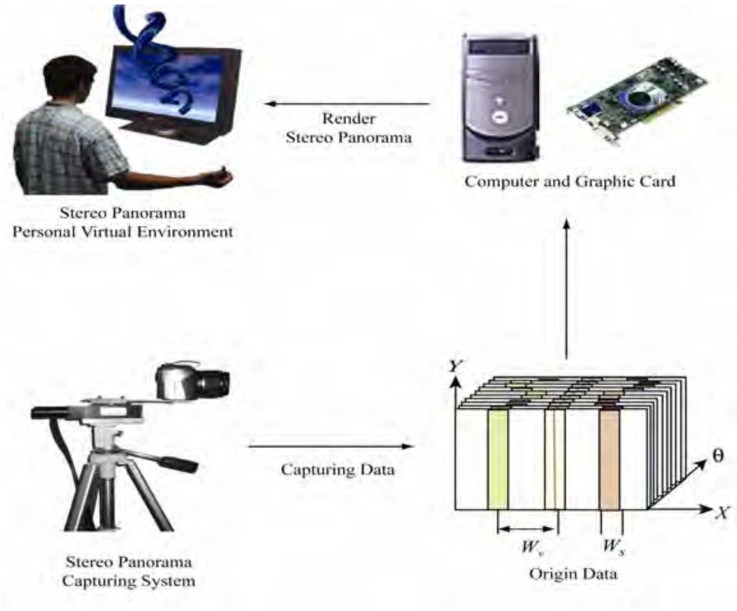


Figure 3.1 System overview.

3.2 Image Capturing Model

Stereo panoramas can be generated by mosaicing images from a rotating camera. The concentric panorama and swing panorama are two common methods for capturing the data needed for stereo panorama mosaicing. The former captures images with a camera mounted on a rotation bar whose optical axis is aligned with the tangent direction of the rotation [89]; while the latter, the swing panorama used in this proposal, captures images with a camera looking outward that is mounted on a rotation bar whose optical axis is perpendicular to the tangent direction of the rotation (see Figure 3.2).

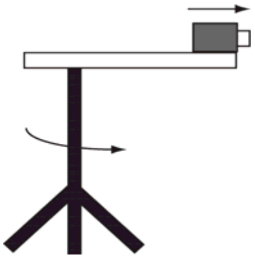


Figure 3.2 Swing panorama capture system.

We construct a stereo panorama pair (left/right) by stitching strips, each with width W_s at location W_v, taken from the left/right side of the captured image sequences as shown in Figure 3.3. The image capture system collects a highly overlapping set of views (typically one hundred or more) that includes all views of a 360 degree panorama. The views are shown in Figure 3.3 as a stack of images indexed by camera notation angle θ. Strips representing portions of left and right panoramic views are combined to create the stereo panorama.

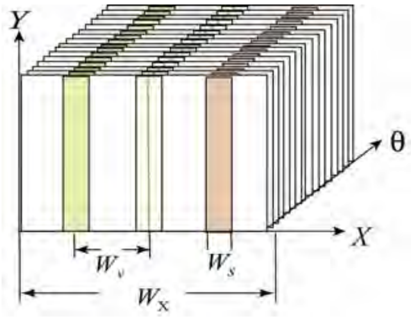


Figure 3.3 Illustration of stereo panorama generation.

The complete set of all views captured are pixel values defined on the (X, Y, θ) coordinate system as shown in Figure 3.3. We denote by $(x_{r1}, y_{r1}, \theta_{r1})$ the location of a particular (right view) pixel included in a strip that is part of the complete right panorama. Note that this same point generally appears in several of the 2D image frames taken at various θ angles. Similar variables with subscript l are used for pixels in the left panorama.

3.3 Geometry Model

Figure 3.4 illustrates the stereo panorama capture geometry model; Figure 3.5 is a detailed view of the part of Figure 3.4. The width and position of sampled strips (see Figure 3.3.) are denoted as W_s and W_v. W_X is the width of the image, and f is the focal length of the camera. The camera center O_c, which rotates with respect to the rotation center O_r, forms a circle with a dotted line shown at the left in Figure 3.4. having radius D_{oc}. Here O_{cl} and O_{cr} are the camera centers of the left and right views. Objects captured for stereo visualization are inside the region of interest (ROI), shown as the shaded region at the left of Figure 3.4., which is bounded by the furthest circle with radius D_{max} and the nearest circle with radius D_{min}. The point of fixation P_{fox} forms a circle of radius D_f. Here, we assume that

$$D_f = \tfrac{1}{2}(D_{max} + D_{min}) . \tag{3.1}$$

$\Delta\theta$ is the rotation angle between two successive shots; $\Delta\theta_{rl}$ is the angle between the left/right shots, and we define $\theta_w = (\Delta\theta_{rl})/2$. The lighter shaded dashed-lines in Figure 3.5 are the left/right viewing direction of the strips that compose the stereo panorama.

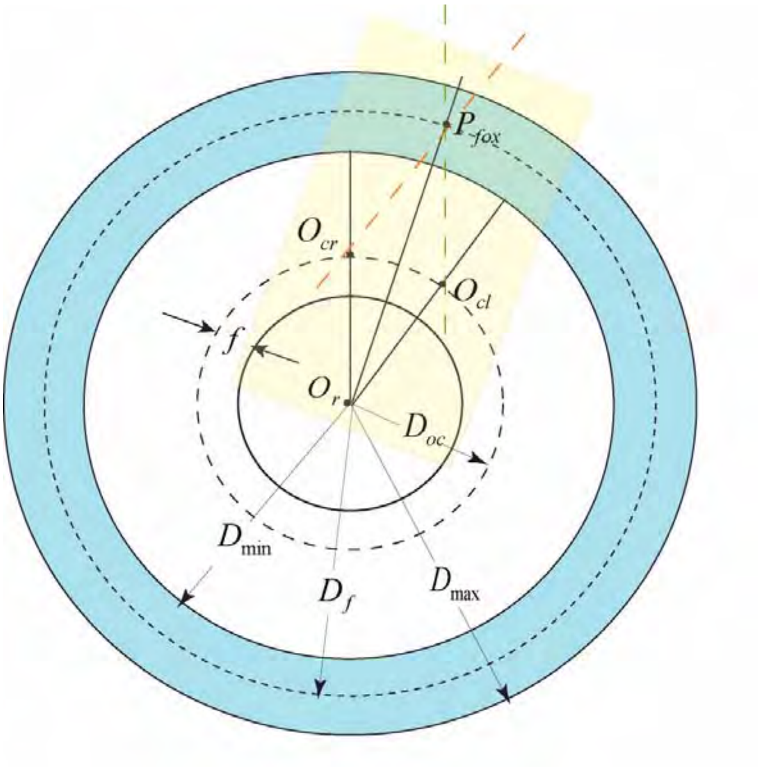


Figure 3.4 Geometry model of stereo panorama capturing system.

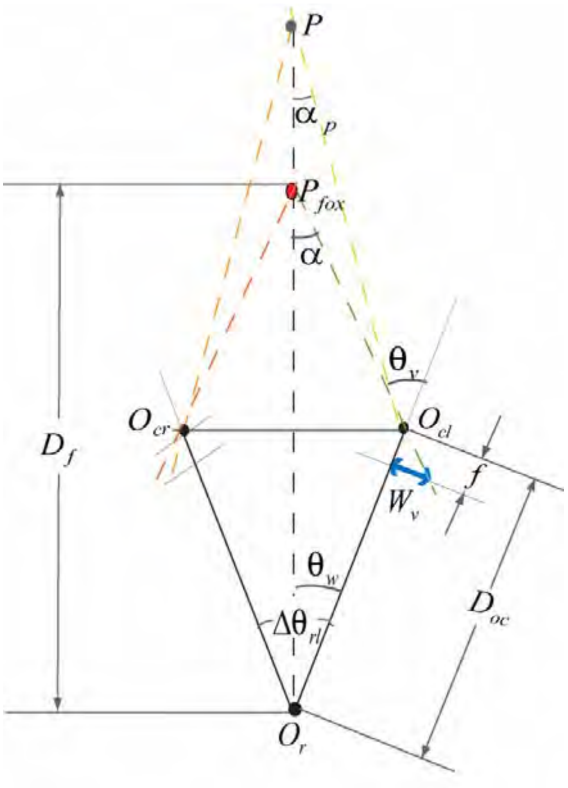


Figure 3.5 Geometry model of stereo panorama capturing system.

Considering the triangle $\Delta O_r O_{cl} P_{fox}$, we get

$$\frac{D_f}{D_{oc}} \approx \frac{\sin(\pi - \theta_v)}{\sin(\alpha)} \approx \frac{\theta_w + \alpha}{\alpha}. \tag{3.2}$$

Using simple geometry, we have

$$W_v = f\tan(\theta_v) \approx f\tan(\theta_w + \alpha) \approx f(\frac{D_f}{D_f - D_{oc}})(\theta_w) \tag{3.3}$$

Assume a strip at location W_v' in the image captured at the camera position $\theta_{w'}$ is corresponding to the strip sampled at location W_v in the image captured at camera position θ_w. Here, $\theta_{w'}$ and θ_w are the camera position of two successive shots ($\theta_{w'} - \theta_w = \Delta\theta$). We can get the sampled strip width W_s as

$$W_s = W_v' - W_v \approx f(\frac{D_f}{D_f - D_{oc}})(\theta_{w'} - \theta_w) \approx f(\frac{D_f}{D_f - D_{oc}})\Delta\theta . \tag{3.4}$$

From Eq. (3.2) and (3.3), when θ_v is small we obtain

$$\alpha \approx \frac{D_{oc}\theta_w}{D_f - D_{oc}} \tag{3.5}$$

Because $\Delta\theta$ is the rotation angle increment, we set $\theta_w = K \cdot \Delta\theta$ and K is an integer. From Figure 3.5, the vergence angle for the fixation point P_{fox} is 2α, the vergence angle for point P far away is $2\alpha_p$, and the range of the binocular disparities is simplified as $\left|2\alpha_p - 2\alpha\right|$. Because humans can only perceive binocular disparity within a limited range as we mentioned in Eq.(2.10), we have

$$\left|2\alpha_p - 2\alpha\right| < 0.5^\circ . \tag{3.6}$$

By setting the point at most far away plane with zero disparity ($\alpha_p \approx 0$), we have

$$\alpha = K\left(\frac{D_{oc} \cdot \Delta\theta}{D_f - D_{oc}}\right) < 0.0044 \tag{3.7}$$

where 0.0044 (rad) equals to 0.25°, and

$$K < 0.0044\left(\frac{D_f - D_{oc}}{D_{oc} \cdot \Delta\theta}\right). \tag{3.8}$$

From simple geometry, we obtain W_v and W_s as Eq. (3.9) and Eq. (3.10) with constraints given by Eq. (3.11).

$$W_s = f(\frac{D_f}{D_f - D_{oc}})\Delta\theta \tag{3.9}$$

$$W_v = Kf\,(\frac{D_f}{D_{f\,-}D_{oc}})(\Delta\theta) = KW_s \tag{3.10}$$

$$K < 0.0044\left(\frac{D_f - D_{oc}}{D_{oc}\cdot\Delta\theta}\right) \tag{3.11}$$

In addition, from Figure 3.3, the whole sampled strip must be inside a simple image, requiring:

$$W_v + (W_s\,/\,2) < (W_{\mathrm{x}}\,/\,2) \tag{3.12}$$

$$W_v - (W_s\,/\,2) > 0 \tag{3.13}$$

The horizontal disparity of a point at distance D_p, denoted by Δs_x, is simply derived from Figure 3.4 as

$$\Delta s_x \approx f\,(\frac{D_p}{D_p - D_{oc}})\Delta\theta_{rl} \tag{3.14}$$

Chapter 4 Stereo Panorama Generation

Stereo panoramas are generated by stitching strips taken from the image sequence. In this chapter, we describe the two key processes of stereo panorama generation: image registration and stitching.

4.1 Global Feature Based Image Registration

Most previous work [48] for capturing panoramas requires specific camera systems with extremely precise and known rotational motion. To build such devices is expensive. Instead, we use image analysis and registration methods that do not need a calibrated camera. This method automatically maps all the captured images into a reference frame to make a high-resolution image mosaic. One method of doing this is to use local registration between neighboring pairs of images. However, in this procedure, the complete panoramic mosaic may have large overall errors due to accumulated mis-registration between the pairs. In addition, this may give a significant mismatch between the first and last frames that make up the panorama. In this section, we describe and apply a feature-based global image registration method based on [84][49] to overcome this problem. Our approach is shown in Figure 4.1 and is described as a sequence of steps. We also describe the computational complexity of each step and summarize the complexity in section 4.3.

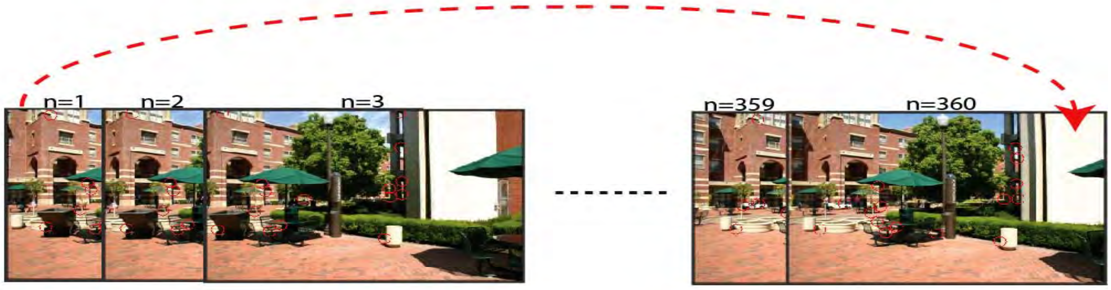


Figure 4.1 Illustration of global image registration based on feature selection.

Step 1: For each image, we select and track features among neighboring frames using Lucas-Kanade [56] and Thomasi's [88] algorithms (Kanade-Lucas-Thomasi feature tracker). We first pre-smooth images with a 7x7 pixel Gaussian filter to avoid trivial features, and use the previous and next 6 frames as neighboring frames.

The Lucas-Kanade algorithm is an optical flow algorithm, whose goal is to estimate the motion of pixels between two frames. It tracks features with similar brightness between two image frames which are taken at time t and $t+\delta t$. Assuming color constancy, and using Taylor series expansion, a pixel at location (x, y, t) with intensity $I(x, y, t)$ will have moved by δx, δy, δz, and δt between the two frames and the image constraint equation is given by

$$V_x I_x(x, y, t) + V_y I_y(x, y, t) + I_t(x, y, t) = 0. \qquad (4.1)$$

Note that $I_x(x, y, t)$, $I_y(x, y, t)$, $I_t(x, y, t)$ are the derivatives of intensity in x and y directions and in time; V_x and V_y are the velocity of a point in x and y directions. Assuming small motion that the flow (V_x, V_y, V_z) is constant in a small window $n \times n$, the optical flow problem reduces to the Lucas-Kanade equations

$$\begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \\ \vdots & \vdots \\ I_{xn} & I_{yn} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = - \begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \\ I_{tn} \end{bmatrix} \quad \text{or} \quad A\vec{v} = -b \qquad (4.2)$$

This equation can be solved by finding the minimum least squares solution of

$$A^T A\vec{v} = -A^T b. \qquad (4.3)$$

Based on Lucas-Kanade [56] algorithm, the algorithm defines a good feature as one that can be tracked well, or a feature that is similar in adjacent frames. The dissimilarity is calculated as the sum of square difference (SSD) of pixel values between a new patch (obtained by warping a path in original frame with affine transformation model) and the corresponding patch in new frame.

Thomasi's algorithm first finds the suitable corners in a frame using a Harris corner detector

$$\mathrm{Z} = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \tag{4.4}$$

, where I_x, I_y are the derivative of the intensity in x and y directions. Assumes constant intensity and small camera motion, the algorithm calculates the minimum eigen $\lambda_{\min}$ value of matrix Z for each image. If the pixel has minimum eigen value larger than certain threshold ($\lambda_{\min} > \tau$), it records the pixel in a list and sort the list by eigen value. If this pixel is already in the neighbor of a pixel on the list, then delete this pixel from the list.

The iterative Lucas-Kanade algorithm first estimates the velocity at each pixel by solving Lucas-Kanade equations, and continues warping the original template toward a new template using the estimated flow field until it converges. Suppose the image size is s and the motion velocity is v, the computational efficiency is $O(vs)$.

Step 2: We then calculate the fundamental matrices (is a 3×3 matrix of rank 2 which related corresponding points in stereo pair) between frames with RANSAC (Random Sample Consensus) algorithm [33] based on selected features.

In epipolar geometry, homogeneous coordinates $p_1 = (x_1, y_1, a_h)^{\mathrm{T}}$ $p_2 = (x_2, y_2, a_h)^{\mathrm{T}}$, are corresponding points in image pair, and Fp_1 describe a line (epipolar line) on which the corresponding point p_2 must lie. The relationship can also be described as

$$p_2^T \mathbf{F} p_1 = 0 . \qquad (4.5)$$

F is the fundamental matrix and can be computed by solving above homogenous linear equations with multiple point matches.

RANSAC is a method to estimate parameters of a mathematical model from a set of observed data which contains outliers. A basic assumption is that the data consists of "inliers", i. e., data points which can be explained by some set of model parameters, and "outliers" which are data points that do not fit the model. RANSAC also assumes that, given a set of inliers, there exists a procedure which can estimate the parameters of a model that optimally explains or fits this data. The generic RANSAC algorithm works as follows:

```
input:
   data - a set of observed data points
   model - a model that can be fitted to data points
   n - the minimum number of data values required to fit the model
   k - the maximum number of iterations allowed in the algorithm
   t - a threshold value for determining when a data point fits a model
   d - the number of close data values required to assert that a model fits well to data
output:
   bestfit - model parameters which best fit the data (or nil if no good model is found)

iterations := 0
bestfit := nil
besterr := infinity
while iterations < k
            maybeinliers := n randomly selected values from data
            maybemodel := model parameters fitted to maybeinliers
            alsoinliers := empty set
for every point in data not in maybeinliers
      if point fits maybemodel with an error smaller than t, add point to alsoinliers

if the number of elements in alsoinliers is > d
      (this implies that we may have found a good model now test
      how good it is)
      bettermodel := model parameters fitted to all points in maybeinliers and alsoinliers
      thiserr := a measure of how well model fits these points
      if thiserr < besterr
          bestfit := bettermodel
          besterr := thiserr
  increment iterations
  return bestfit
```

For the computational efficiency, there is no upper bound on the time it takes to compute these parameters. If an upper time bound is used, the solution obtained may not be the most optimal one.

Step 3: We then create a weighted-graph for global registration as shown Figure 4.2. Each node represents one frame. An arc is created between two nodes if the nodes are neighbors. The weight of an arc is the cost of the transformation between two frames. The cost functions used here are: (1) local correlation error $I(M_{BA}P_A^i) - I(P_B^i)$; and (2) position accuracy $M_{BA}P_A^i - P_B^i$. Here, $I()$ is the intensity field; P_A^i represents the position of the ith feature point in frame *A*; M_{BA} is the fundamental matrix calculated based on frames *A* and *B*.

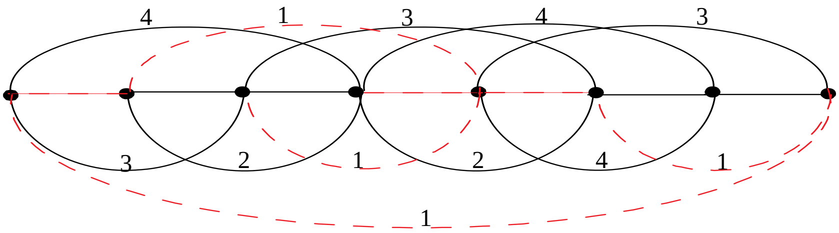
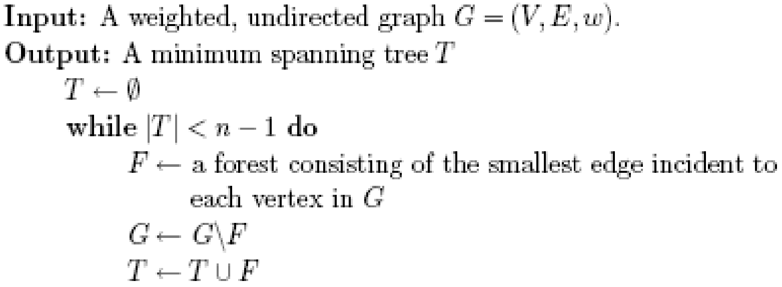


Figure 4.2 Graph showing the cost of transformation (weight linking) between frames (nodes). The dashed links represent the optimal path.

Step 4: We search the sets of links as in Figure 4.2 that provides the overall minimum cost mapping of all nodes using the minimum spanning tree algorithm (MST) [61]. A minimum spanning tree of a connected, undirected, weighted graph is a sub-graph which is a tree and connects all the vertices together with weight less than or equal to the weight of every other spanning. Several algorithms can solve minimum spanning tree problems with different computational complexity. Assume that m is the number of edges and n is the number of vertices. In our case $m = 3n$, and we chose Bor˚uvka’s algorithm for less computational complexity. The algorithm is as follows



```
Input: A weighted, undirected graph $G = (V, E, w)$.
Output: A minimum spanning tree $T$
    $T \leftarrow \emptyset$
    while $|T| < n - 1$ do
        $F \leftarrow$ a forest consisting of the smallest edge incident to
            each vertex in $G$
        $G \leftarrow G \backslash F$
        $T \leftarrow T \cup F$
```

The computational complexity is $O(m \log n)$. The resulting minimum set is called the optimum path; the dashed links in Figure 4.2 are the optimal (minimum cost) path.

Step 5: We warp each image toward the reference frame based on optimal path. The computational complexity of this step is only constrained by the size of image (s): $O(s)$.

4.2 Mixture Gaussian Density Image Stitching Procedure

In assembling the panoramic image from strips, there are several sources of error:

- Illumination varies from frame to frame
- Moving people or objects (waving flags, fluttering leaves) exist in the dataset, leading to differences in the left/right images that are not due to stereo disparity.

Traditional panorama stitching algorithms require a good object removing algorithm and the manual selection of ground truth images that exclude moving objects. Szeliski [104] proposed a graph method to eliminate the ghosts, but it still leaves some moving objects (people) in the panorama. In this section, we consider the strips taken from successive frames as in Figure 4.3, and develop a novel stitching algorithm based on mixture-Gaussian distributions to overcome dynamic illumination effects and remove moving objects simultaneously without manually selecting ground truth images. In Figure 4.3, strips with the same color represent the corresponding strips in different frames and the current sampled/processing strip labeled with a dotted boundary.

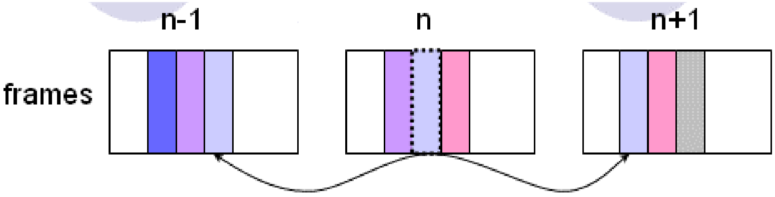


Figure 4.3 Corresponding strips in neighboring frames.

There are six steps in the stitching algorithm:

Step 1: Get strips imaging the same part of the scene as shown in frames $n-2$, $n-1$, n, $n+1$, $n+2$ of Figure 4.4.

Step 2: At each corresponding pixel i in strip k from $n-2$ to $n+2$, convert the (R,G,B) color components to (Y,I,Q) representation. Here $Y(i,k)$ is the luminance component at a given pixel.

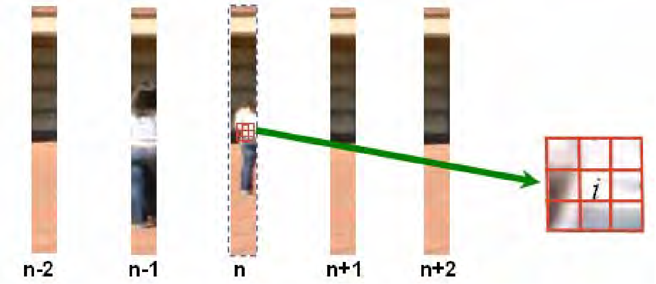


Figure 4.4 Sampled strips in neighboring frames.

Step 3: Compute the local variance $V(i,k)$ for each pixel $Y(i,k)$ using

$$V(i,k)=\frac{1}{9}\sum_{3\times 3\,window}[Y(i,k)-Y(neighbors,k)]^2 \qquad (4.6)$$

, where $Y(neighbors,k)$ is the set of nearest neighbor pixels surrounding $Y(i,k)$.

Step 4: Calculate a Gaussian probability density $P(i,k)$ with mean $Y(i,k)$ and variance $V(i,k)$ for all k frames.

Step 5: Plot the luminance probability $\tilde{P}_c(i)=\sum_{k=n-2}^{k=n+2} w_k P(i,k)$, where w_k is a weighting factor for frame k and $\sum_{k=n-2}^{k=n+2} w_k = 1$. We use $(w_{n-2}, w_{n-1}, w, w_{n+1}, w_{n+2}) = (0.1, 0.2, 0.4, 0.2, 0.1)$ in our experiment.

Step 6: Select an output luminance level $P_{out}(i)$ for pixel i as that value of $\tilde{P}_c(i)$ having the maximum probability effectively (a maximum likelihood estimate) as in Figure 4.5. Note that when the pixel belongs to an unexpected object, its gray level value (color) is very different from the maximum peak. However, if there is no unexpected object, their gray level values are very close to each other.

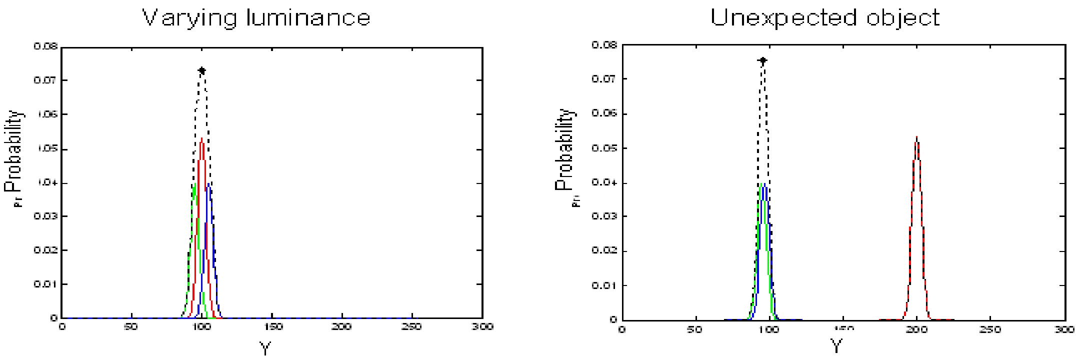
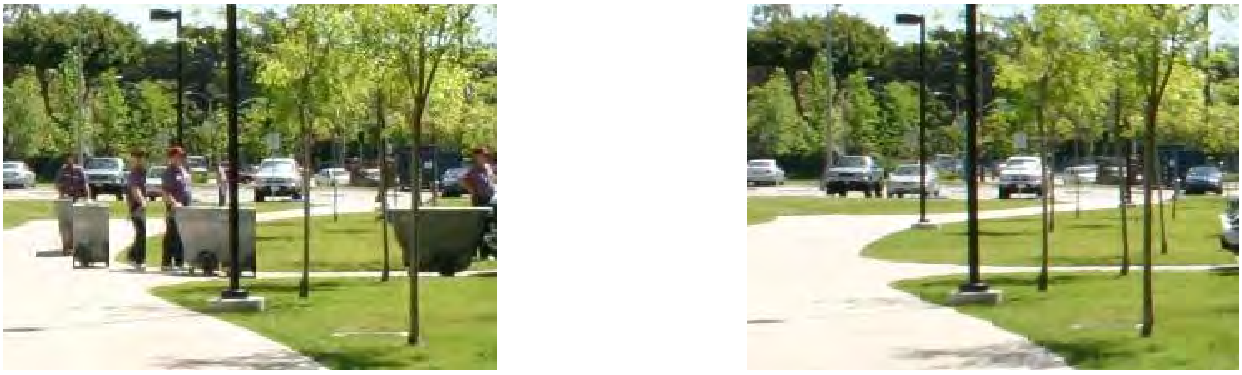


Figure 4.5 Example of Mixture-Gaussian Maximum-likelihood selection.

The mixture-Gaussian stitching is demonstrated in Figure 4.6. We can see that several ghosts appear in the left image created without our mixture Gaussian algorithm, while the right image is clean in which we apply our Gaussian stitching algorithm.



without Gaussian stitching algorithm

with Gaussian stitching

Figure 4.6 Results of the stitching algorithm.

4.3 Computational Complexity of Stereo Panorama Generation

The rotation stage speed and the camera features decide the image capturing speed, and the computational efficiency for generating a stereo panorama is mainly constrained by image registration and image stitching processes.

4.3.1 Global Feature Based Image Registration Process

The global feature based image registration process consists three different parts: Lucas-Kanade algorithm feature selecting and tracking algorithm, RANSAC (Random Sample Consensus) algorithm for selecting useful features, transformation error calculation, and minimum spanning tree algorithm (MST) for selecting an optimal registration. The computational complexity of each algorithm is discussed in section 4.1.

4.3.2 Mixture Gaussian Density Image Stitching

Our mixture Gaussian density image stitching has three main steps: (1) calculate the mean and variance for each pixel and create Gaussian density function. 2) create mixture Gaussian density function for each pixel based the Gaussian density of each pixel in neighboring sampled strips. 3) select the color/gray value which has maximum probably to represent the color/gray value of that pixel. The computational complexity of the first step is mainly decided by the size of the window (ω) which defined the neighbors of a pixel. The computational complexity of the second step is constrained by the number of neighboring frames used for the corresponding strips (m). The computational complexity of the last step is decided by the number of pixels in the set, and the set is defined by the minimum and maximum gray value of peaks or intersections and is less than 255. In conclusion of the three steps above, the total computational complexity of mixture Gaussian image stitching is $O(\omega mn)$.

Chapter 5 Disparity Measurement and Adjusting

One of the main physical depth cues for humans is binocular disparity, in which each eye views an object from a slightly different position and the fusion of the two slightly different images in the brain creates the perception of depth. The distance between the projections on the left and right eye is called horizontal disparity. If the projections are on the same side as the respective eyes, it is called positive disparity. Thus, positive disparity is obtained for objects that lie behind the point-of-fixation circle. Similarly, when we move our head up and down, we will experience vertical disparity. However, if the stereo pair (left/right images) is created with projection planes that are not identical (see Figure 5.1), there may be a distortion between the images and vertical disparity is introduced. These phenomena cause viewing discomfort and eyestrain. We have observed vertical disparity in our stereo panorama pairs, so we discuss horizontal and vertical disparity in stereo panoramas, and algorithms to adjust it within a comfortable viewing range. Note that vertical disparity becomes very small when the object distance is very large compared to the inter-ocular distance.

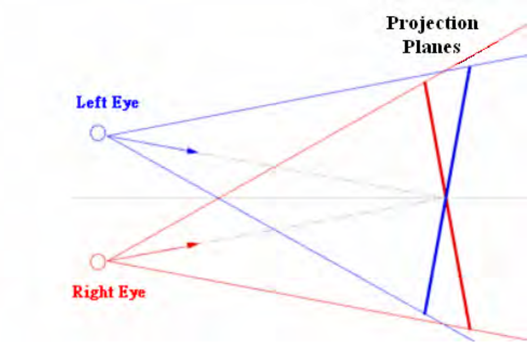


Figure 5.1 Vertical disparity introduced by different projection planes.

5.1 Vertical Disparity Measurement

Vertical disparity is the vertical distance between the projected positions of a 3D scene point in the left/right view, and is obtained with reference to Figure 3.4 and Figure 5.1 [57]. For perspective projection, the vertical position y of a 3D point at vertical coordinate Y and distance Z along the camera axis is $y = f(Y/Z)$. Thus, a 3D point at height Y located at distance D_p from the rotation center O_r, is mapped to the 2D vertical position $y_r = f(Y/Z_r)$ in the right view and $y_l = f(Y/Z_l)$ in the left. Referring to Figure 5.2, we have

$$Z_r = D_p - D_{oc}\cos\theta_r \tag{5.1}$$

where

$$\frac{D_p}{\sin(180^o - \phi_r)} = \frac{D_{oc}}{\sin(\phi_r - \theta_r)} \tag{5.2}$$

Similarly,

$$Z_l = D_p - D_{oc}\cos\theta_l \tag{5.3}$$

where

$$\frac{D_p}{\sin(180^\circ - \phi_l)} = \frac{D_{oc}}{\sin(\phi_l - \theta_l)} \tag{5.4}$$

Thus, the vertical disparity $\Delta y = y_r - y_l = f(\frac{1}{Z_r} - \frac{1}{Z_l})Y$. For the ideal symmetrically sampled left/right strip, the angle $\phi_r = -\phi_l$ so that $\theta_r = -\theta_l$. Thus, we get zero vertical disparity ($\Delta y = 0$), meaning that the epipolar lines (described in page 37) are horizontal.

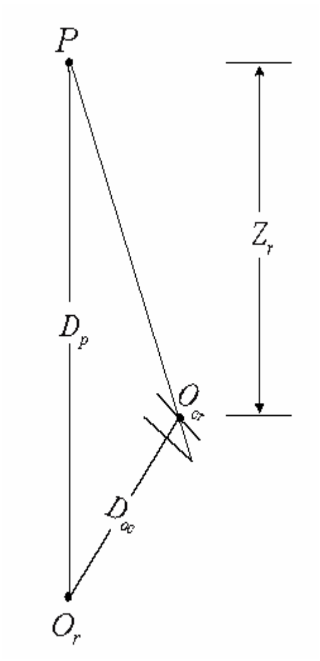


Figure 5.2 Geometry illustration of the distance along camera optical axis.

However, the strips for the left/right panoramas are taken from different images, and small registration errors still exist between them. The projection planes for left/right strips are not the same, and the epipolar lines cannot remain horizontal. Consequently, vertical positions of projected corresponding points are not equal and vertical disparity appears in our generated stereo panorama. (If there is no small registration error, we can project images into same plan and the vertical position of the projected corresponding points will remain the same.) Figure 5.3 shows this phenomenon.

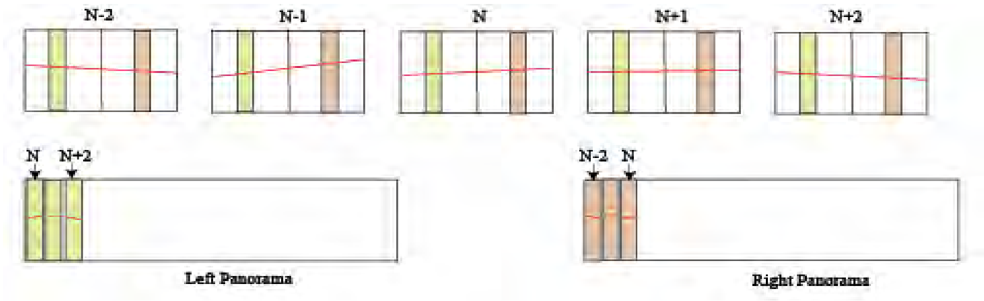


Figure 5.3 Registration errors induce vertical disparity.

Assume we have a triangle with top vertex far from the viewer, middle vertex on the projection plan, and the bottom vertex is the closest vertex. The depth monotonically decreases from the top to the bottom. Figure 5.4 plots the original triangles (dashed lines) and projected triangles (solid lines) when the epipolar line is horizontal/slanted. Notice vertical disparity at the vertices.

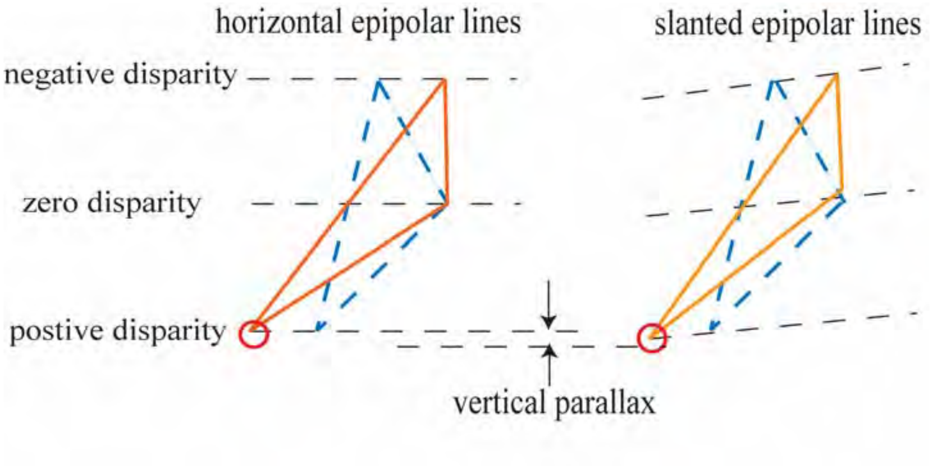


Figure 5.4 Illustration of vertical disparity.

To measure the vertical disparity, we find feature points for each sampled strip in one (right) panorama, and track these features in another (left) panorama using optical flow techniques [9]. There are many methods for computing optical flow (the image velocity of each pixel in image sequences), and Beauchemin *et al* [9] compares the accuracy, reliability and density of the most nine famous methods. The optical flow techniques are classified as: differential techniques, region-based techniques, energy-based techniques, and phase-based techniques. There are three major stages in optical flow methods: (1) prefilter or smooth images with low-pass/band-pass filthers; (2) extract of basic measurements such as spatiotemporal derivatives or local correlation surfaces; (3) integrate the previously extracted measurements to produce a 2D flow field. The differential techniques compute the velocity from spatiotemporal derivatives of image intensity. One famous method is Lucas-Kanade as we introduce before. The region-based methods define the velocity as the shift that yields the best fit between image regions at different times. Normalized cross-correlation and sum of squared difference (SSD) are usually used for the best match measurement. However, the region-based matching methods only perform well with translation images and produce poor results when motion field involves small velocities. The energy-based matching methods are based on the output energy of velocity-tuned filters in Fourier domain, and often found not reliable and sensitive to initial conditions. The phase-based matching methods define the velocity in terms of the instantaneous motion normal to level phase contours in the output of band-pass filter outputs, one of method is Fleet-Jepson. It produces the most accurate results, but computation load is very high due to large number of filters. After testing both real and synthetic image sequences, [9] shows that two most reliable methods are Lucas-Kanade and Fleet-Jepson. In this proposal, we choose Lucas-Kanade method to calculate the optical flow.

Figure 5.5 shows the vertical disparity values for each part of feature points as a function of vertical location (measured in pixels) in a strip as a function of horizontal strip location (measured in pixels). The horizontal axis shows 5,000 pixels out of a total image width of 7,560 pixels.

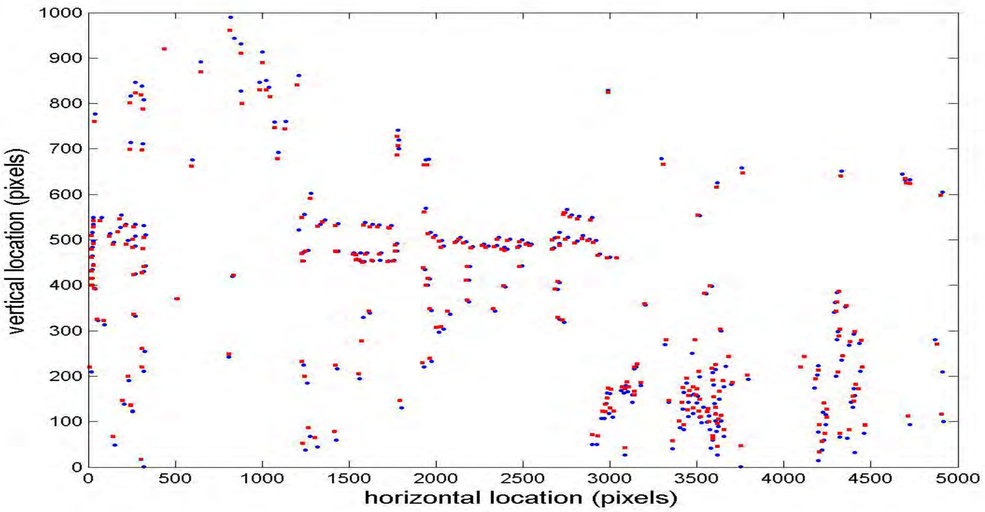
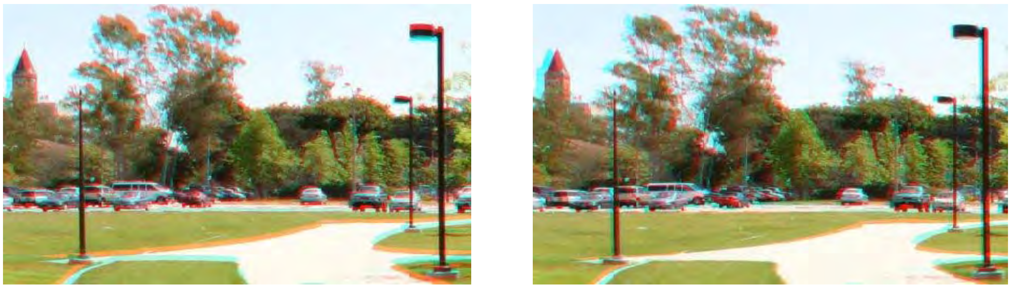


Figure 5.5 The vertical axis shows coordinates of feature points in the left and stereo pairs as function of horizontal location.

5.2 Vertical Disparity Correction

We tried two methods to correct vertical disparities: (1) Find a piecewise linear vertical disparity function $y_r = f(y_l)$ by linear interpolation among the feature points for each column. (2) Morph the strips by shifting the points vertically in the reverse direction following the inverse function $f^{-1}(y_l)$; or Calculate a simple transformation matrix for each sampled strips according to the tracked feature point pairs as (x, y_r) (x, y_l) (we neglect the horizontal disparity). Warp the sampled strips to achieve same projection plans for each strip pair. Both methods give satisfactory results. The anaglyph images in Figure 5.6 demonstrate the vertical disparity correction result. Note that there are vertically disparities (the vertical difference between red-cyan contour) in the left image (before vertical disparity correction), and almost none in the right image (after vertical disparity correction)



before vertical disparity correction after vertical disparity correction

Figure 5.6 Result of vertical disparity correction.

5.3 Horizontal Disparity Measurement and Adjusting

The horizontal disparity Δs_x of a point at distance D_p is simply derived from Figure 3.4 as

$$\Delta s_x \approx f(\frac{D_p}{D_p - D_{oc}})(\Delta\theta_{rl}) \tag{5.5}$$

where $\Delta\theta_{rl}$ is the angle difference between left/ right eye shots.

However, the position of the 3D scene point is unknown in real situations. R. Szekusji, *et al* [57] extracted the depth D_p from horizontal disparity using a tensor-voting method. Rather than reconstructing a 3D scene model, our goal is to adjust the disparity within the human stereo fusion range and enhance the stereo sense. To achieve this goal in general, a larger distance between strips is needed for distant scenes to get larger disparity; while the strips should be closer to the center to decrease the disparity for closer scenes.

For a more complicated scene structure, an accurate disparity measurement and adjustment is needed. Hence, we modify Peleg's [75] correlation-based disparity measurement algorithm by using optical flow methods to measure the disparity. The following are the steps in our disparity adjusting procedures.

Step 1: Create a stereo panorama pair with the methods mentioned in section 1 and 2.

Step 2: Align left/right panoramas: define objects at infinity to have zero disparity.

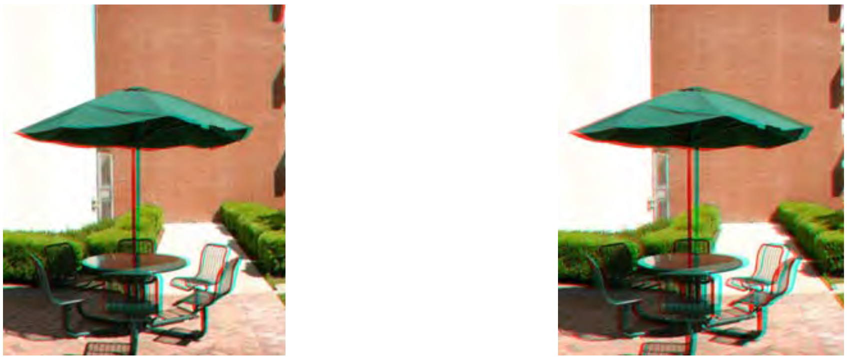
Step 3: According to the evaluation of optical flow techniques [9], we use Lucas-Kanade method to measure disparity.

Step 4: Find the maximum disparity of each column.

Step 5: Re-sample strips from data according to the maximum disparity. Stretch the distance between strips where the disparity is small to enhance the distance sensation; decrease the distance between strips to keep 30 pixels disparity for human stereo fusion limits. (Adjust W_v based on the disparity measurement)

Step 6: Keep the same left-right relationship for corresponding stereo points by maintaining a monotonic relationship between the separation of strips and the rotation angle.

Figure 5.7 shows the horizontal disparity adjusting results. We can see that the horizontal disparities of the umbrella and chairs are enhanced after we applied the algorithm. (the right image)



before

after

Figure 5.7 Results of horizontal disparity enhancement (anaglyph).

5.4 Computational Complexity of Disparity Adjusting for Stereo Panorama

Optical flow feature selection and tracking is the main process in our disparity adjusting method for stereo panorama. From Sec 4.1, the computational complexity of the optical flow algorithm is $O(vs)$ where the image size is s and the motion velocity between two frames is v. However, the typical size of one panorama is 7000x1024 pixels. Performing the feature selection and tracking will take a long time, if optical flow algorithms to the whole image. Therefore, our method is to divide stereo panoramas into several segments where a segment is twice as wide as a strip (the segment starts 1/2 w columns before the strip start point. w is the strip width as shown in Figure 5.8). For the vertical disparity correction, we resize the segment to be half the width of the original and apply optical flow tracking because only the vertical disparity is to be considered. The computational complexity in this case is $O(whv_p)$. Here, w is the strip width; h is the strip height; and v_p is the velocity between stereo panoramas which is less than the strip width. Similarly, we resize the segment with half height of the original (Figure 5.8) for feature selection and tracking in horizontal disparity adjustment.

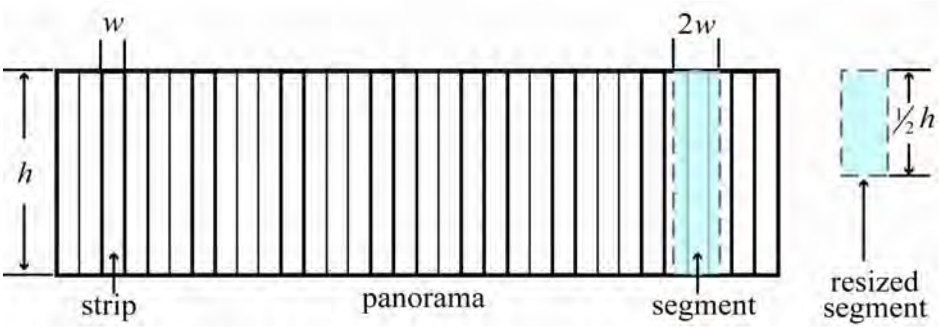


Figure 5.8 Illustration of a segment for vertical disparity adjusting.

For vertical disparity correction, the next step is to find the transformation matrix for each strip using the features that have been identified. Assume that the (x, y) coordinates of features in one segment in the left panorama can be written as a vector P_l ; and the coordinates of the corresponding features in one segment in right panorama are be denoted as P_r. We define the coordinate transformation matrix A as $P_r = AP_l$, and it can be found as the solution of a set of linear equations in the form $A = P_r P_l^T (P_l P_l^T)^{-1}$. The computational complexity of this solution using LUP decomposition is $O(n^3)$ [21], where $n = 2$ is the dimension of matrix A. Finally, we correct the vertical disparity by calculating the new vertical positions for every point p_l in left panorama via $p_r' = Ap_i$. This requires $sn^2 = 4s$ multiplications, where s is number of pixels in the images.

For horizontal disparity adjustment, we then find the maximum disparity value for each column. The computational complexity is $O(h)$ (h is the height of a column, the maximum number of feature can have in one column) for this operation. Then, the positions for re-sampled strips are given from the maximum and desired disparity adjustment as required. We then create a new stereo panorama by stitching these re-sampled strips. The computational complexity for this is same as our image stitching algorithm for stereo panorama generation.

Chapter 6 Object Based Disparity Morphing for Stereo Panorama

6.1 Object Selection and Image Segmentation

We describe an object selecting tool for stereo image pairs based on mean-shift segmentation. The tool, coded with the help of OpenCV [43], effectively extracts the object boundary specified by users, and also encodes the contour by B-splines to allow users to modify and refine the object boundary. It also automatically selects the object contour in both left and right images of stereo pair, and is useful for stereo image editing and stereo disparity adjusting. Once the selected region is found, we use disparity adjustment algorithms in chapter 5 to change the perceived stereo effect in images or panoramas.

6.1.1 Interactive Object Selecting

Some level of human interaction is still required for traditional image editing. Vision-based image editing can make traditional spatial-based image editing more efficient by helping the user interaction or guiding the vision algorithm to improve the results, which is very useful for video or image editing. Boundary-based and region-based methods are the two most common techniques used in current vision-assisted image editing tools [58]. Boundary-based methods allow users to input some seed nodes close the desired region boundary, and the system traces the boundary and finds the optimal contour, such as Snakes [50], Intelligent Scissor [63][64], and Image Contour Editing (ICE) [28]. Region-based methods allow users to locate seed points to select a region, which does not enclose the desired object within pixel accuracy. Example systems are Magic Wand in Photoshop, Intelligent Paints [80][101].

In this section, we present an object selecting method for stereo images, which has the advantages of both region-based and boundary-based methods. Like region-based algorithms, our tool is built based on the image segmentation results of the mean-shift algorithm [22][23] to let users easily select the region of interest with less effort. An object contour is presented after the user inputs the vertices to enclose the significant parts of the object. In this way, no extensive work is needed to guess or carefully draw a contour tracing of the object boundary. Then, we encode the initial contour by B-spline curves. The users can then deform, refine, smooth the contour of the selected object, and override any of the results. Finally, a template matching algorithm is used to find the corresponding object in the companion stereo image. Thus, we provide users with an interactive, efficient and automatic stereoscopic object image selecting tool, which can be used for modifying the disparity of certain objects in a stereo panorama or to change the property of certain objects in stereo image pairs simultaneously.

- User Interface: Initial Steps

To begin, users mark a set of points that serve as ordered polygon vertices to specify the object of interest, instead of tracing the object boundary or carefully selecting the seed points. By clicking the left button of the mouse, a set of red circle marks is placed on the image. Note that the algorithm does not require marks to be very precisely located inside, outside or exactly on the object boundary as shown by the example in Figure 6.1.

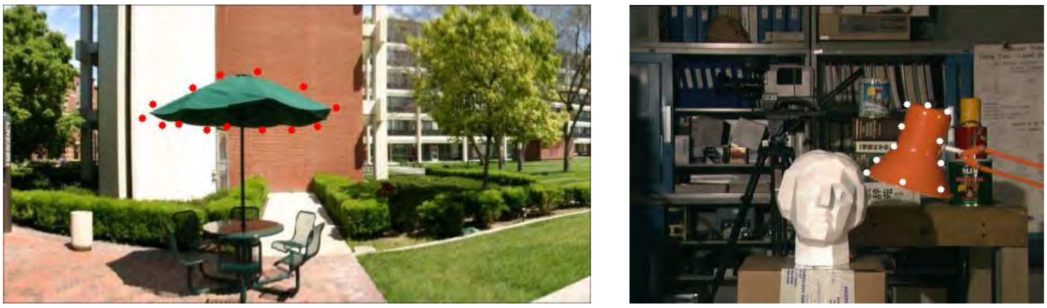


Figure 6.1 Left images of stereo pairs with user input marks.

6.1.2 Initial Object Selecting with Mean-Shift Segmentation

After the user inputs the marked point as shown in Figure 6.1, our algorithm calculates the minimum bounding rectangle that contains all the marks. Assuming the rectangle is $w \times h$ pixels centered at (x_c, y_c), we apply the mean-shift algorithm [22][23] to segment the image in the area (the gray rectangular box in Figure 6.2) with size $(w + m_w) \times (h + m_h)$ and same centroid as the minimum bounding rectangle.

The mean-shift algorithm is a robust nonparametric density estimator algorithm, which does not require the prior knowledge of the number of clusters, and does not constrain the shape of the clusters. It can be applied to both gray-level and color images, and has many applications, including discontinuity preserving filters and image segmentation. For image segmentation, the mean-shift algorithm uses a dynamic kernel window. Given n data points $\mathbf{x}_i$, $i = 1...n$ on a d-dimensional space R^d, and the kernel density estimate obtained with kernel $K(\mathbf{x})$ and window radius h is

$$f(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h} \right) \tag{6.1}$$

For any symmetric kernel $K()$, the profile satisfying

$$K(\mathbf{x}) = ck(\|\mathbf{x}\|^2) \tag{6.2}$$

, where c is a normalizing parameter which assures $K(\mathbf{x})$ integrates to 1. Then it moves the window in the direction of the maximum increase in the joint density gradient and stores the convergence points. The procedure is: (a) compute the mean-shift vector $m_h(\mathbf{x}^t)$ as

$$m_h(\mathbf{x}) = \frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x} \tag{6.3}$$

, where $g(s) = -k'(s)$; (b) translation of the window from $\mathbf{x}^t$ to $\mathbf{x}^{t+1} = \mathbf{x}^t + m_h(\mathbf{x}^t)$. Finally, it groups the clusters that have convergence points close to each other.

Here, we set m_w and m_h as 10 pixels. Figure 6.3 shows the mean shift segmentation result after the user inputs the marks.

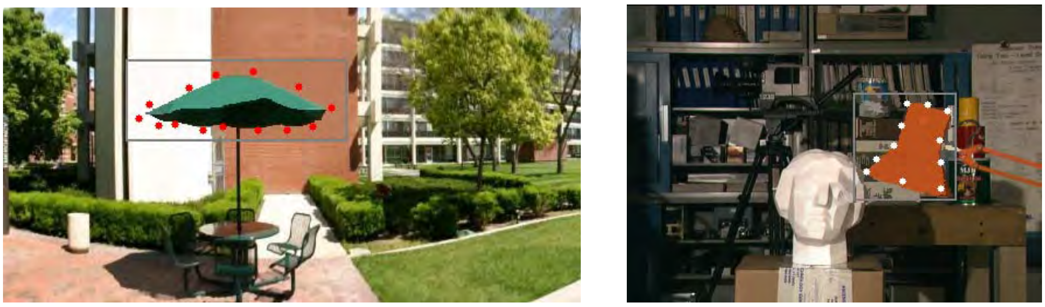


Figure 6.2 Image by the mean-shift segmentation method after the user marks.

6.1.3 Initial Object Contour Selection and Polygon Filling Algorithm

A region map, in which each pixel is assigned a region label, and a region list which lists all the regions with information, such as total number of pixels, pixels in the region, boundary points, and neighboring area, is created for the processed area after mean-shift segmentation. Since our input is a sequence of ordered vertices of a polygon, we modify the efficient scan-line polygon filling algorithm often used in computer graphics [32], to check if a segmented region has been selected as part of interested object. We check the region label of every point inside the polygon in the region map by this algorithm, and calculate the percentage of pixels inside the polygon for each segmented region. A segmented region belongs to our desired object when the percentage of pixels inside the polygon exceeds a certain threshold. Our initial object contour is the overall contour of all selected segmented regions. In Figure 6.3, the yellow curve shows the initial contour of the user selected object boundary.

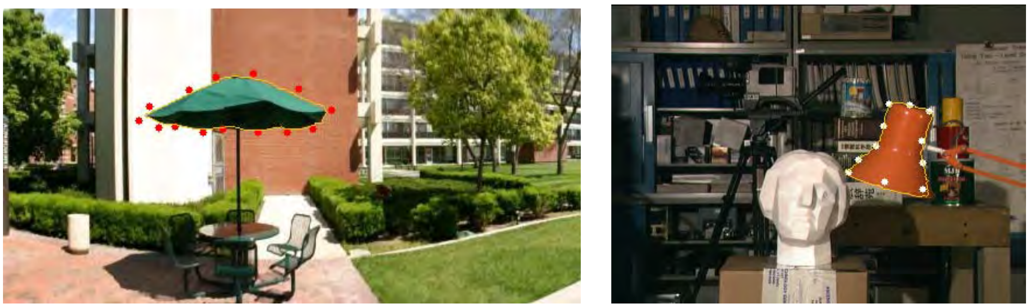
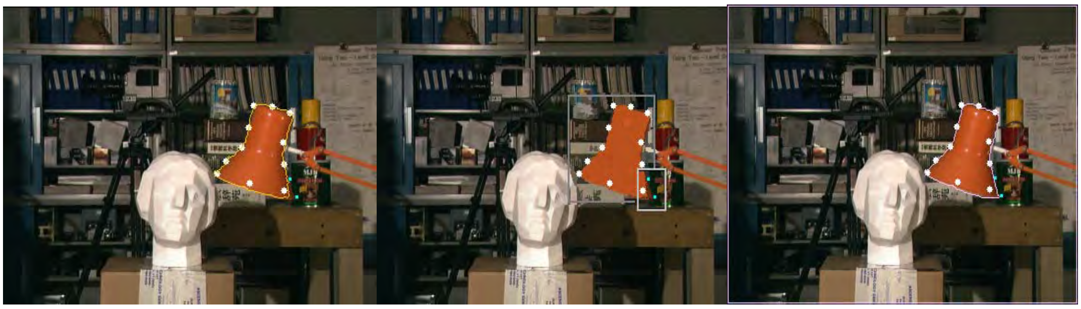


Figure 6.3   Initial contour (light colored trace) of the user selected object.

Users may add or delete selected polygon vertices near the correct boundary to modify the initial contour further. We then apply the mean-shift based method algorithm again to an added region (shown in the gray bounding box in Figure 6.2) to find the contour of this region. Then, we update the initial contour to include the new contour. The sequence of steps is shown on Figure 6.4.



initial contour user added marks modified initial contour

Figure 6.4 The process of user modified initial contour.

6.1.4 Interactive Contour Refinement and B-Spline Curve

Our contour refinement method is based on B-splines [20]. Some control points are automatically placed along the initial contour as shown in Figure 6.5, and the number of control points can be decided by the user. B-splines are one kind of spline functions (flexible globally smooth strips for modeling any arbitrary function) that are currently very popular in shape control. They have many advantages such as: (1) moving a control point only affects the local curve; (2) any number of points can be added without increasing the degree of the polynomial; and (3) closed curves can be created by repeating the first few control points at the end of the control point sequence. In B-splines, a curve is divided into several segments, and the nodes joins two curve-segments are called knots. A simple method to specify the knot position is a uniform spacing, and control points can be defined near the knots at the side of the normal direction of the curve. Assume we have k control points denoted by $V_i = (x_i, y_i)$, and the curve is then approximated by a linear combination of the basis functions as

$$C(t) = \sum_{i=0}^{k} V_i N_{i,p}(t) . \tag{6.4}$$

Where p is the degree of a basis function, and the basis functions are defined as

$$N_{i,0}(t) = \begin{cases} 1 & \text{if } t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{6.5}$$

$$N_{i,p}(t) = \frac{t - t_i}{t_{i+p} - t_i} N_{i,p-1}(t) + \frac{t_{i+1+p} - t}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(t) \tag{6.6}$$

Here we use a third-degree basis function, generally called a cubic B-spline. With these control points, the user can drag the curve to further refine its shape.

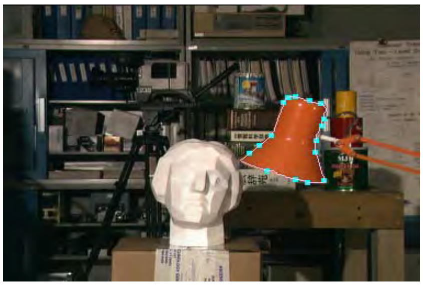


Figure 6.5 The initial contour with control points.

6.1.5 Object Selecting in Stereo Images

Once the contour of the selected object is defined, the minimum bounding box of size $w_t \times h_t$ enclosing the contour is calculated. Our software automatically crops this part of the image, finds its center at (x_t, y_t), and saves it as a template. We then apply a template matching algorithm which simply slides the template through the region of interest (ROI) in another (target) image, of size $(w_t + d_h) \times (h_t + d_v)$ centered at (x_t, y_t), and compares overlapped patches with the template by calculating the correlation coefficient to find the corresponding part as the process shown in Figure 6.6. Here d_h and d_v is the maximum horizontal and vertical disparity assumed. Figure 6.7 gives the result. The yellow line shows the contour. Images at the left side are the image for left eye in the stereo image pair, and vice versa.

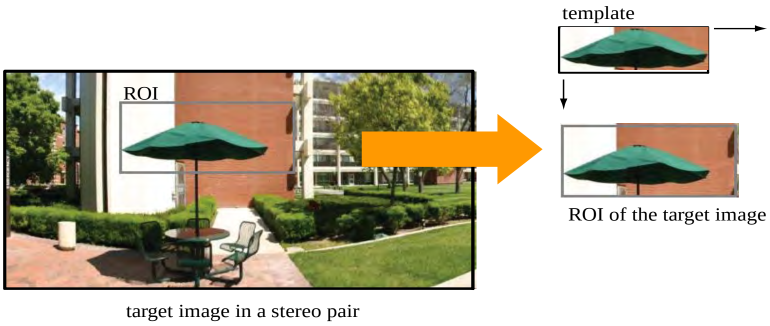


Figure 6.6 The template matching process.

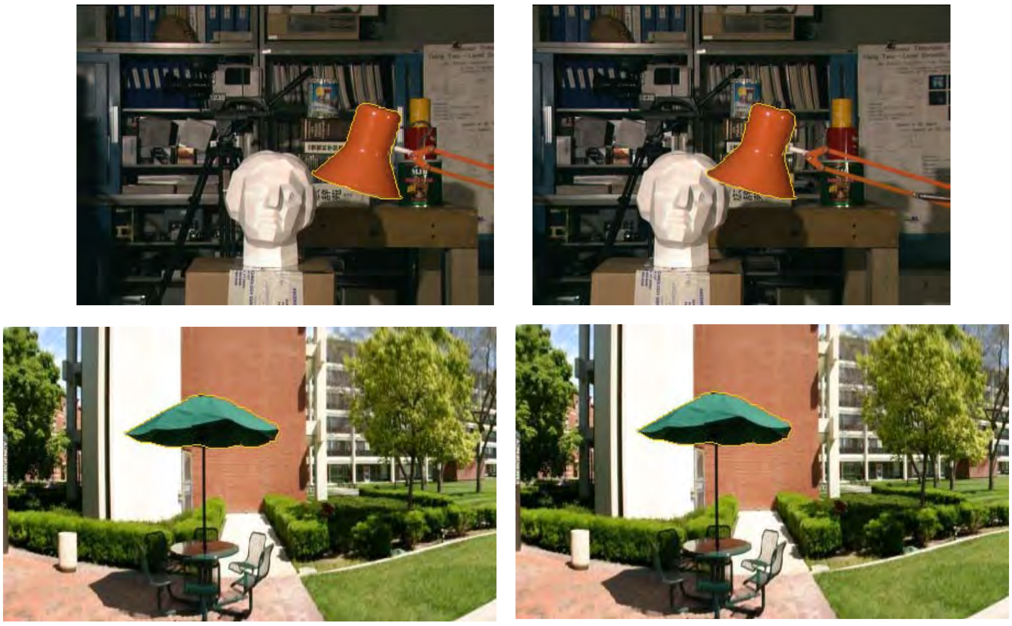


Figure 6.7 Stereo object selecting results.

6.1.6 Automatic Object Selecting

We describe here an alternate procedure that automatically selects objects that fall into user specified disparity ranges. The user selects one or more disparity ranges specifying the upper and lower disparity limits for each in pixels. We tried two different stereo image segmentation and disparity map generation methods (belief propagation and cooperative stereo matching algorithm [99]). However, we found the results of both methods are highly influenced by input parameters and the quality of images. Also the second method is time consuming. To achieve the goal of efficient and robust selection of regions fall within a given disparity range, we developed our own algorithm. The advantage of this method is that we do not need carefully chosen template-matching window sizes. The computational efficiency is constrained primarily by the efficiency of the mean-shift algorithm.

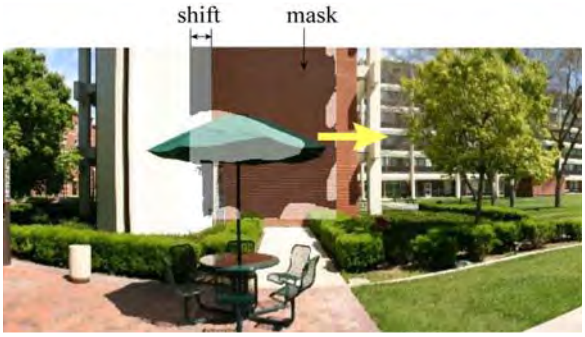
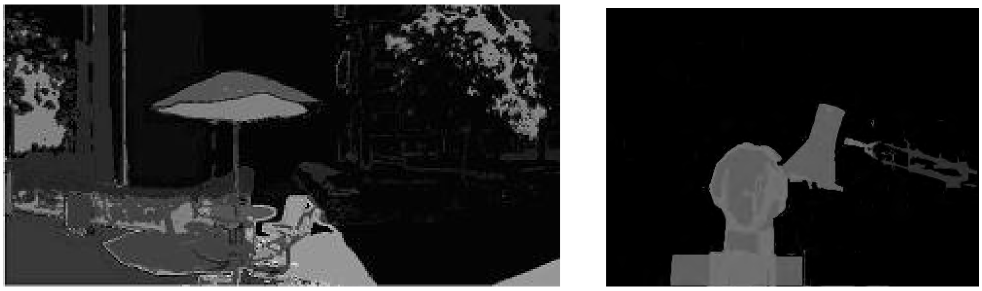


Figure 6.8 Mask shift illustration.

Our method is as follows. We first segment one image (left image) using the mean-shift segmentation method. We then create three separate color histograms for the red (R), green (G) and blue (B) components of the image and a mask including all pixel positions for each segment. We call these template histograms (H_t) for each color component. Figure 6.8 shows a mask for an example segment (a building wall). This mask is then shifted left and right over a range $\delta_{\min}$ to $\delta_{\max}$ pixels above the other image (right image). We then calculate the color histogram of the pixels under the mask, and compare the dissimilarity between the current histogram H_i and the template histogram H_t using the Chi-Square test

$$d(H_i, H_t) = \frac{1}{2}\sum_k (H_i(k) - H_t(k))^2 / (H_i(k) + H_t(k)) \tag{6.7}$$

where $d(H_i, H_t)$ is between 0 to 1. The dissimilarity measure in Eq. (6.7) is also calculated separately for the R, G, B components and these results are averaged to give a composite $d(H_i, H_t)$. The disparity of a segment is equal to the horizontal shift giving the minimum composite dissimilarity. The user also inputs a dissimilarity threshold that will be exceeded by the composite $d(H_i, H_t)$ when there is an occlusion that significantly changes the shape of a segmented region between left and right views. In addition, the disparity may be outside any of the user-selected regions. In these cases user-specified thresholds eliminate these regions from the disparity measuring procedure. Figure 6.9 gives our automatic object selection results.



(Left: $\delta_{\min} = 1$, $\delta_{\max} = 10$ (1000x512) ; Right: $\delta_{\min} = 10$, $\delta_{\max} = 14$ (384x288)

Figure 6.9 Automatically selected regions.

We also tried to use a *k*-means image segmentation algorithm with five input parameters (red, green, blue and spatial positions) to improve the speed of the algorithm. The *k*-means segmentation algorithm generally does not work as well as the means-shift algorithm, and objects within a scene may be segmented into two or more pieces with slightly different disparities. In this case, the segments can be regrouped to form a layer, and used to iteratively revise the segmentation and disparity maps. We can also extend this method to generate disparity maps and mark occlusions efficiently for general stereo images.

6.2 Region Based Horizontal Disparity Adjusting

The two main reasons for adjusting the horizontal disparity are to keep the two views within the human stereo fusion range and to enhance selectively the perceived sense of depth. Peleg [75] computed horizontal disparities between left/right panorama views using simple correlation windows. He then adjusts the local vertical strip separation in the left/right views using a larger distance between strips for distant objects and a close distance for nearby objects.

In this section, we describe a more general method to adjust the disparity of selected regions. This method modifies the disparity of selected 2D regions without modifying other pixels located in the same columns of the image containing these regions.

6.2.1 Panorama Index Table

In assembling right and left panoramas from the sequence of images, we define the coordinate system of the right panorama with the variables (X_{pr}, Y_{pr}) as shown in Figure 6.10. A particular point in this panorama is denoted by (x_{pr}, y_{pr}) as shown. This point corresponds to a pixel $(x_{r1}, y_{r1}, \theta_{r1})$ within some strip taken from a particular frame indexed by rotation angle θ. We create a panorama index table that contains this mapping, and create another table for the left panorama mapping with variables having subscript l as before.

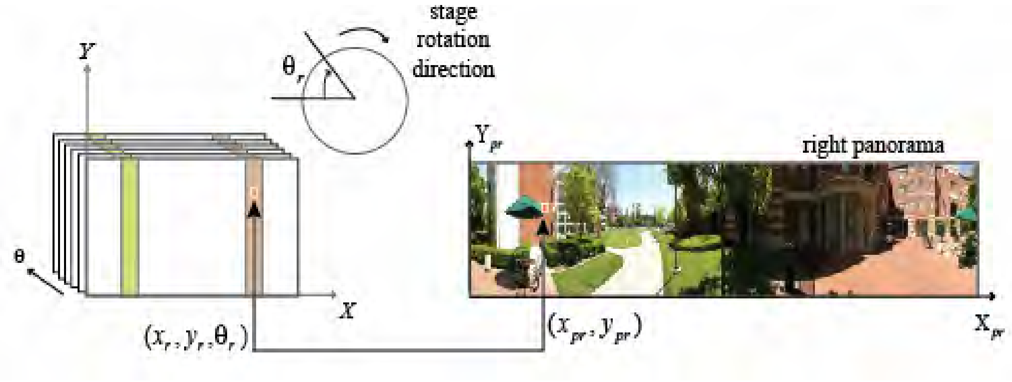


Figure 6.10 Panorama index table illustration.

6.2.2 Image Segmentation

To begin the procedure, we segment one panorama using the mean-shift algorithm [22] [23], as summarized in section 6.1.2. After segmentation, we apply the region based horizontal disparity adjusting to enhance/decrease the disparities within certain region. Figure 6.11 shows part of a segmented right panorama. Given the segmented image, in this proposal we select a region for disparity adjustment by hand for the sake of illustration. Automated selection of regions for disparity adjustment is also possible.

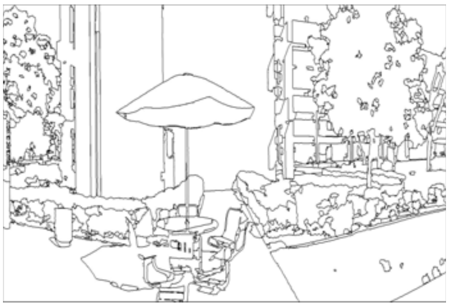


Figure 6.11 Segmented right panorama using the mean-shift algorithm.

6.2.3 Region Based Horizontal Disparity Adjusting

To simplify the explanation of our region-based disparity adjusting, assume that we keep the right image fixed. Figure 6.12 shows the process. At the upper left is a selected region in the left image; at the center is the same region as it appears in the right image. The upper right shows the disparity Δs_x between the two, which can be estimated by using optical flow [9]. The lower left shows the selected region with enhanced disparity, and the lower right shows the adjusted disparity $\Delta s_x'$.

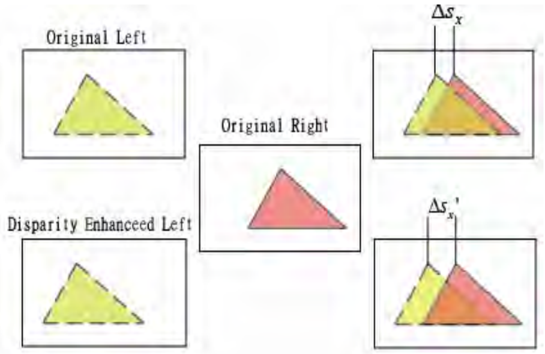


Figure 6.12 Illustration of simplified disparity enhancement.

6.2.4 Disparity Adjusting Process

We create a mask coincident with the region selected for disparity enhancement. To enhance the disparity in this region by an amount Δx, we shift the selected region in the left image to the left (in the direction of increasing disparity) as in Figure 6.13. Next, we fill the blank region by looking up in the “left panorama index table” to find the location of the original pixel and calculate the location of a replacement pixel as we describe.

To understand the process, we compute the amount of shift of the image of a point object from frame to frame. In Figure 6.14, $\Delta\theta_{rl}$ is the angle between the left and right views;

$\Delta\theta'_{rl}$ is the angle between the adjusted left and right views. Also shown in Figure 6.14 are four points: a filled circle (dot); a filled square; a star; and an arrowhead.

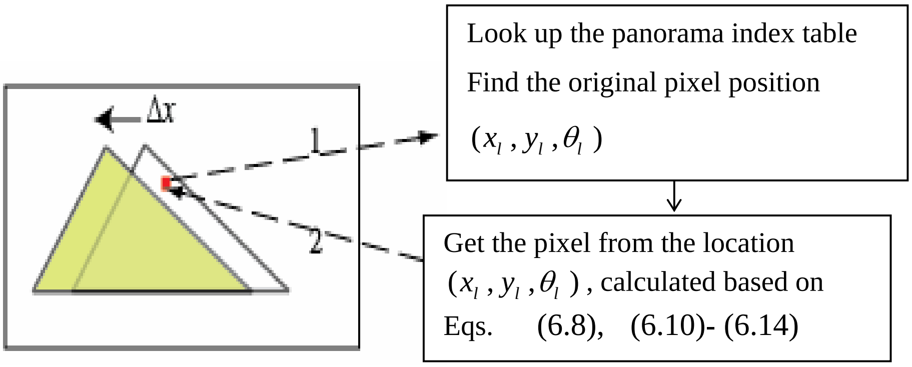

Δx
1
2
Look up the panorama index table
Find the original pixel position
(x_l, y_l, θ_l)
Get the pixel from the location
(x_l, y_l, θ_l), calculated based on
Eqs. (6.8), (6.10)- (6.14)

Figure 6.13 Region based disparity adjusting process.

For a camera at position O_{cr}, the dot and square are located in the plane of fixation and their images map to the left and right boundaries of the strip that is assembled to make the panorama. The images of the star and arrowhead map to positions between the dot and the square as shown in the diagram just to the left of the view on the circle in Figure 6.14. Just to the right of the circle in Figure 6.14 are similar results for cameras at positions O_{cl} and O_{cl}'. Note that the dot and the square still mark the boundaries of the sampling strip, but the relative positions of the star and the arrowhead have shifted compared to the image taken at O_{cr}. It is this shift that produces the disparity leading to stereo perception. Note that the image of a point on the "point of fixation plane" is shifted λ pixels when the camera moves from location O_{cl} to O_{cl}', while the images of the objects not on the point of fixation plane will also shift λ pixels and an additional amount $(\Delta s'_x - \Delta s_x)$ because of the change in disparity. From Eq. (5.5), when

$\Delta\theta'_{rl} - \Delta\theta_{rl} = \Delta\theta$, λ is equal to the sampled strip width W_s (this is how strips sampled from the same position are concatenated to form a panorama), and $\Delta s'_x - \Delta s_x$ can be linearly approximated as ds because $\Delta\theta$ is small

$$ds \approx \Delta s'_x - \Delta s_x = (\Delta\theta / \Delta\theta_{rl})\Delta s_x \tag{6.8}$$

Thus, the images of an object shift by the amount

$$(W_s + ds)(\theta' - \theta) \tag{6.9}$$

from frame indexed as θ to frame indexed as θ'.

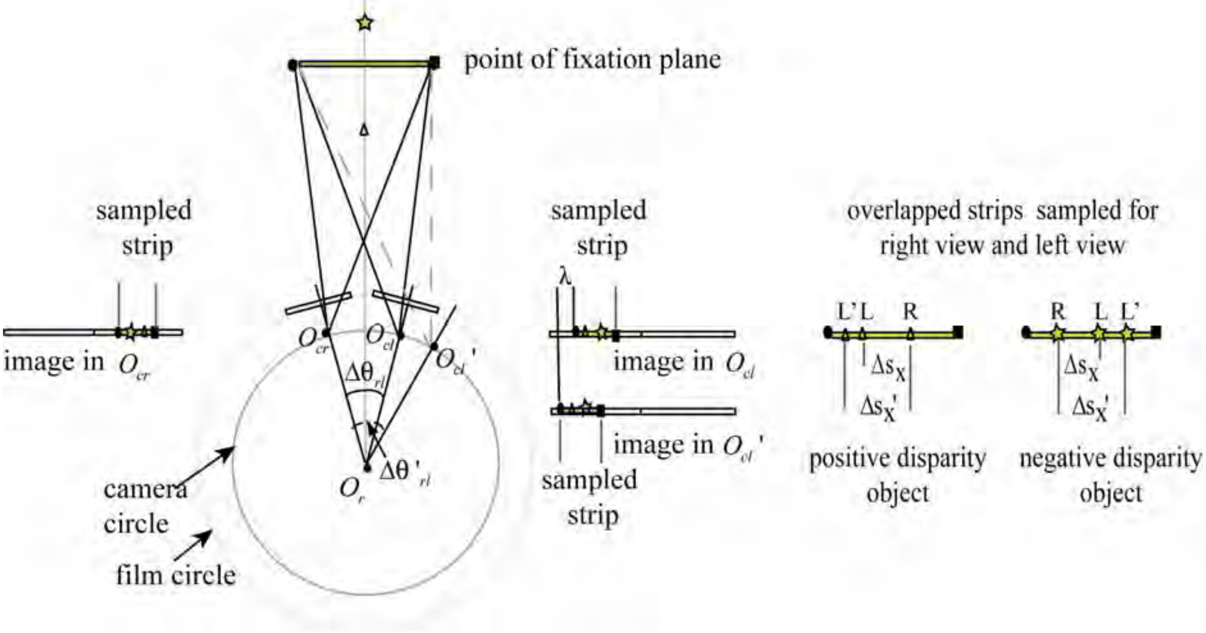


Figure 6.14 Sampled strip position and disparity

As we estimate the shift of a pixel mapped from the same point in a scene, we derive Eq. (6.10)-Eq. (6.14) based on the strip sampling process and geometry. Here, the strip is sampled at position W_v with strip width W_s in the original strip sampling process that generated the left/right panorama, and θ_l^s is the index of the starting frame. Assume we have a new adjusted strip sampling process to enhance the disparity, where the sampled strip is located at W_v' with the same strip width, and the index of the starting frame is $\theta_l^s{}'$. The pixel removed after moving the mask is located x_l^s, x_l in the θ_l^s and θ_l frames; while the pixel we need to fill the blank is located at x_l^s, x_l' in θ_l^s and θ_l' frames, where these parameters are given by

$$W_v' = W_v + \Delta x \quad . \tag{6.10}$$

$$\theta_l^s{}' = \theta_l^s + \frac{\Delta x}{W_s + ds} \quad . \tag{6.11}$$

$$x_l^s = x_l - (\theta_l - \theta_l^s)(W_s + ds) \tag{6.12}$$

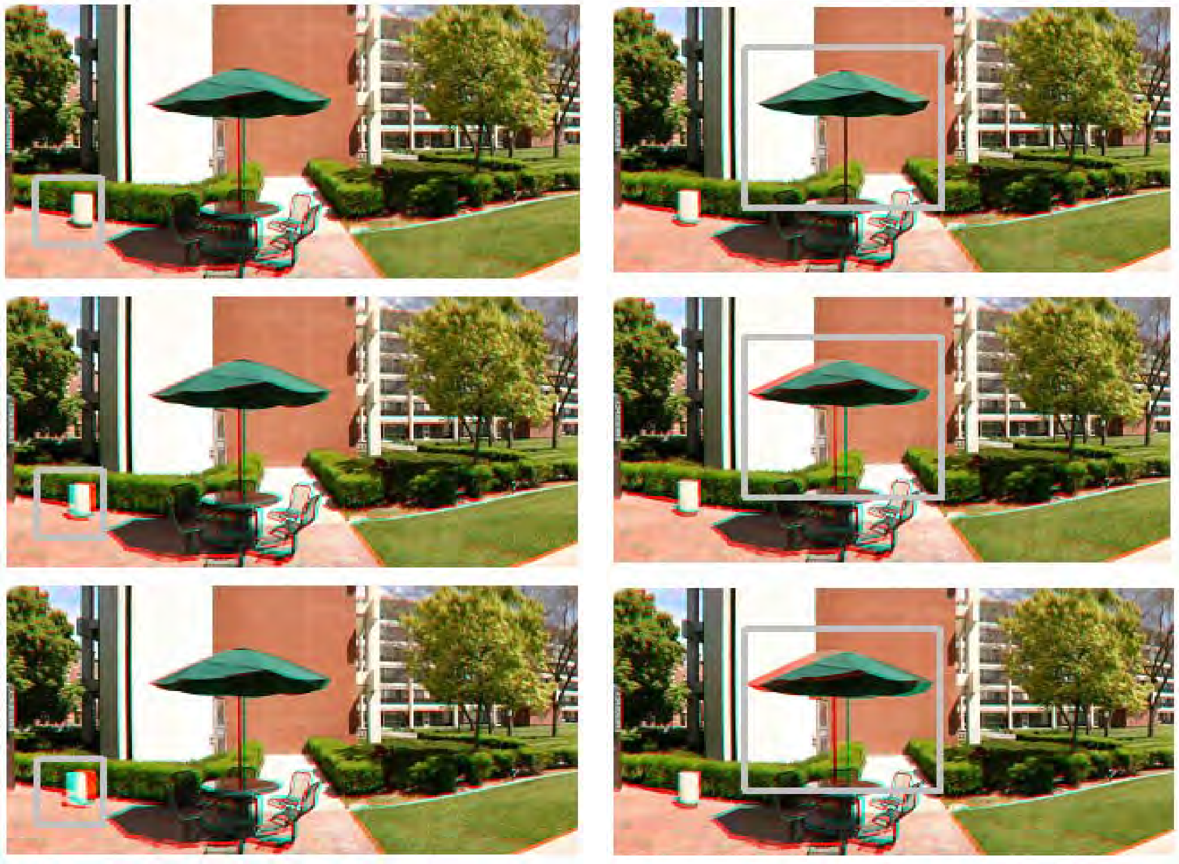
$$\theta_l' = \theta_l^s{}' + fix(\frac{W_v' + W_s/2 - x_l^s}{W_s + ds}) \tag{6.13}$$

$$x_l' = x_l^s + fix(\frac{W_v' + W_s/2 - x_l^s}{W_s + ds}) \cdot (W_s + ds) \tag{6.14}$$

and fix means the integer quantized result.

With known $x_l, \theta_l, \theta_l^s$ and Δs_x, we can calculate the x_l' and θ_l' using Eq. (6.8) and Eq. (6.10) - Eq. (6.14) to get pixels out of the dataset and fill the blank region. Note that we do not discuss y_l', because usually $y_l' = y_l$. In order to avoid moving objects appearing in the results, we apply the Gaussian mixture stitching method described earlier to the data set used to fill in the missing pixels as well. A similar process can be applied to the right panorama.

Figure 6.15 demonstrates our region-based disparity adjusting results, where each image (2000x1024 pixels) is part of the high-resolution stereo panorama. The disparities of the trash can are enhanced in the left three images, while the disparities of the umbrella are enhanced in the right three images. Note that all other disparities remain the same except in the region that we have selected.



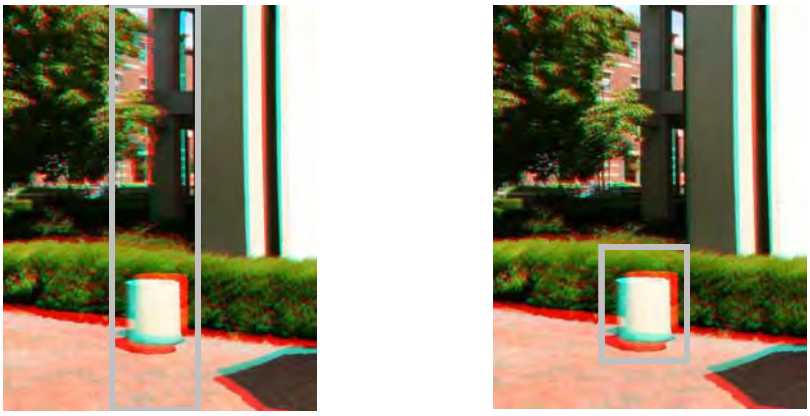
trash can at bottom left

umbrella

Figure 6.15 Results of region-based disparity adjusting.

Figure 6.16 and Figure 6.17 compare our results with those generated by Peleg's disparity adjusting method. We can see that in the left image of Figure 6.16 (Peleg's method), not only the disparities of the trash can but also the disparities of the background pillar, which is in the same column as the trash can, are enhanced. Similarly, the disparities in the same columns of the selected region such as the chairs and the door behind the umbrella are also enhanced in the left image of Figure 6.17. However, our method only changes the disparities of selected region as in the right image of Figure

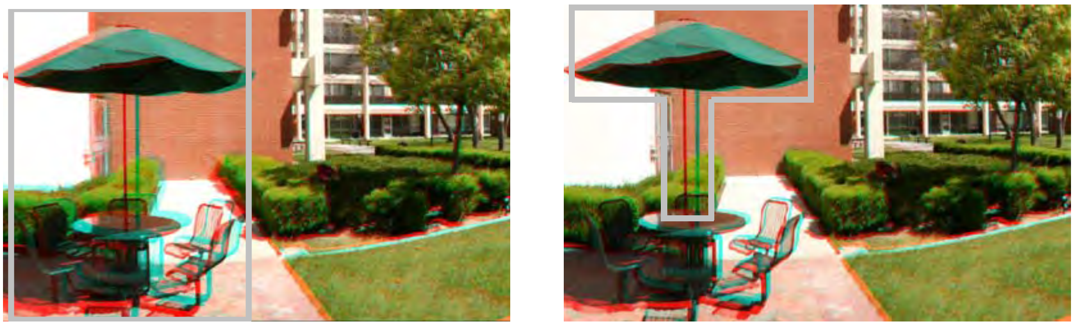
6.16 and Figure 6.17. Thus, we have developed a method to adjust the disparities in selected regions. A similar process can also be developed for different image capture setups.



Peleg’s disparity adjusting method

Our region-based method

Figure 6.16 Disparity-adjusted result of the trash can (anaglyph).



Peleg’s disparity adjusting method

Our region-based method

Figure 6.17 Disparity-adjusted result of the umbrella (anaglyph).

Chapter 7 Disparity Morphing for General Stereo Images and Video

In this section, we develop a general framework for disparity manipulation and morphing for stereo images and video. Figure 7.1 illustrates this framework: (1) disparity maps are generated from given data using various techniques; (2) different methods are provided for the user to morph the disparity maps; (3) new stereo pairs are created according to the original input data and morphed disparity map using depth-based image rendering methods that we describe.

First, we discuss disparity map generation techniques for different original input data types, including monoscopic images, monoscopic video, stereo image pairs, and stereo video. When the input data is a set of stereo images/video captured from different perspectives, the disparity map can be calculated using current stereo-matching techniques [85][99]. When the input is a general monoscopic video (2D video), the disparity maps can be generated using current stereoscopic video (3D video) conversion techniques [102][44]. In addition, we can generate an artificial depth map using depth from shadow or foreground-background separation techniques, if the data is only one monoscopic image. We discuss details of these techniques in section 7.2.

Next, we describe several methods for the user to manipulate the disparity map. In the first, the user defines input-output disparity mapping curves for selected regions, where the user may alternatively employ an interactive object-selecting tool by inputting seed points near the desired object boundary, from which the object is segmented using a mean-shift algorithm [23]. In the second method, the user can interact with 3D free-form disparity surface. A third method provides corrections to and conversions between the two common stereo camera capture setups: “toe-in” and “off-axis, and we give details of their mathematical model. The user can synthesize stereo image pairs or videos with specified stereo camera settings, or provide automatic disparity conversion between stereo capture formats. We show several morphed disparity map examples for each disparity manipulation method.

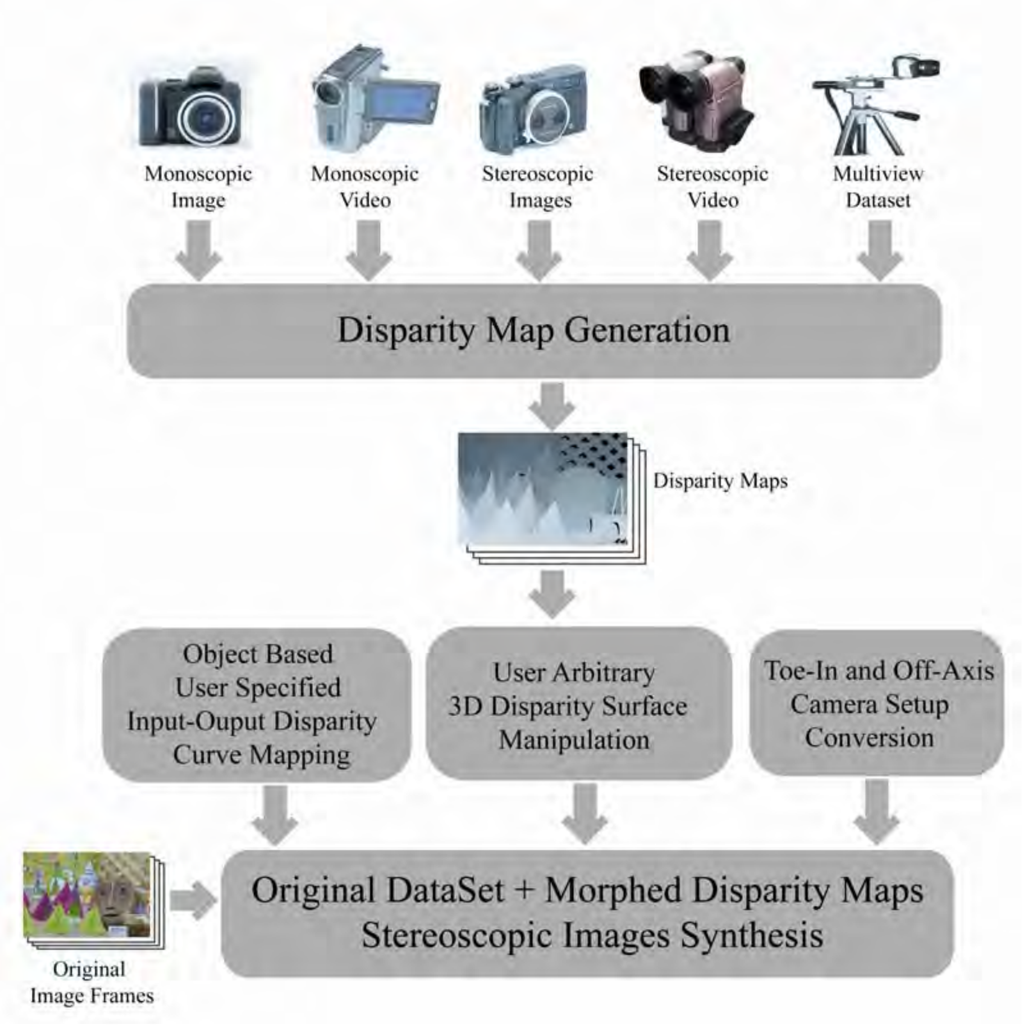

Monoscopic Image
Monoscopic Video
Stereoscopic Images
Stereoscopic Video
Multiview Dataset
Disparity Map Generation
Disparity Maps
Object Based User Specified Input-Ouput Disparity Curve Mapping
User Arbitrary 3D Disparity Surface Manipulation
Toe-In and Off-Axis Camera Setup Conversion
Original DataSet + Morphed Disparity Maps Stereoscopic Images Synthesis
Original Image Frames

Figure 7.1 General disparity morphing tool framework.

Finally, we describe disparity-based image rendering to synthesize new stereo image pairs from given original stereo image pairs based on a morphed disparity map. The synthesis method includes image warping, data-filling and disparity map smoothing procedures. We generate a set of corresponding views based on the morphing depth map using our developed algorithm, and render it on various displays including autostereoscopic (AS), HMD displays, or shutter glasses.

7.1 Disparity (Depth) Map Generation

We classify disparity (depth) map generation methods into five categories according to the available original data: Monoscopic Images, Monoscopic Video, Stereo Images, Stereo Video, and Multiperspective Image Data Set.

7.1.1 Monoscopic Image

Given a single monoscopic image, several methods can be used to generate a coarse disparity map. One method is to use foreground and background extraction methods with user guidance. This method can only segment images into a small number of discrete disparity layers. Shape from shading [91] provides another way to create depth/disparity maps from single monoscopic image. However, this technique is more suitable only for scenes that have fairly uniform color and texture, and the resulting disparity map is not very accurate. Saxena [83] took a supervised learning approach to estimate the depth map from a single monocular image. He began by collecting a training set of monocular images of unstructured outdoor environments and the corresponding ground-truth depth maps. Then, he used a discriminant-trained Markov Random Field, which incorporates multiscale local and global image features and models depths at individual points as well as the relation between depths at different points, to predict the depth map as a function of the image. Their results showed that it is able to recover fairly accurate depth maps even for unstructured scenes. If the given data is a portrait containing the head and shoulders of a human (e.g. a passport photo), Weerasinghe [115] presented an algorithm based on facial features (which first removes the existing background, locates main facial features, and uses simple geometry representation) to generate a low complexity parametric depth map.

7.1.2 Monoscopic Video

Motion vectors (the motion of an object relative to the camera) can be calculated from monosopic video using optical flow techniques. Matsumoto [60] developed a method (implemented in Sanyo's commercial product) to extract depth maps using a motion-based depth decision algorithm. However, this method assumes a relatively small camera panning across a stationary scene, and can only recover relative depth accurately if the motion of all objects is directly proportional to their distance from the camera. Garcia [35] developed a method based on spatial-temporal interpolation, which calculates depth information using motion variation in images and determines the maximum disparity value using image offset due to the depth information. Both of these above methods require stable image analysis to determine direction and velocity of a moving object. Choi [19] developed a real-time stereoscopic image converter that generates stereoscopic image pairs with different perspective depth using motion parallax by computing the motion parallax between two adjacent 2D video frames with methods for motion detection, region segmentation and depth map generation. This converter can automatically produce a realistic 3D effect regardless of the direction and velocity of a moving object in the 2D video frames.

7.1.3 Stereoscopic Image Pair

Stereo matching is the technique to calculate a disparity map based on stereo images. Scharstein and Szeliski [85] performed a complete evaluation of the twenty best known dense two-frame stereo correspondence (matching) algorithms. According to their evaluation, global optimization methods based on 2-D MRFs generally perform the best in all regions of the image (overall, textureless, and discontinuities). Kolmogorov's graph cut [53] and Sun's belief propagation [99] algorithms also generally give most good disparity maps. However, these methods may be too computationally intensive for real-time applications. If efficiency is an issue, stereo matching using a simple shiftable window method is a good choice. In particular, Hirschmüller's method [39] is among the fastest and produces very good results.

7.1.4 Stereoscopic Video

To calculate the disparity map for a stereo video stream, we can use a approach similar to that used in stereo image pairs. An advantage is that the disparity information of previos frames can be used to generate the disparity map of the next frame. In some cases where regions of a video have little texture, there will be some inaccuracy in estimating the disparity map. However, in this case, the viewer generally will not perceive any artifacts. It is more important that there be a smooth evolution of disparity maps form frame to frame to avoid the generation of perceived artifacts. Criminisi [24] developed a four-state dynamic programming method to calculate the disparity map and generate new views at real-time video rate. Other real-time dense disparity map stereo systems and their performance are also summarized in [15].

7.1.5 Multiperspective Image Data Set

Given a multiple view image data set, a more detailed 3D model can be calculated and disparity maps can be generated accordingly. Hartley and Zisserman [37] and Faugeras and Luong [30] provide detail information on the geometric aspects of multipleview stereo. Fua and Leclerc [34] modeled the scene as a mesh that is iteratively updated to minimize an objective function. Faugeras and Keriven [31] proposed a similar method that models the scene using level sets. Kutulakos and Seitz [54] represented the scene as a volume and proposed a space carving method to refine the surface.

7.2 Disparity (Depth) Map Morphing

In this section, we let users morph disparity maps with three different techniques as follows: an object-based user interactive interface; a 3D surface user interactive interface; and toe-in and off-axis disparity conversion.

7.2.1 Object-Based User Interactive Interface

In this method, the user selects an object from a segmented disparity map. As an example, we use a simple stereo image pair with synthetic texture and only one foreground object in this chapter. The stereo image pair and ground truth depth map are given in Figure 7.2 where the depth is represented in grayscale form 0 to 255 [86]. Notice that there are three depth levels (foreground object, background, and reference plane of display) in the depth map. Segmented regions or objects that are closer to camera are displayed with larger grayscale values. Given this, the user then specifies a disparity input to output mapping. Our program automatically generates a morphed disparity map based on this user specified mapping. Here, we give some simple examples that illustrate: (a) enhancing the stereo effect for a certain range of disparities of the foreground object (b) decreasing the stereo effect for a certain range of disparities of the foreground object (c) enhancing the stereo effect of the background (d) decreasing the stereo effect of the background. The results are shown in Figure 7.3 and 7.4. The disparity input-output mapping curves are as shown at the left in three figures; the morphed disparity maps are shown at the right of these figures. Based on these simple examples, we discuss in section 7.3 how to generate a pair of stereo images according to these morphed disparity map and original stereo image data.

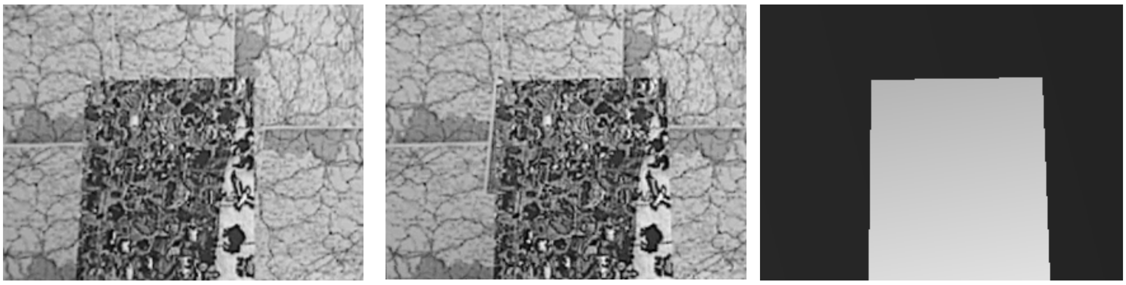


Figure 7.2 Sample stereo pair and its depth map. From left to the right are: left image, right image, the disparity map for the right image. The camera setup is off-axis, meaning that the optical axes of the left and right cameras are parallel.

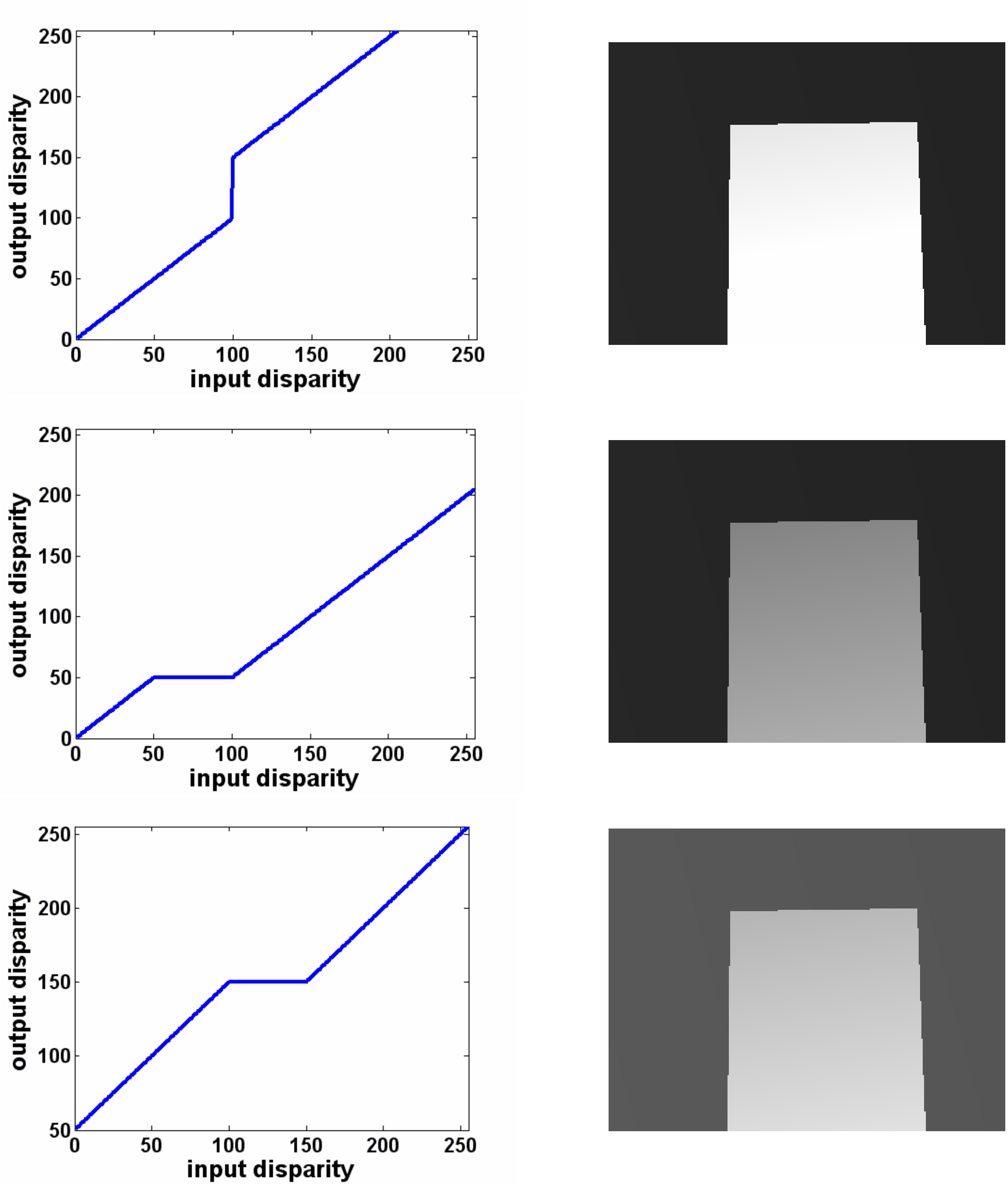


Figure 7.3 Examples of disparity mapping curve and morphed disparity map. Images at the left side show the input and output disparity mapping curves; Images at the right side are the morphed disparity map for the right image of the stereo pair shown in Figure 7.2.

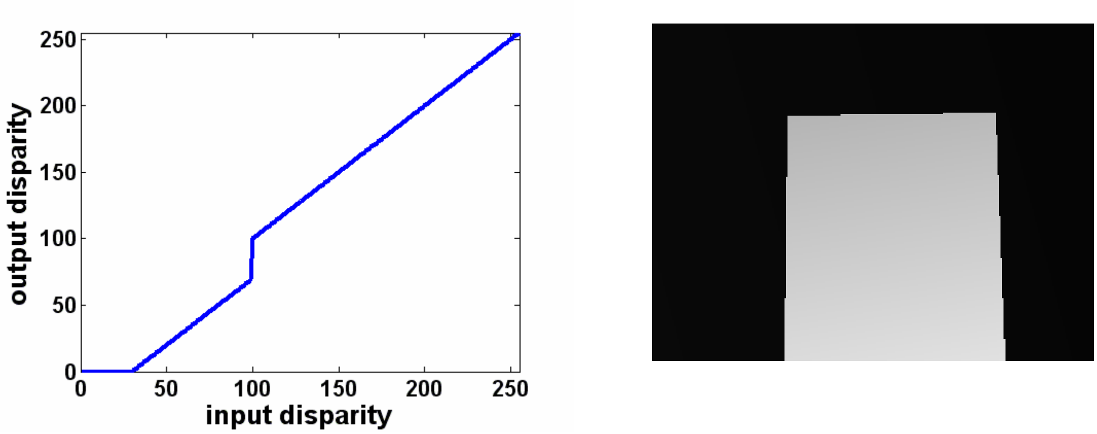


Figure 7.4 Example of disparity mapping curve and morphed disparity map. Image at the left side shows the input and output disparity mapping curves; Image at the right side is the morphed disparity map for the right image of the stereo pair shown in Figure 7.2.

7.2.2 3D surface user interactive interface

In computer graphics, surface representations fall into two major categories: explicit surfaces and implicit surface representations.

Implicit surface representations define a surface with mathematical functions such as multidimensional splines. NURBS (Non-Uniform Rational B-Splines) are a standard modeling method [77]. The functions are defined by values on a control mesh. The user then directly manipulates the surface by moving the control points. Working with NURBS requires much expertise and often time consuming. Also, it is difficult to describe/ manipulate a real world object (especially with sharp edges) with mathematical functions having only a few degrees of freedom.

Polygon mesh surfaces, one of most often used explicit surface representations, is generally composed of polygons or triangles. Rendering efficiency is the advantage of polygon meshes, and most graphic cards have built-in support for polygon meshes. In this technique, we build a 3D surface model similar to those described in [73][12][93]. We use concepts based on the given disparity map to give user a simple and direct control of disparity morphing.

Here, we denote the input given disparity map by $d_{in}(x,y)$, and the output disparity map by $d_{out}(x,y)$. First, we partition the given disparity map $d_{in}(x,y)$ into triangle meshs as shown in Figure 7.5. We first overlay a grid with (x,y) pitch of $k \times k$ pixels on the top of the disparity map $d_{in}(x,y)$. We then divide each patch into two triangles by connecting two diagonal vertices. Assume $\mathbf{v}_1 = (x_1, y_1, d_{in}(x_1, y_1))$, $\mathbf{v}_2 = (x_2, y_2, d_{in}(x_2, y_2))$, $\mathbf{v}_3 = (x_3, y_3, d_{in}(x_3, y_3))$ are three vertices of a triangle. Using these three vertices, we compute an equation of a plane intersecting the three vertices as

$$\overrightarrow{\mathbf{v}\mathbf{v}_1} \bullet (\overrightarrow{\mathbf{v}_1\mathbf{v}_2} \times \overrightarrow{\mathbf{v}_1\mathbf{v}_3}) = 0 . \qquad (7.\ 1)$$

Note that $\mathbf{v} = (x, y, \hat{d}_{out}(x,y))$ represents a point in the plane and $\hat{d}_{out}(x,y)$ is the estimated disparity value at point (x,y) using Eq. (7.1).

Then, we recursively subdivide the triangle into four small triangles by connecting the midpoints of three edges as shown in Figure 7.4, according to a set of subdivision rules.

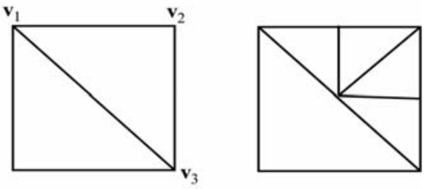


Figure 7.5 Subdivide triangular mesh.

A triangle is split when the mean square error

$$E = \frac{\sum_{(x,y)\in block} [d_{in}(x,y) - \hat{d}_{out}(x,y)]^2}{N} \tag{7.2}$$

between the real disparity value $d_{in}(x,y)$ and our triangular mesh approximation $\hat{d}_{out}(x,y)$ is larger than a certain threshold. Here, $\hat{d}_{out}(x,y) = ax + by + c$ is the estimated disparity value, $d_{in}(x,y)$ is the disparity value from the given disparity map, and N is total number of pixels in a block. The parameters a,b,c are calculated from Eq. (7.1).

To adjust the disparity, the user selects the anchor (fixed) points and the handles. The anchor is composed of all vertices of the triangle mesh that remained unmoved; the handle is composed of all vertices that move with a user-defined displacement. As described in [93], the Laplacian Coordinate (Differential Coordinate) of a vertex $\mathbf{v_i}$ is defined as

$$L(\mathbf{v}_i) = \mathbf{v}_i - \sum_{j\in N(\mathbf{v}_i)} w_{ij}\mathbf{v}_j \tag{7.3}$$

where $N(\mathbf{v}_j)$ is neighbor of $\mathbf{v}_j$, and $w_{ij} = \cot\alpha_{ij} + \cot\beta_{ij}$. α_{ij}, β_{ij} are the two angles opposite edge (i, j).

Let $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2 ... \mathbf{v}_n)$ be a list of mesh vertices. We can find new mesh vertex positions $\mathbf{V}' = (\mathbf{v}_1, \mathbf{v}_2 ... \mathbf{v}_n)'$ by solving minimum least square equations

$$\tilde{\mathbf{V}}' = \min\left(\left\| L(\mathbf{V}') - L(\mathbf{V}) \right\|^2 + \sum_{\mathbf{V}_k \in constrain} \left\| \mathbf{v'}_k - \mathbf{u}_k \right\|^2 \right) \tag{7.4}$$

Here $\mathbf{v}_k$ denotes the mesh vertices in the constrained areas (include anchors and handles), and $\mathbf{u}_k$ is the constrained position of $\mathbf{v}_k$ that are already known after user adjustment. This is equivalent to solving a sparse linear system

$$\mathbf{AV}' = \mathbf{b} \tag{7.5}$$

in least squares sense, where **A** is the Laplacian operator with handle position information and **b** is a vector consisting of constraint information. Also, is generally solved using the equation

$$\mathbf{A}^{\mathbf{T}}\mathbf{AV}' = \mathbf{A}^{\mathbf{T}}\mathbf{b}\,. \tag{7.6}$$

Figure 7.6 gives an example of our 3D disparity map.

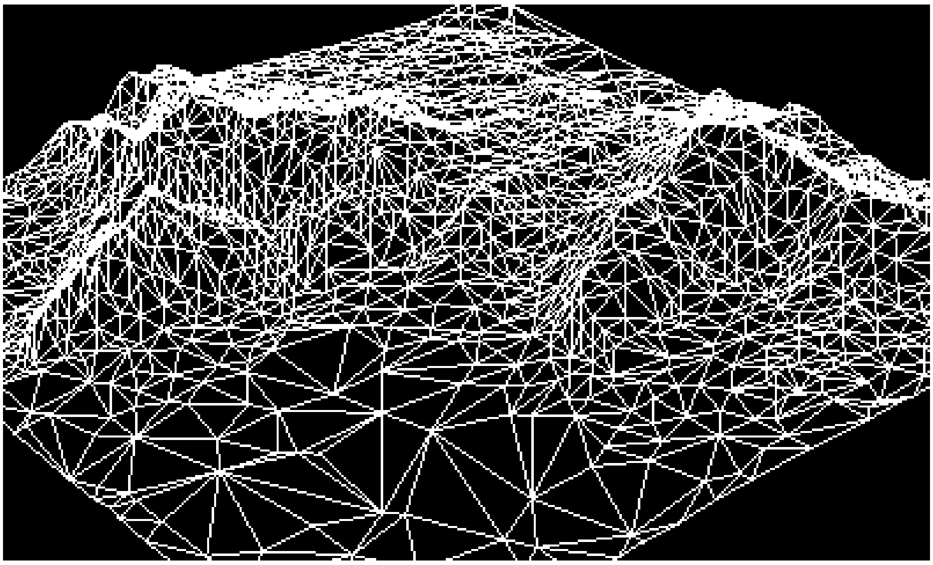


Figure 7.6 Example of 3D disparity surface mesh.

Figure 7.7 also shows a one-dimensional example of Laplacian surface editing. First, the user selects anchors at position $(1, 2)$, $(2, 3)$, $(11, 1)$, and handles at position $(5, 5)$,

$(6, 7)$, $(7, 8)$. Then, the user move the handles to position $(5, 4)$, $(6, 6)$, $(7, 7.5)$. We apply the method mentioned above and calculate the new curve.

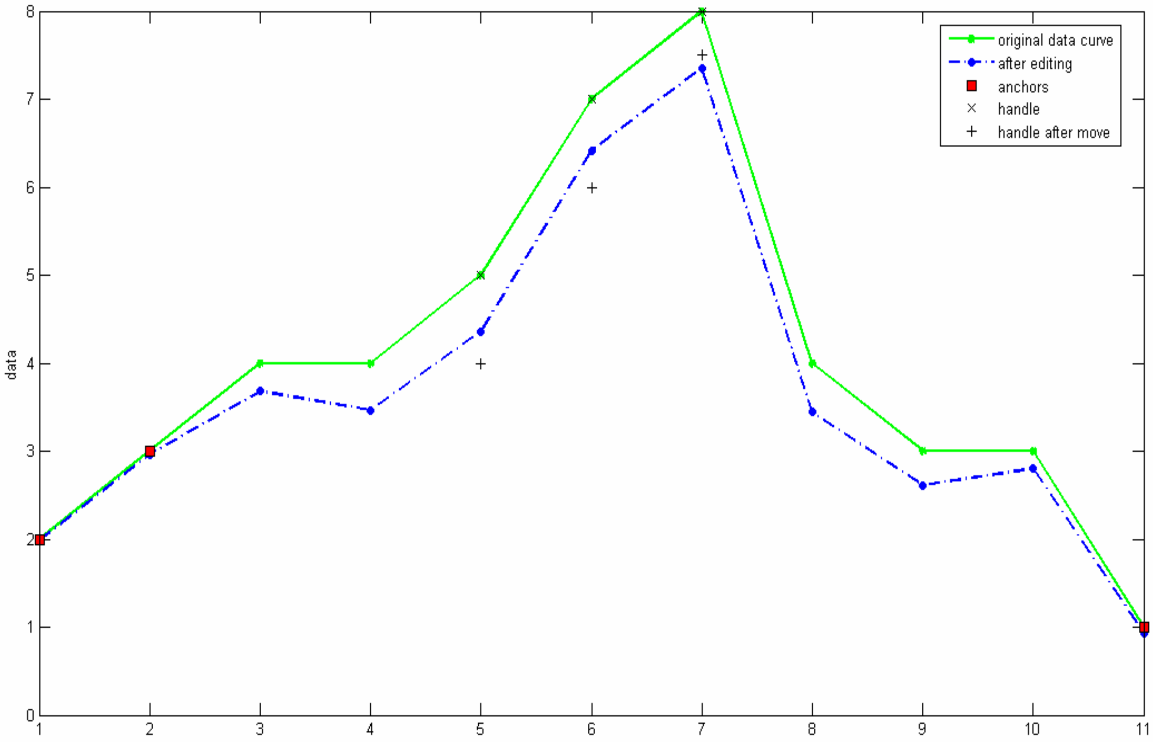


Figure 7.7 Example of 1D Laplacian surface editing.

7.2.3 Off-axis and toe-in disparity map conversion

In this section, we discuss two different camera setups to capture stereo pairs: off-axis and toe-in [71]. We derive relations between points in 3D scene and their projections on image planes based on the geometry model shown in Figure 7.8. Once we morph the disparity map, the disparity based image rendering described in later sections may be used to create new stereo pairs describing views form specified camera setups.

The optical center of the left and right cameras are located at $(-B/2, 0, 0)$ and $(B/2, 0, 0)$. A 3D point $(X, 0, Z)$ is projected onto the position $(x_l, 0)$ and $(x_r, 0)$ on the image plane of the left and right camera; it is also projected onto the position $(x_c, 0)$ on the image plane of the virtual central camera.

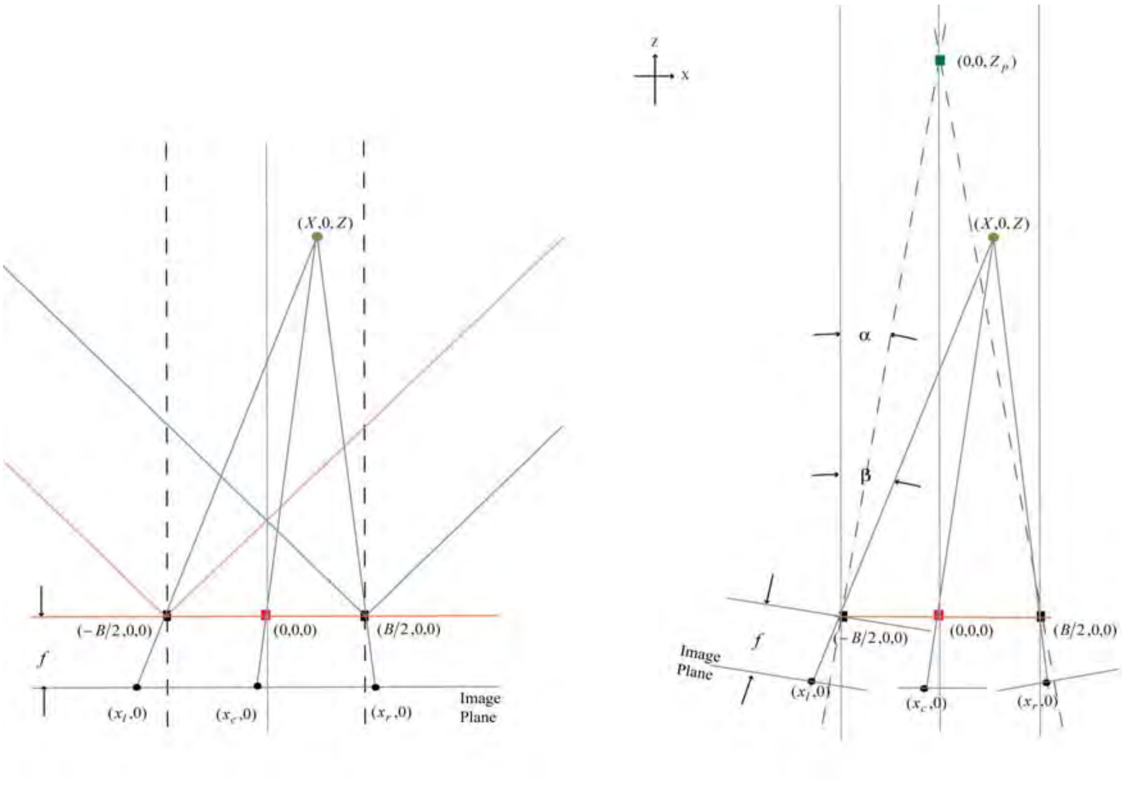


Figure 7.8 Stereo Camera Configurations.

- **Off-Axis Camera Setup**

In the off-axis camera setup, the optical axes of the two cameras are parallel. We denote the disparity for the off-axis camera setup as d_{off} , which is the difference of the corresponding image position in the image plane of the left and right camera. From above geometry model shown on the left of Figure 7.8, we obtain

$$x_c = -f\,\frac{X}{Z} \tag{7.7}$$

$$x_l = -f\,\frac{(X - B/2)}{X} = x_c - f\,\frac{B}{2Z} \tag{7.8}$$

$$x_r = f\,\frac{(B/2-X)}{2X} = x_c + f\,\frac{B}{2Z} \tag{7.9}$$

$$d_{off} = x_r - x_l = f\,\frac{B}{Z} \tag{7.10}$$

- **Toe-In Camera Setup**

In the toe-in camera setup, the optical axis of the left and right camera intercept at a 3D point $(0,0,Z_p)$. We denote the disparity for the off-axis camera setup as d_{toe}. From the geometric model shown on the right of Figure 7.8, we obtain

$$\tan\alpha = \frac{B}{2Z_p} \tag{7.11}$$

$$\tan\beta = \frac{X + B/2}{Z} \tag{7.12}$$

$$\tan\gamma = \frac{B/2 - X}{Z} \tag{7.13}$$

$$x_l = -f\,\tan(\beta-\alpha) = -f\,\frac{\tan\beta - \tan\alpha}{1+\tan\beta\tan\alpha} \tag{7.14}$$

$$x_r = -f\,\tan(\alpha-\gamma) = -f\,\frac{\tan\alpha - \tan\gamma}{1+\tan\gamma\tan\alpha} \tag{7.15}$$

$$x_{avg} = x_l + x_r = f\left(\frac{(\tan\gamma - \tan\beta)(\tan\alpha^2 - 1)}{1+C_1}\right) = -2\frac{fX}{Z}\left(\frac{(\tan\alpha^2-1)}{1+C_1}\right) = C_4 x_c \tag{7.16}$$

$$\begin{aligned} d_{toe} = x_r - x_l = {} & f\left(\frac{-\tan\alpha+\tan\gamma}{1+\tan\gamma\tan\alpha} - \frac{\tan\beta-\tan\alpha}{1+\tan\beta\tan\alpha}\right) \\ & = f\left(\frac{2\tan\alpha(\tan\gamma\tan\beta-1)+(\tan\gamma+\tan\beta)(1-\tan\alpha^2)}{1+\tan\alpha(\tan\gamma+\tan\beta)+\tan\alpha^2\tan\gamma\tan\beta}\right) \\ & = f\left(\frac{C_2(\frac{B-4X^2}{4Z^2}-1)+C_3(\frac{B}{Z})}{1+C_1}\right) = f\left(\frac{C_2(\frac{1}{4B}(d_{off}/f)^2-(\frac{x_c}{f})^2-1)+C_3(\frac{d_{off}}{f})}{1+C_1}\right) \\ & \approx C_3 d_{off} - C_2 f(1+(\frac{x_c}{f})^2) \end{aligned} \tag{7.17}$$

Here, we define

$$C_1 = \tan\alpha(\frac{B}{Z}) + \tan\alpha^2(\frac{B^2/4 - x^2}{Z^2}) \tag{7.18}$$

$$C_2 = 2\tan\alpha \tag{7.19}$$

$$C_3 = (1-\tan\alpha^2) \tag{7.20}$$

$$C_4 = \frac{-2C_3}{1+C_1} \tag{7.21}$$

From Eq. (7.16) and Eq. (7.17), we generate the d_{toe} map given d_{off} and x_c; or the d_{off} map given d_{toe} and x_{avg}.

Figure 7.9 shows an example of simulated disparity maps for both camera setups (off-axis and toe-in) as a function of X and Z. In the off-axis setup, the disparity is inversely proportional to the distance from the camera plane and is not influenced by the X coordinates as given in Eq. (7.10). The disparity in the toe-in setup is influenced both by X and Z coordinates as given in Eq. (7.17). Figure 7.10 plots the disparity for both setups for $x = 0$ as a function of Z. Figure 7.11 shows a simulated disparity map of a toe-in camera setup based on a disparity map captured by an off-axis camera setup.

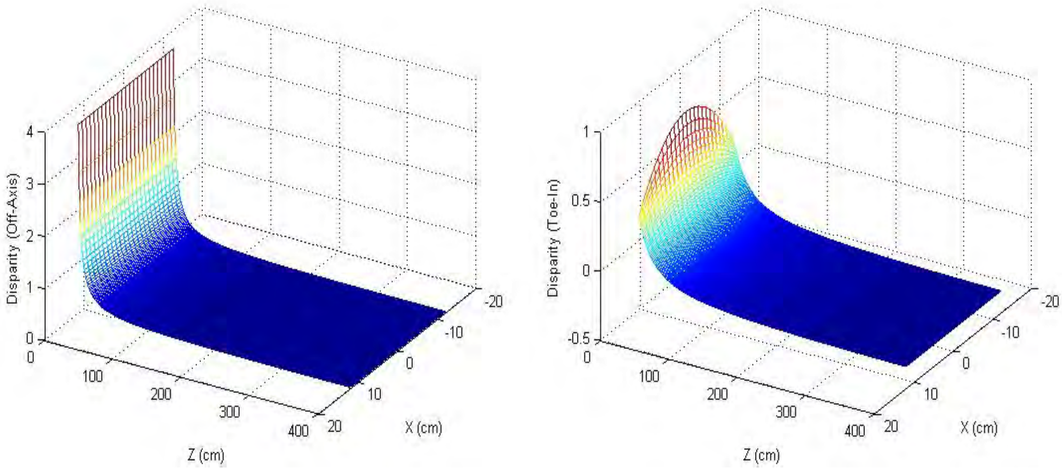


Figure 7.9 Simulated disparity maps for the off-axis (left) and toe-in (right) camera setups. The optical centers of the left and right, respectively are at $-B/2$=-3.25 cm and $B/2$=3.25 cm.

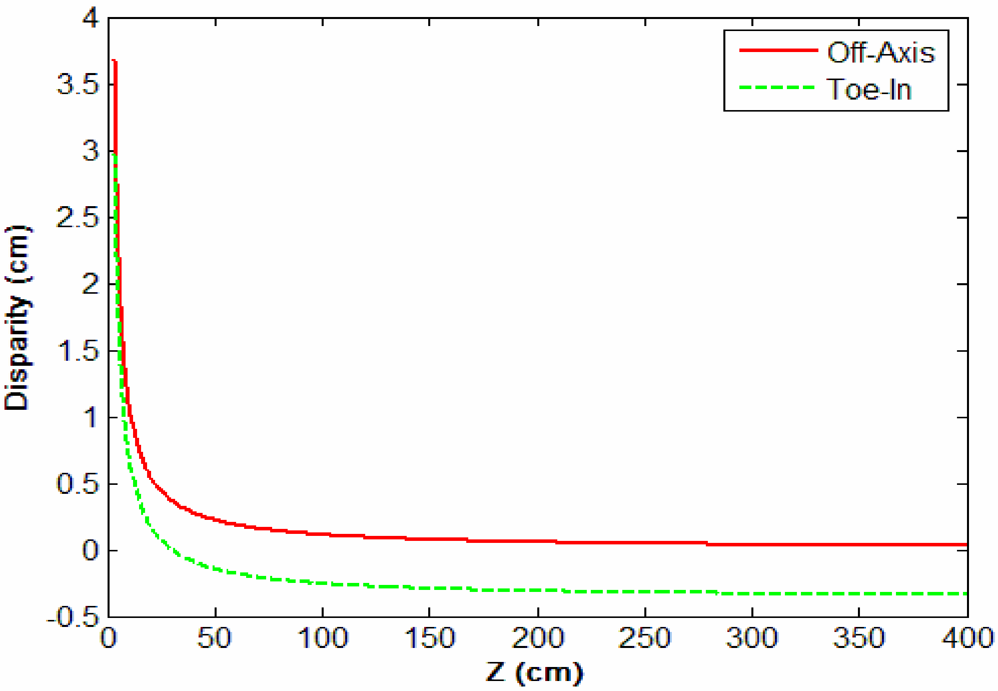


Figure 7.10 Disparity plot at $x = 0$ position for the off-axis and toe-in camera setups as a function of object distance Z, with parameters $f = 1.7$ cm, $B = 6.5$ cm, and $Z_p = 30$.

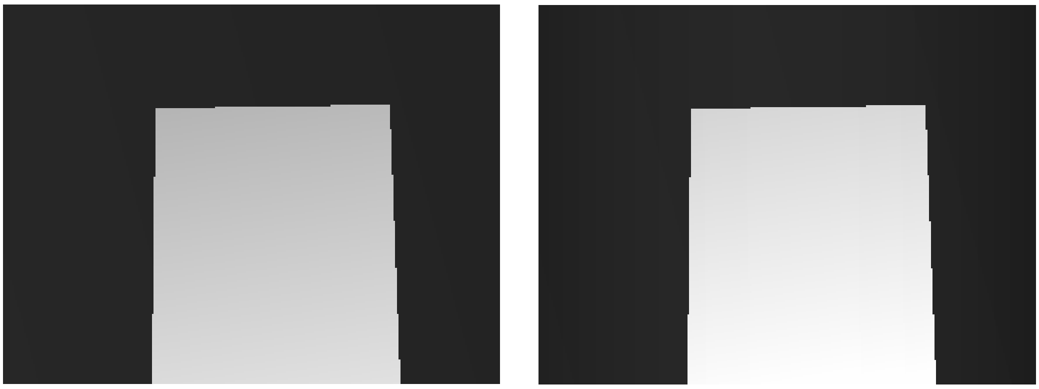


Figure 7.11 Disparity morphing result. The left image is original disparity map from an off-axis camera setup, and the right image is the generated disparity map for the toe-in camera setup with our conversion formula Eq. (7.17).

7.3 Stereo Image Pair Synthesis

Based on the given original data and morphed disparity map, we can generate new stereo image pairs using our stereo image synthesis method. Three steps in our stereo image synthesis process: image warping, data filling, and disparity map smoothing.

7.3.1 Image Warping

Consider the off-axis camera setup (left side of Figure 7.8) for generating virtual stereoscopic images from one single image (input data is monoscopic image/video) or from a stereo image pair (input data is a stereo image pair/ video). In this camera setup, the vertical coordinates of the projection of any 3D point on left and right image plane are the same. From the geometry shown in Figure 7.8, given a monoscopic image captured by the left/right camera viewpoint, we generate the right/left view by copying pixels from coordinate x_l / x_r to a new coordinate $\tilde{x}_r / \tilde{x}_l$ using Eq. (7.22) and Eq. (7.23). Similarly, when the data is captured from the center camera viewpoint, located at the midpoint of the baseline between the left and right cameras, the right and left views can be generated by copying pixels from coordinate x_c to new coordinate $\tilde{x}_r$ and $\tilde{x}_l$ using Eq. (7.24) and Eq. (7.25).

$$\tilde{x}_r = x_l + f\frac{B}{Z} \tag{7.22}$$

$$\tilde{x}_l = x_r - f\frac{B}{Z} \tag{7.23}$$

$$\tilde{x}_r = x_c + f\frac{B}{2Z} \tag{7.24}$$

$$\tilde{x}_l = x_c - f\frac{B}{2Z} \tag{7.25}$$

Eq. (7.22)-Eq. (7.25) are derived from Eq. (7.8). Note that x_c , x_l , x_r are point coordinates in the respectively images captured by the center, left, and right cameras. $\tilde{x}_r$ and $\tilde{x}_l$ are the estimated point coordinates in the left and right images. f is the focal length, B is the length of the baseline, and Z is the distance which is calculated from the disparity map and minimum and maximum scene distance.

For different camera setup data, different views can be synthesized given the geometry of the camera setup and camera matrix. Detailed data interpolation and image warping techniques can be found in [116].

Figure 7.12 shows an example of image warping in which we create a synthetic right image given a disparity map. Notice that there are empty regions.

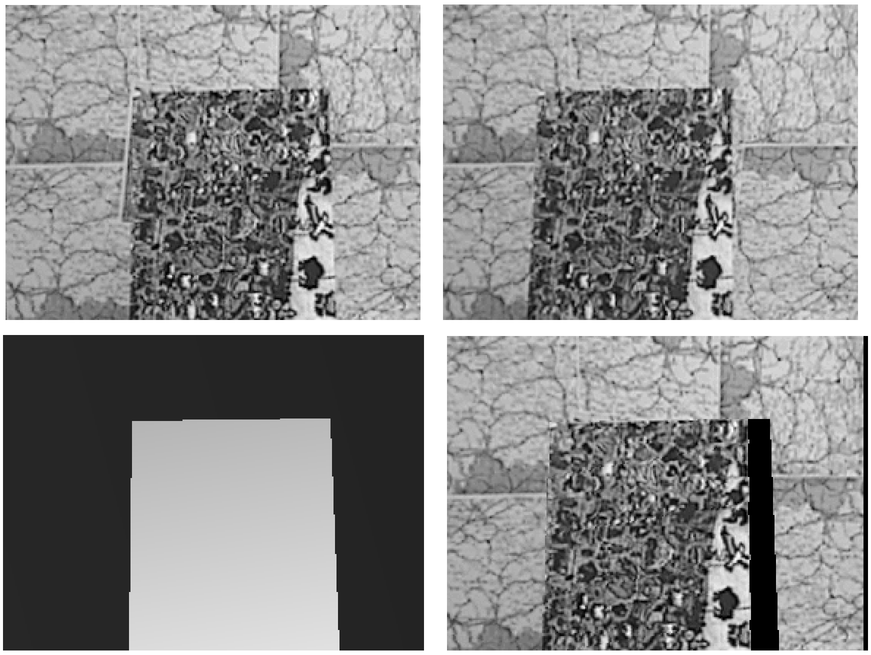


Figure 7.12 Disparity based Image rendering example. Top left and right are original left and right images. Bottom left is the original disparity map displayed relative to the left image. Bottom right is the synthetic right image after image warping process without any data filling algorithm.

7.4 Data Filling

As shown in the previous section, there are generally empty regions in any synthesized image after performing the image warping process. These regions correspond to occlusions in the original left/right image, thus we can fill in missing data given other view containing image data located behind the occlusion. Figure 7.13 shows an example

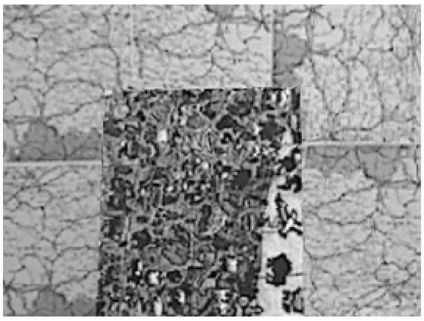


Figure 7.13 Example of synthetic right image with filled occlusions.

However, without this ground truth image data, the empty regions cannot be filled. One method to solve this problem is to smooth the morphed disparity map using an asymmetric Gaussian filter [120] having different horizontal and vertical smoothing scales. The asymmetric nature of the filter reduces the amount of geometric distortion that might be perceived otherwise, by reducing the sharp disparity changes at object boundaries. Assuming the disparity of pixel (x, y) is $d(x, y)$, the disparity after processing is

$$d'(x,y) = \frac{\sum_{v=-\frac{w_v}{2}}^{\frac{w_v}{2}} \left\{ \sum_{u=-\frac{w_u}{2}}^{\frac{w_u}{2}} (d(x-u, y-v)g(u,\sigma_u))g(v,\sigma_v) \right\}}{\sum_{v=-\frac{w_v}{2}}^{\frac{w_v}{2}} \left\{ \sum_{u=-\frac{w_u}{2}}^{\frac{w_u}{2}} g(u,\sigma_u)g(v,\sigma_v) \right\}} \tag{7.26}$$

where $g(u,\sigma_v)$ and $g(v,\sigma_v)$ are the horizontal and vertical Gaussian filter.

7.5 Disparity Map Smoothing

We describe here a set of techniques that may improve the visual quality of stereo pair images whose disparity fields have been altered by the techniques we described in Sec. 7.3. Assume we generate a right image from a morphed disparity map with given an image taken by the left camera (the left image). The corresponding occlusions are areas that can be seen in the right image but cannot be seen in the left image. In these situations, we may not have enough data except the given ground truth occlusions to fill in empty regions in the synthetic image that created by the image warping method based on the modified disparity map. To explain the process, we consider four different examples containing one foreground object plane and a textured background object plane in the scene as shown in Figure 7.14.

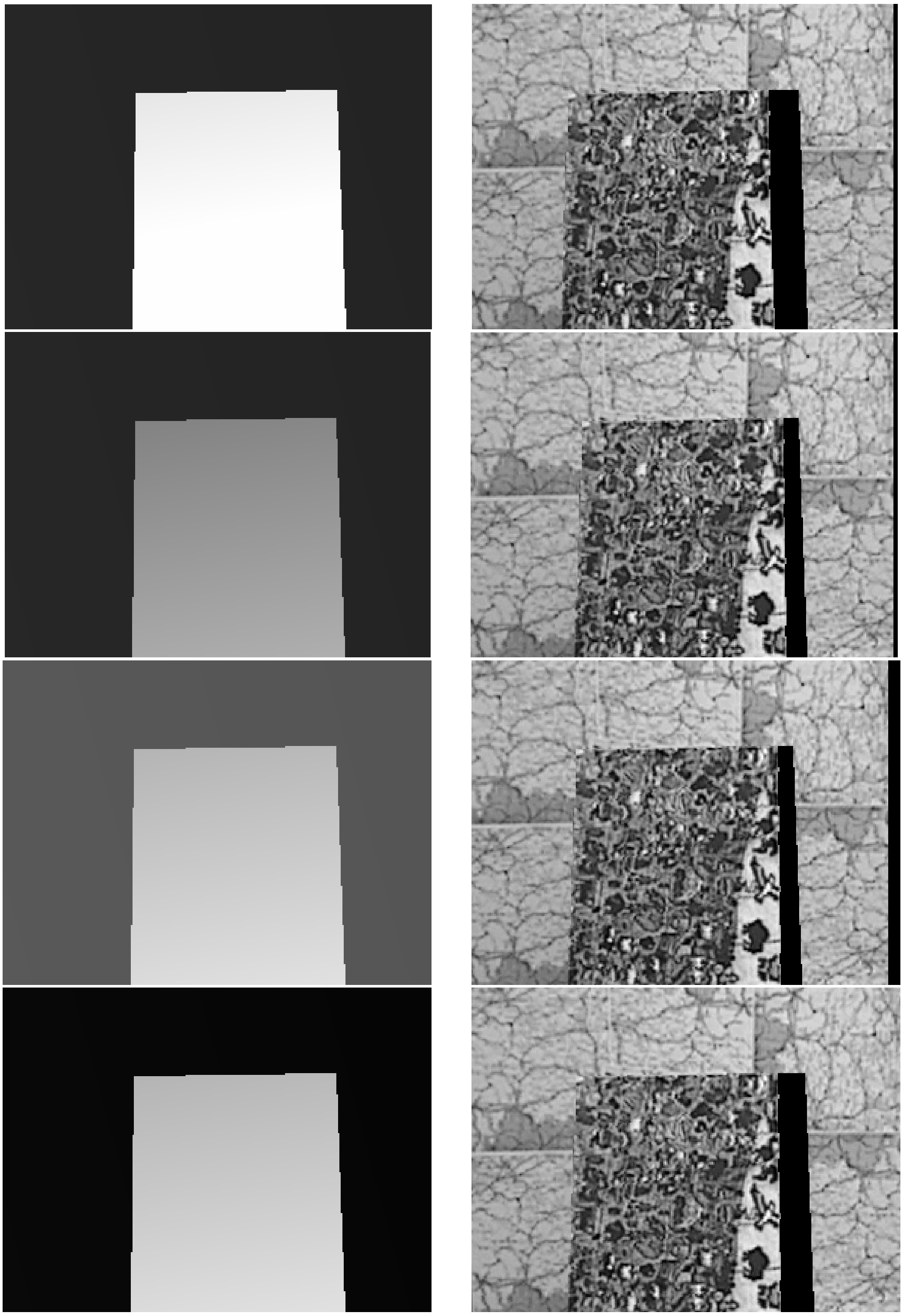


Figure 7.14 Results of synthetic right images created from a single left image after image warping (without data-filling) based on morphed disparity maps. Top to the bottom: foreground enhanced, foreground decreased, background enhanced, background decreased. Left column: morphed disparity map; Right column: synthetic right images without data filling.

The four examples are: (a) enhancing the stereo effect for a certain range of disparities of the foreground object (b) decreasing the stereo effect for a certain range of disparities of the foreground object (c) enhancing the stereo effect of the background (d) decreasing the stereo effect of the background. Comparing the empty regions with original empty regions in the right bottom image of Figure 7.12, we notice that lager empty regions appear in case (a), (c), (d). A method of filling the missing data is needed to overcome this problem.

Given the disparity map, we compute the discontinuity in the disparity map at the boundary between background and foreground. In effect, we measure the height of the disparity map edge discontinuity. If this discontinuity becomes bigger after morphing the disparity map, we perform an edge smoothing process on the disparity map to soften the perceived sharp height transition between background and foreground. If the foreground object's disparity increases δ pixels, we blend the boundary layer between the foreground and background of the morphed disparity map with that of the original disparity map using a disparity smoothing filter. First, we consider a smoothing filter as in Eq. (7.27).

$$d_p(x) = (1-\alpha(\tau))d_n(x) + \alpha(\tau)d_o(x) \tag{7.27}$$

Here, d_o is the original disparity; d_n is the morphed disparity; d_p is our final disparity value. τ is the absolute distance to the edge and always positive. $\alpha(\tau) = e^{-\frac{\tau}{\beta}}$.

Figure 7.15 shows an illustration of applying Eq. (7.27) on the four examples mentioned before: (a) moving the foreground object closer (increasing the positive disparity: $+\delta$

pixels) to the camera, (b) moving the foreground object away from the camera (reducing the positive disparity: -δ pixels), (c) moving the background object closer to the camera, and (d) moving the background object away from the camera.

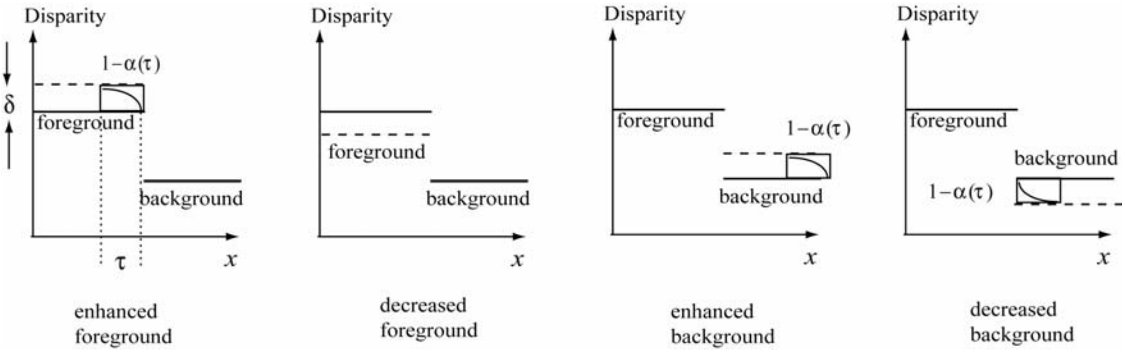


Figure 7.15 Illustration of disparity morphing and discontinuity smoothing. The dash line represents the new disparity level after morphing; the solid line is the original disparity level. Notice that three disparity levels (foreground, background, and reference plan of display) at beginning.

Note that in case (b) the empty regions (foreground decreased) of the synthetic right (image at the right side in the third row of Figure 7.14) is smaller than the one generated based on original disparity map, because the height of discontinuity in the morphed disparity map is smaller than that of the original disparity map. Therefore, we do not need to apply edge smoothing in this case. We apply same blending method at the disparity discontinuity (the edge of the background) for case (c), and at the disparity discontinuity (the boundary layer of background object between background and foreground) for case (d). The results of disparity edge smoothing using Eq. (7.27) (morphed disparity maps after processing with disparity edge smoothing) and corresponding synthetic right images which generated by image warping without data filling are shown in Figure 7.16. We perceive that the empty regions in the synthetic right images are kept as same as those in the synthetic right image based on the original disparity map in Figure 7.16, where we can fill in ground truth occlusions later.

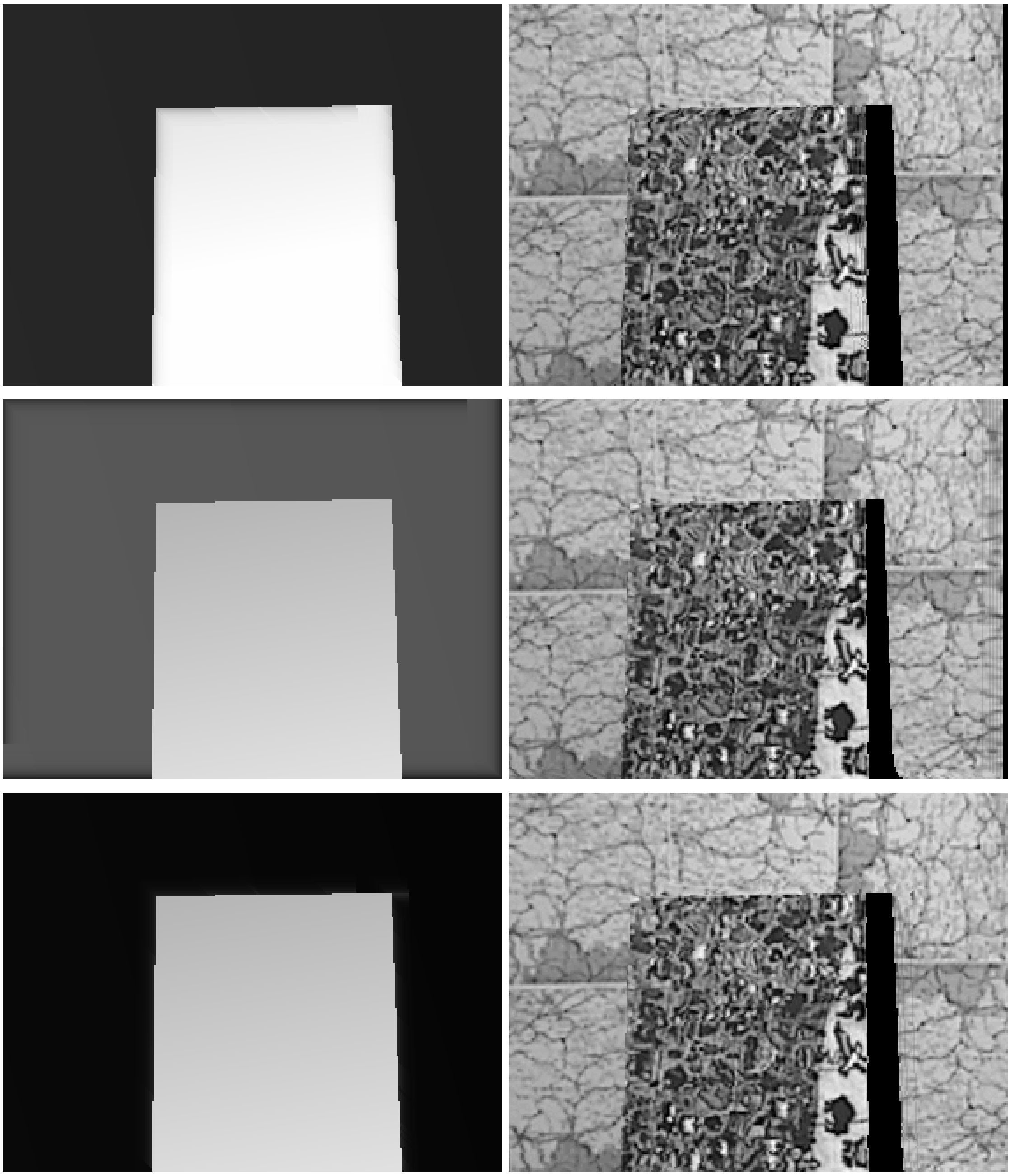


Figure 7.16 Disparity edge smoothing results using Eq. (7.27). ($\delta = 10$, $\beta = 20$)

Top to the bottom: foreground enhanced, background enhanced, background decreased. Left column: morphed disparity maps that are processed with our edge smoothing method; Right column: (without hole-filling) based on processed morphed disparity maps.

Then, we consider another smoothing filter as following equations

$$d_p(x) = \alpha(\tau)d_n(x) + (1-\alpha(\tau))d_o(x)\,. \qquad (7.28)$$

Figure 7.17 shows an illustration of applying this equation on the four examples, and the results of disparity edge smoothing (morphed disparity maps after processing with disparity edge smoothing) and corresponding synthetic right images which generated by image warping without data filling are shown in Figure 7.18.

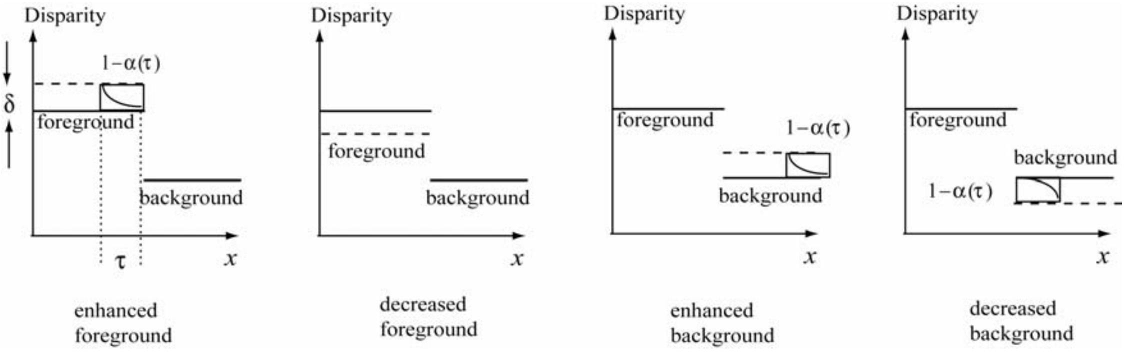


Figure 7.17 Illustration of disparity morphing and discontinuity smoothing. The dash line represents the new disparity level after morphing; the solid line is the original disparity level. Notice that three disparity levels (foreground, background, and reference plan of display) at beginning

Comparing the results in Figure 7.16 and Figure 7.17, we note that the disparity adjustment using Eq. (7.28) creates visible artifacts. This effect can be explained by the two transaction points at edge of the disparity in Figure 7.17, where the derivative of the disparity curve doesn’t even exist. Therefore, we use Eq. (7.27) as our smoothing filter in our disparity edge smoothing method.

We have also tried using a different β (the parameter controls the smoothing extent in Eq. (7.27)), and Figure 7.18 shows the results using $\beta = 10$. It appears in this case that using value $\beta = 10$ gives better balance between the creation of artifacts and data-filling in synthesized images.

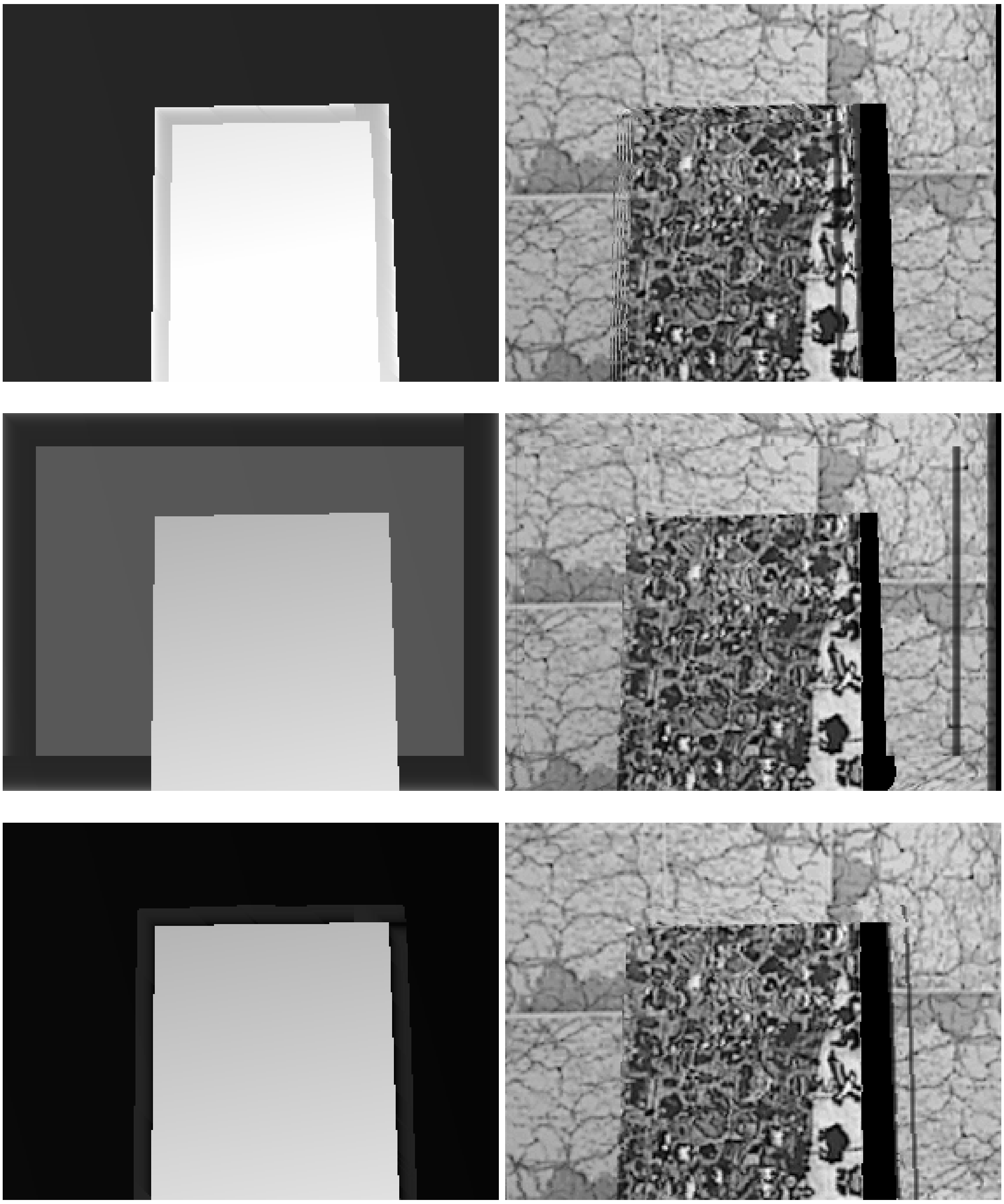


Figure 7.18 Disparity edge smoothing results using Eq. (7.28). ($\delta = 10$, $\beta = 20$)

Top to the bottom: foreground enhanced, background enhanced, background decreased. Left column: morphed disparity maps that are processed with our edge smoothing method; Right column: (without data-filling) based on processed morphed disparity maps.

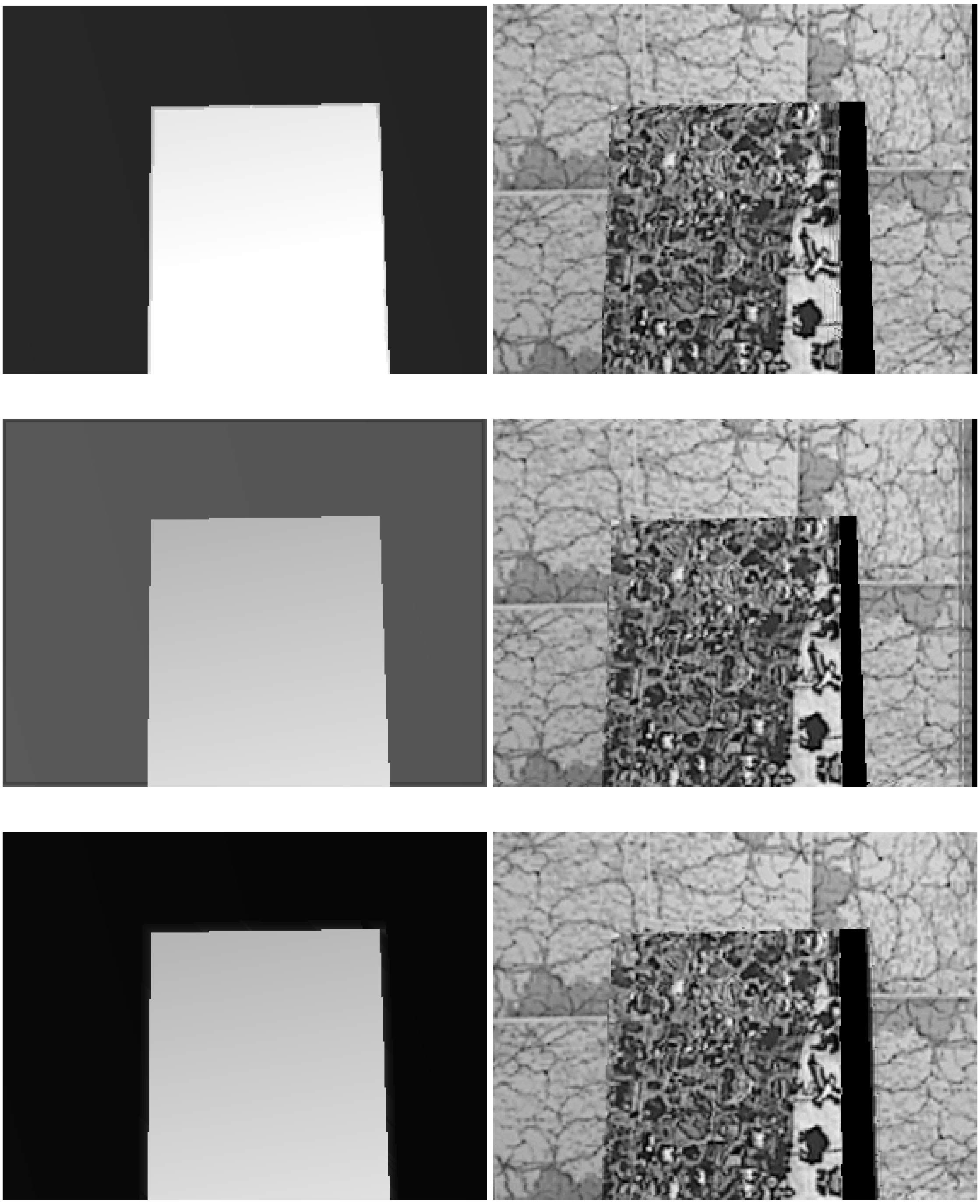


Figure 7.19 Disparity edge smoothing results using Eq. (7.27). ($\delta = 10$, $\beta = 10$)

Top to the bottom: foreground enhanced, background enhanced, background decreased. Left column: morphed disparity maps that are processed with our edge smoothing method; Right column: (without data-filling) based on processed morphed disparity maps.

7.6 Summary

We apply edge smoothing at discontinuities in the disparity map where there is an increase of disparity discontinuity between two regions (e.g. height of the foreground compared to the background or height of the background compared to the zero disparity level). Our procedure is to compare the discontinuity levels of all the boundary areas in the morphed and original disparity map and decide where to apply blending methods for the complex scene. After the blending process, we warp the left image according the final disparity map to create the new left and right images and fill remaining empty regions caused by the disparity discontinuity (gaps) of the original disparity map. The sequence of processes to synthesize a right image on the original disparity map is: Image Warping + Data Filling. The sequence of processes to simulate a right image based on morphed disparity map is: Disparity Edge Smoothing+Image Warping + Data Filling. Figure 7.20 shows an illustration of the process. Figure 7.21 shows results synthetic right image results, and Figure 7.22 shows the synthetic stereo pair results for four cases previously shown in Figure 7.14. Figure 7.23 shows an example for tow-in and off-axis conversion result.

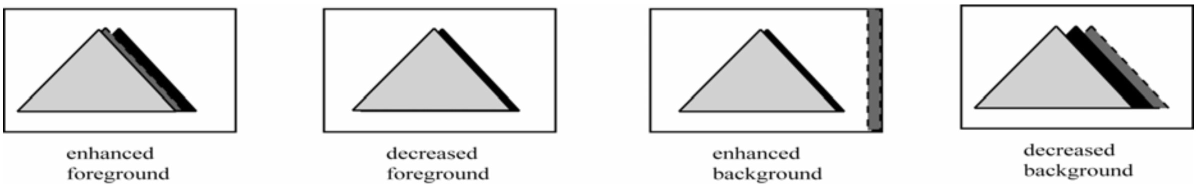
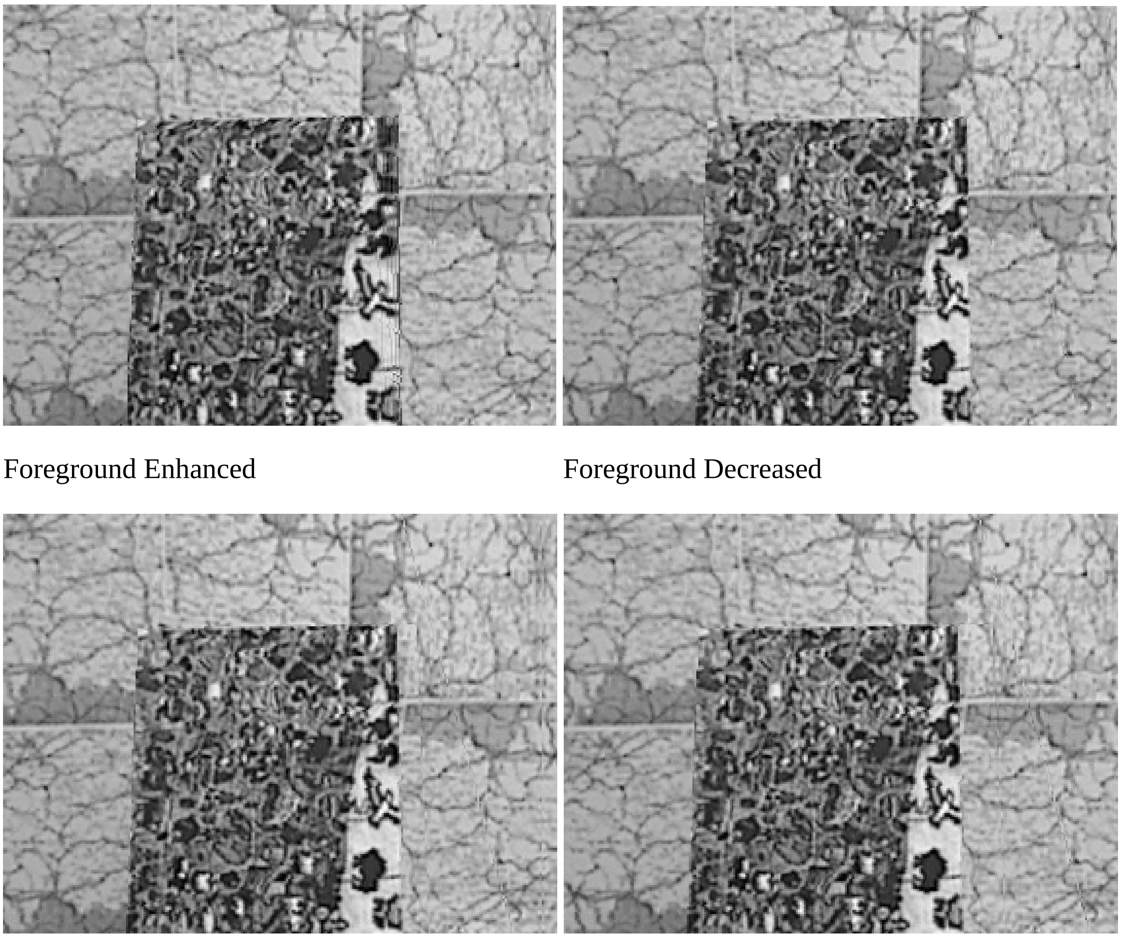


Figure 7.20 Illustration of synthetic right images (without data-filling) based on disparity maps processed by our disparity edge smoothing method. The black areas represent the occlusions. The dark gray areas with dash boundary are the areas where disparity map blending is applied. The light gray areas represent the foreground object.



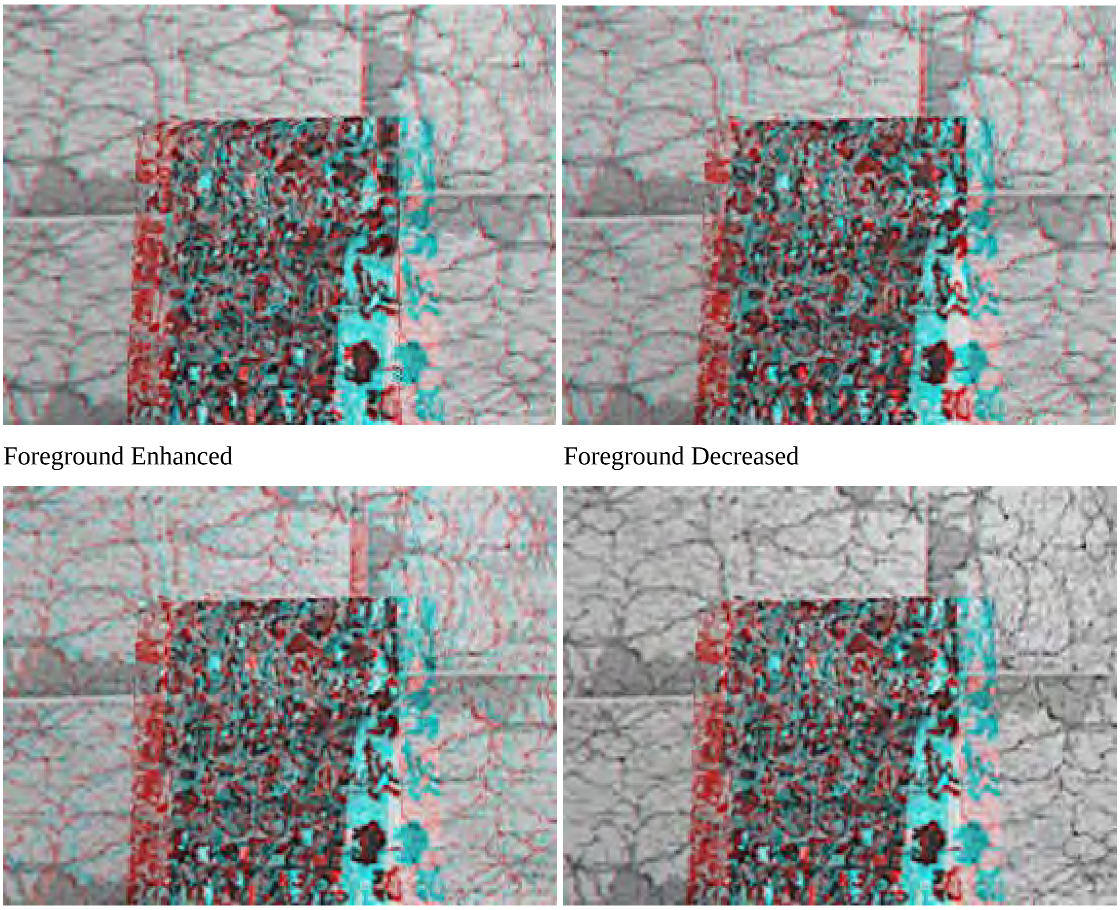
Foreground Enhanced

Foreground Decreased

Background Enhanced

Background Decreased

Figure 7.21 Results of our synthetic right images (with data-filling) based on morphed disparity maps that are processed by disparity edge smoothing. From left to right and top to bottom are: foreground enhanced, foreground decreased, background enhanced, background decreased. ($\delta = 10$, $\beta = 10$)



Foreground Enhanced

Foreground Decreased

Background Enhanced

Background Decreased

Figure 7.22 Results of our synthetic stereo pair in anaglyph form. ($\delta = 10$, $\beta = 10$)

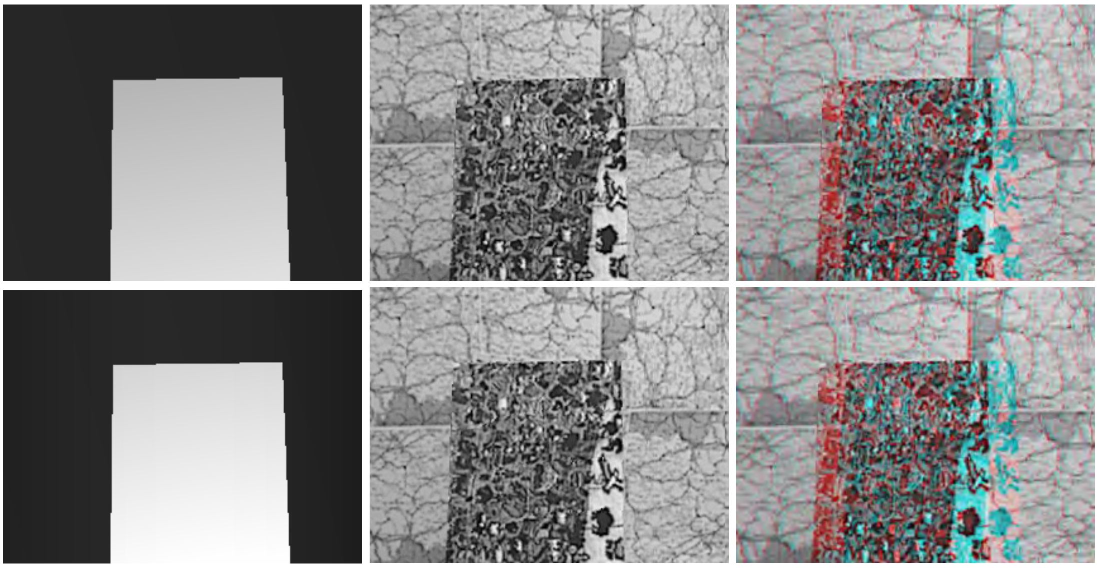


Figure 7.23 Disparity based image rendering results (off-axis to toe-in conversion). Top row shows the original data from an off-axis camera setup. From left to right are disparity map, right image and stereo anaglyph. The second row shows the morphed toe-in camera setup result for $\alpha = 1^\circ$. From left to right are: simulated toe-in disparity map, generated right image from the simulated-disparity using our disparity map based image rendering algorithm, and corresponding stereo anaglyph.

Chapter 8 Virtual Environment and User Interaction

According to Aukstakalnis and Blatner [8], there are three key components of a virtual reality environment:

- A virtual environment is a computer-generated environment, which can provide some kind of realism
- Users should be allowed to interact with the environment in an effective manner
- Users can feel immersed in the environment

In this chapter, a virtual environment is developed, which equips the key components of virtual environment and has following features:

- Renders stereo panorama on different displaying devices (HMD, shutter glasses, sharp auto-stereoscopic display
- Creates a virtual scene which users can walk around and interact with different interaction tools
- Match the coordinate of virtual environment with that of the real world for training application

The following sections describe the techniques of our virtual environment in details and show examples of their uses.

8.1 Our Virtual Environment and User Interaction Overview

The task of our virtual environment can be divided as two categories: navigation and selection/manipulation, and the virtual environment provide the user input with few degrees of freedom and one output (display). Because of the small sweet spot for AS displays, we let user sit in one place. And the task of selection can be achieved by using keyboard or tracking user hand with magnetic tracker [2] or optical tracking device [119]. The system is as shown in Figure 8.1, and the details of each device are listed as follows:

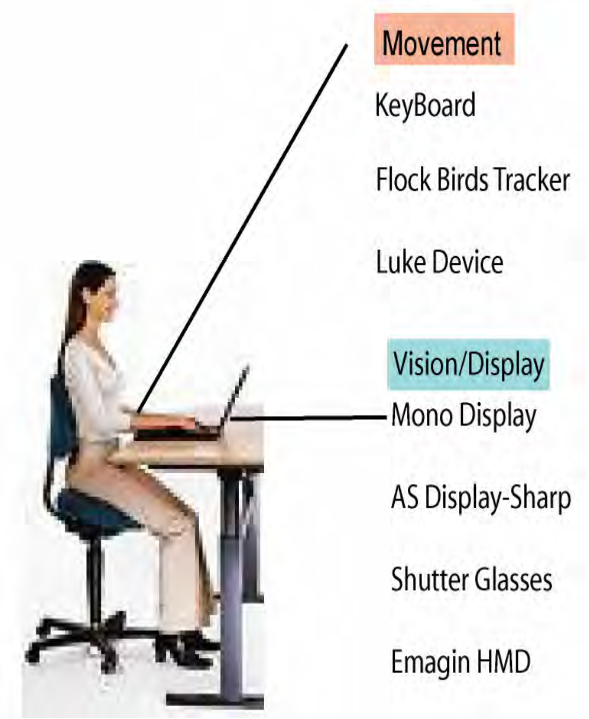


Figure 8.1 Illustration of a personal virtual environment.

- Key Board:

A/L: user move left/ right and forward/backward

U/D: user move up/down

←/→: user rotate view direction clockwise or counter clockwise along z axis(Roll)

↑ / ↓ :user rotate view direction clockwise or counter clockwise along y axis(Yaw)

J/M: user rotate view direction or clockwise or counter clockwise along x axis (Pitch)

- Magnetic Tracking Device (Flock of Birds Tracker) [2]:

The Flock of Birds Tracker (FOB) is a 6 degree of freedom (6DOF) tracker used for tracking 1-4 sensors simultaneously. Each FOB receiver makes up to 144 position and orientation measurements per second. Here, we let the user hold one sensor, and user can change the view direction and position by rotating or moving the sensor. Whichever way you turn, every motion is captured and the data is transferred to the computer through RS232 interface.

However, the FOB's pulsed DC magnetic fields only permeate all non-metallic objects. If in the presence of metallic structures, the measurements of FOB will have a fraction of the distortion errors.

- Optical Tracking Device [119]:

The optical tracking, which uses two cameras to track a light source, can provide 6 degree of freedom. Here, we simulated the optical tracking device’s input as a key input to our VR system to control the user position.

- Mono Display:

We only render one image (left-eye panorama) for mono display.

- AS Display-Sharp:

We use DDD SDK to interpolating two images (left and right eye panorama) as the stereo image that we described in Sec. 2.2 to shown on Sharp AS display.

- Shutter Glasses and CRT display:

Shutter glass contains a polarizing filter which can becomes dark or transparent according to the volage it applied. The monitor alternately displays different perspectives for each eye with the refresh rate. This method is called alternate-frame sequencing. At sufficiently high refresh rates, the user’s visual system does not notice the flickering.

Each eye receives a different image, and the user can perceive the stereo. For enought highest fresh rate (120Hz), we use CRT monitor.

- eMagin HMD:

The eMagin Z800 3DVisor provides 360 degree horizontal field of view with a six degrees of freedom head tracker. It also provides stereovision with hi-fi sound for an immersive experience. It uses superb high-contrast OLED displays and very lightweight. And we use the eMagin Z800 SDK to create a stereo virtual environment and let the user move and look around in the virtual environment by changing his head position.

Figure 8.2 shows the procedure of our user interaction algorithm for stereo panorama virtual environments. First, we let the user stands at the original point. Then our system listens to the key board, FOB tracker or optical tracking devices [119] to see if the user moves/walks in the virtual environment. If the user walks, our software moves the virtual viewer to the corresponding location in virtual space. After that, the corresponding mono/stereo image can be rendered by stereo panorama rendering algorithm (Sec. 8.2) with the help of OpenGL and DDD and shown on the displaying device.

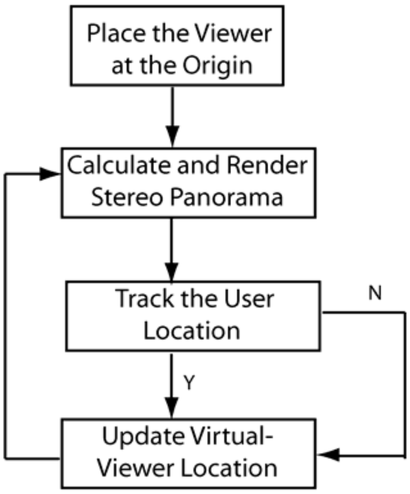

Place the Viewer at the Origin
Calculate and Render Stereo Panorama
Track the User Location
N
Y
Update Virtual-Viewer Location

Figure 8.2 Procedure of our virtual environment user interaction.

8.2 Stereo Panorama Rendering

We propose a way to adaptively adjust the horizontal and vertical parallax for different viewing positions and directions to provide extremely realistic stereo views. The algorithm should give the impression of horizontal/ vertical parallax as Figure 8.3.

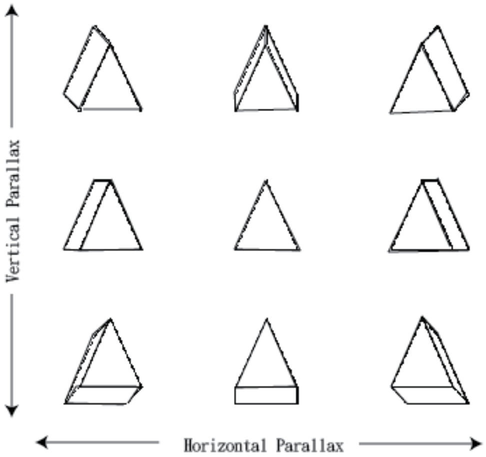


Figure 8.3 Illustration of horizontal and vertical parallax.

We set the horizontal parallax as a function of a viewer's position. When the viewer moves closer, we increase the horizontal parallax by applying algorithm described in Sec. 5.2. The disparity is only measured the first time to mark the closest and most distant scene points to constrain the adjusting range and sampling strip position. After that, we do not measure the disparity each time, but we record the sampled strip position and disparity for fast look up. For vertical parallax, as the viewer moves up, the object below the central horizontal line is adjusted for increased negative vertical parallax, which we could achieve by morphing one image vertically as in Sec. 5.3 to control the vertical parallax. Here, we only describe the parallax adjusting methods. Other view synthesis techniques are still required for these cases [79][33].

8.3 Space Registration

In this section, we create two coordinates as shown in Figure 8.4: 1) world coordinate (orange lines) in which the original is a point that we calibrated beforehand (ex. the center of the keyboard); 2) user coordinate (blue lines) attached to the user's head for tracking or input device.

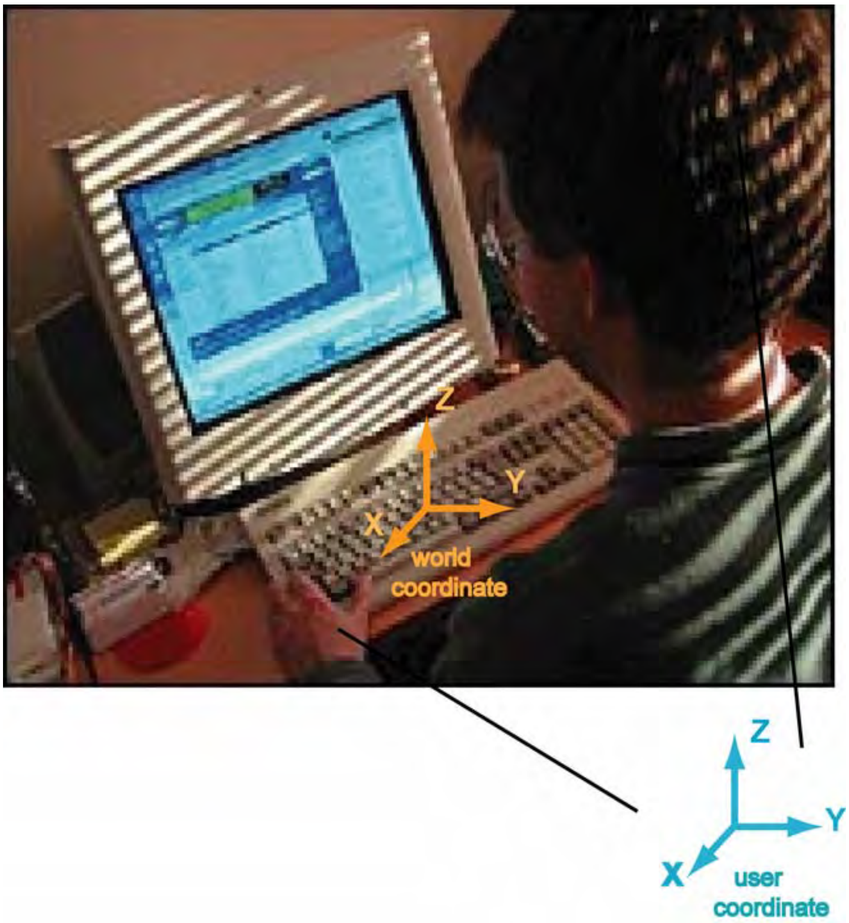


Figure 8.4 Coordinates for space registration.

At the beginning, we use one flock of birds senor to decide the relative position of the user coordinate to the world coordinate, then we can decide the homogenous transformation matrix between two coordinates as ${}^{U}_{W}A$. As the user uses the key board, FOB tracker or optical tracking device to move, the position to the current user coordinate as ${}^{U}P$ can be decided. We update the view by calculating the current user view direction through ${}^{U}_{W}A{*}^{U}P$. Here, ${}^{U}_{W}A = R_\theta R_\alpha R_\beta ST$. θ (Yaw), α (Roll), β(Pitch) are rotation angles around Z, X, Y axis. S is the scaling matrix. T is the matrix used for translation.

$$R_\theta = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8.1)$$

$$R_\alpha = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8.2)$$

$$R_\beta = \begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8.3)$$

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8.4)$$

$$T = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8.5)$$

Chapter 9 User Tests

In this chapter, we design experiments to exam our virtual environment with comparing different displaying and tracking devices.

9.1 Preliminary Experiment Design

First, we list devices with related roles/functions that we plan to experiment as in Table 9.1. Four types of displays (monoscopic display, shutter glasses, AS display, HMD) are considered here. Shutter glasses and AS display provide the perception of stereo whilst monoscopic display does not. Shutter glasses are worn just as a normal pair of eye glasses, therefore allowing the user to move their head freely without losing stereo effect. AS displays do not require the use of glasses, therefore the user must have their head positioned within the sweet spot (a limited zone in order to view the stereo effect on the screen). HMD (head mounted device) devices provide participants either mono view or stereo view and with build-in head tracker to allow a participant moving head and changing view freely.

Table 9.1 List of devices

Display Device	**View Panning Controller**	**Hand Position Controller**
Mono Display	Keyboard	Keyboard
AS Display (Sharp)	Magnetic Tracking Device [2] (FOB)	Magnetic Tracking Device [2] (FOB)
CRT+ Shutter Glasses	Optical Tracking Device [119]	Optical Tracking Device [119]
HMD	HMD Build-in Head Tracker	

Except a common interaction devices, keyboard, we consider two kinds of interaction devices: magnetic tracking system (FOB) [2] and optical tracking device [119]. The magnetic tracking system, a high end motion capturing device, provides information of six degrees of freedom with a sampling rate of 120 Hz. The optical tracking device was developed by S.-C. Yeh [119] and utilizes low-cost dual webcams to track multiple LEDs, providing information of six degrees of freedom with a sampling rate of 60 Hz.

9.1.1 Evaluation Methodology

To have complete evaluations of different displays and interaction devices and consider different influences factors, we not only measure participants' performance (complete rate, complete time, and error rate, etc) while playing VR game, but also measure participants' perception with a series of questionnaires in regard to display comfort, functionality of interaction devices, presence of VR game and system evaluation. Also, we carefully choose the schedule for questionnaires and participants' performance measurement. If a questionnaire is taken at in-appropriate time (too early/late), it may can't reflect the user point of view after they finish the task or we may can't get a lot of useful information because participants may get tired and can't focus on questions. Our evaluation methodology is shown in Figure 9.1.

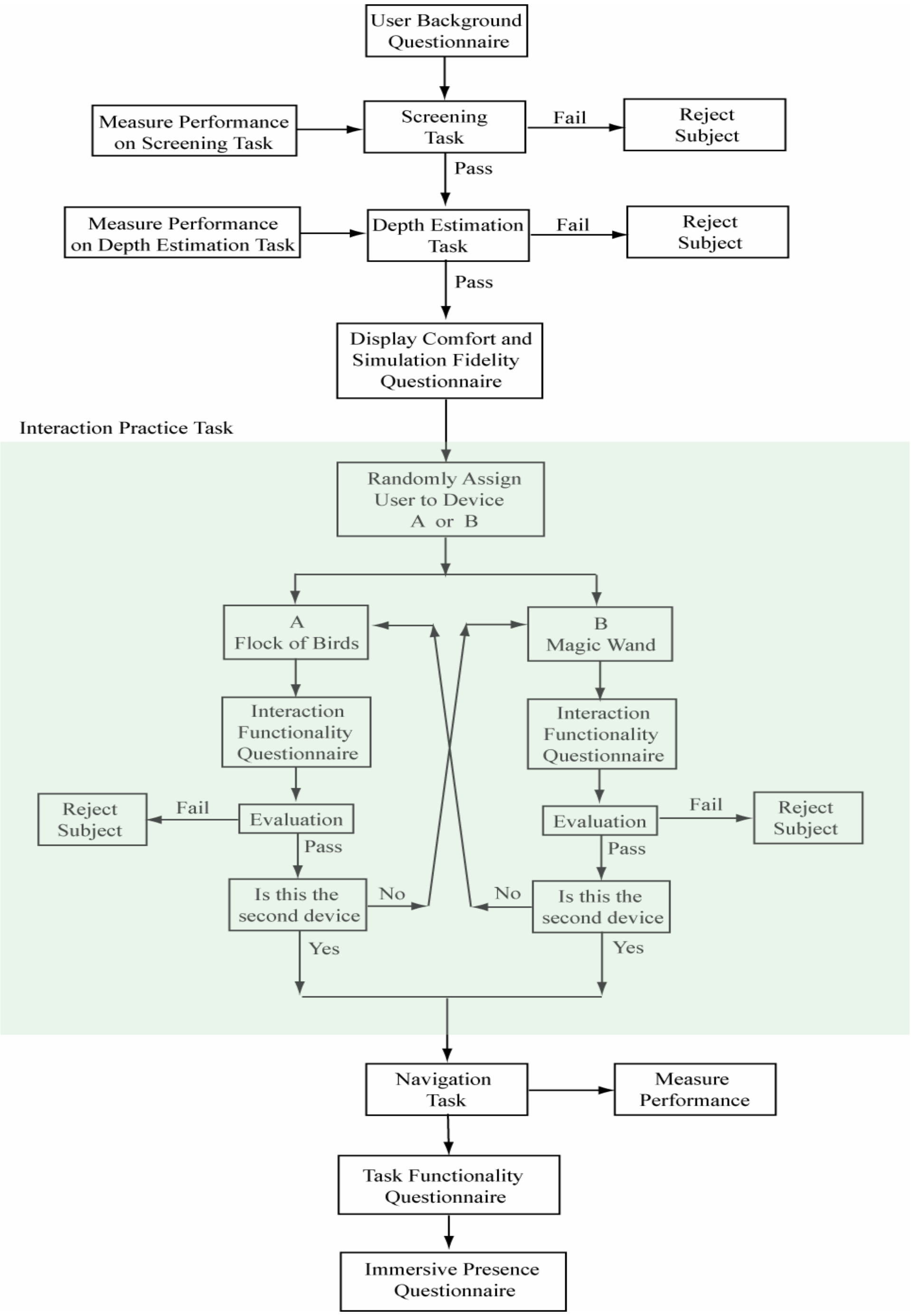

User Background Questionnaire
Measure Performance on Screening Task
Screening Task
Fail
Reject Subject
Pass
Measure Performance on Depth Estimation Task
Depth Estimation Task
Fail
Reject Subject
Pass
Display Comfort and Simulation Fidelity Questionnaire
Interaction Practice Task
Randomly Assign User to Device A or B
A Flock of Birds
B Magic Wand
Interaction Functionality Questionnaire
Interaction Functionality Questionnaire
Reject Subject
Fail
Evaluation
Evaluation
Fail
Reject Subject
Pass
Pass
Is this the second device
No
No
Is this the second device
Yes
Yes
Navigation Task
Measure Performance
Task Functionality Questionnaire
Immersive Presence Questionnaire

Figure 9.1 Evaluation Methodology (preliminary experiment design).

9.1.2 Task Descriptions

In this, we consider four kinds of tasks: screening task, depth estimation task, interaction task, and navigation task. The description of each task is listed in Table 9.2.

Table 9.2 Description of tasks

Tasks	**Description/ Procedure**
Screening	Stereo Test (using 3D anaglyph book), Color Blindness Test, and Visual Acuity Test
Depth Estimation	a. Keep the user view -point at a fixed point in the center (the origin). b. Show the user a well-known reference object at a given distance. c. The user estimates/compare the distance of one specified objects (as in Figure 9.2) in the field of view. d. Ask the user following questions: 1. Is target object closer to you than the reference object? Y/N 2. What's the relative distance from target object to the reference object? (multiple choice) e. Ask the user move the target object forward/backward using keyboard until the target object and reference object are on the same plane.
Interaction	User experiments with various 3D interaction devices. Tests: a. Rotate and translate a geometric shape. b. Test user's ability to manipulate an object with some combination of $x, y, x, \theta_x, \theta_y, \theta_z$. Pass/Fail Criteria: Measure the time they need to finish the tests. If they take too long, fail the subject.
Navigation	a. The computer randomly generates a set of fixed objects with uniform distribution in θ, Z space as in Figure 9.3. b. The user must follow the sequence to reach all the objects. Touch number 1→ Touch number 2→Touch number 3.. Comments: 1. To be fair in comparing different displays, use the same panning angle θ control device. 2. HMD has an advantage in that θ is controllable by head motion and eye look direction. 3. If we need to ask user to back to original point after touch each object? 4. Exist at least three different sequences: numbered balls generated in time

Table 9.2: Continued

	order, all balls in generated in parallel-no numbers, all balls in parallel-with numbers.

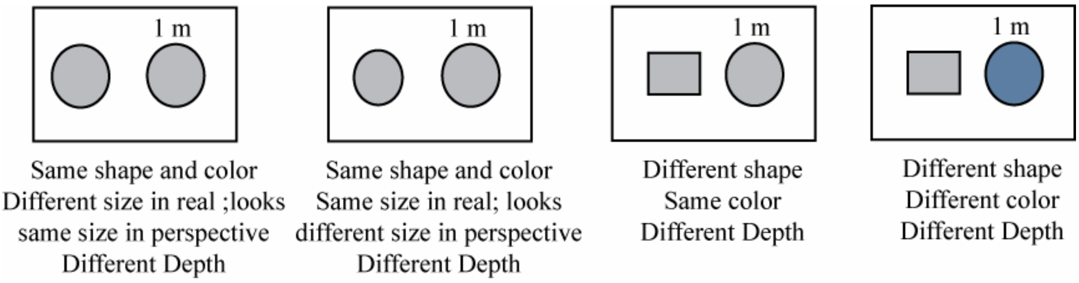


Figure 9.2 Example of depth estimation task.

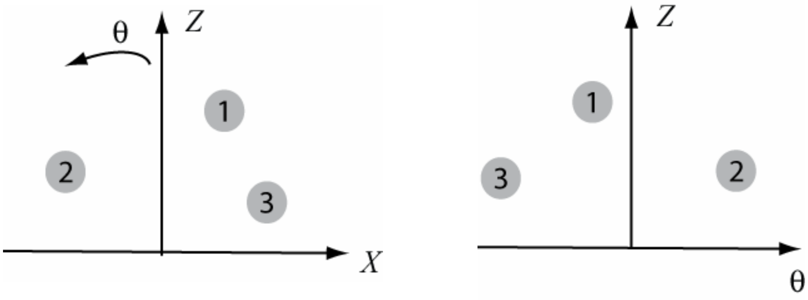


Figure 9.3 Illustration of object locations for navigation task.

9.1.3 Questionnaires

As described in Figure 9.2, we design five questionnaires. Except user background questionnaire, a 7- point scale is used as in Figure 9.4. The detailed questionnaires are listed in Table 9.3 -Table 9.8.

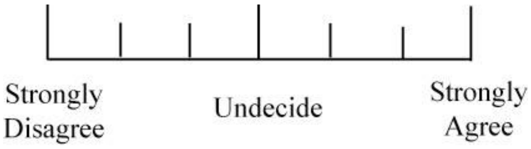


Figure 9.4 7-point scale.

Table 9.3 User background questionnaire

	Questions	**Answers**
1	Age	Fill in Blank
2	Gender	Fill in Blank
3	Occupation	Fill in Blank
4	I wear glasses	Y/N
5	I have used following displays. (multiple choices)	HMD, Auto-stereoscopic, Shutter glasses
6	I have used following tracking device to interact with a display. (multiple choices)	Magnetic tracking device, Optical tracking device
7	I am familiar with a virtual reality (VR) system	Y/N
8	How often do you play a video game? (multiple choices)	At least once a day, Approximately every two or three days, Less than once a week, Have never played video game.

Table 9.4 Display comfort and simulation fidelity questionnaire

	Questions
1	The display device is appropriate for the task.
2	The display resolution is adequate for the task.
3	The display field of view is appropriate for the task.
4	The images I see in the display are distorted.
5	The quality of the displayed image is good.
6	The objects and scenery in this virtual environment looked very realistic.
7	It was easy to get used to the display.
8	I can easily sense the depth in the displayed image.
9	The visual display quality interferes with or distracts me from performing assigned/required tasks.
10	I feel discomfort after using this system.
11	I feel eye-strain after using this system.
12	I feel difficulty adjusting to the real-world environment after using this system.
13	I feel fatigue after using this system.
14	I feel difficulty in concentrating when using this system.
15	I feel blurred vision when/after using this system.

Table 9.4: Continued

16	Overall: I am very satisfied about the display system.

Table 9.5 Interaction functionality questionnaire

	Questions
1	The interaction function is appropriately designed for the task.
2	It is easy to interact with the system using this interaction device.
3	I have difficulty remembering the interaction functions.
4	I don’t need to use all the interaction functions.
5	The input device is appropriate for the task.
6	The input device is too sensitive for me.
7	The interaction interface behaved as I expected.
8	I can’t perform some specific tasks using this interaction device.
9	I feel confused by the response of this system.
10	I have no ability to control events.
11	Interaction with the virtual reality environment is intuitive and natural.
12	The interaction device does not interfere with the performance of assigned tasks or with other activities.
13	The control functionality is distracting.
14	I experience severe delays between my actions and expected outcomes.

Table 9.6 Task functionality questionnaire

	Questions
1	It is easy to make unawareness mistake.
2	I have difficulty to remember and follow system instructions to finish the task.
3	System feedback is adequate to the task.
4	The image update is suitable when I move my head or interaction devices.
5	Displayed information is too complicated to understand..
6	I am able to take a short cut to finish the task or get unreal performance.
7	The task is very unusual in a very unusual manner.
8	I understand the meaning of this task.
9	This task is meaningful to me.
10	I can think about the application of this task. _____(fill in blank)

Table 9.7 Immersive presence questionnaire

	Questions
1	Immersion helps me to complete the task.
2	I feel I am really here and have a sense of presence.
3	I feel immersed in this virtual reality environment.
4	The image quality reduces my sense of presence.
5	The field of view enhances my sense of presence.
6	The display resolution reduces my sense of immersion.
7	I feel isolated and not part of this virtual environment.
8	I often do not know where I am in the virtual environment.
9	I got a good sense of scale in the virtual environment.
10	All of my senses are completely engaged with this environment.
11	I am able to anticipate what happens next in response to the action I performed.
12	The information coming from my various senses are inconsistent or disconnected.
13	My experiences in the virtual environment seem consistent with my real-world experiences.
14	My sense of moving around inside the virtual environment is compelling.
15	I am totally involved in this virtual environment experience.
16	It is easy to adjust to this virtual environment.

Table 9.8 Overall system performance questionnaire

	Questions
1	I have difficulty in learning how to use this system.
2	I feel comfortable using the system for a long period of time.
3	The system response is inadequate.
4	I find it is difficult to work in 3D.
5	I feel in control with this system.
6	I can see a real benefit in using this system.
7	I feel confused or distracted at the beginning of breaks in the experiment session.
8	I feel confused or distracted at the end of breaks in the experiment session.

9.2 Final Experiment

However, our methodology in Figure 9.1 is too long for a regular experiment. Most participants won't be able finish the whole experiments. Therefore, we authors [119] simplify the experiments with the model as shown in Figure 9.5.

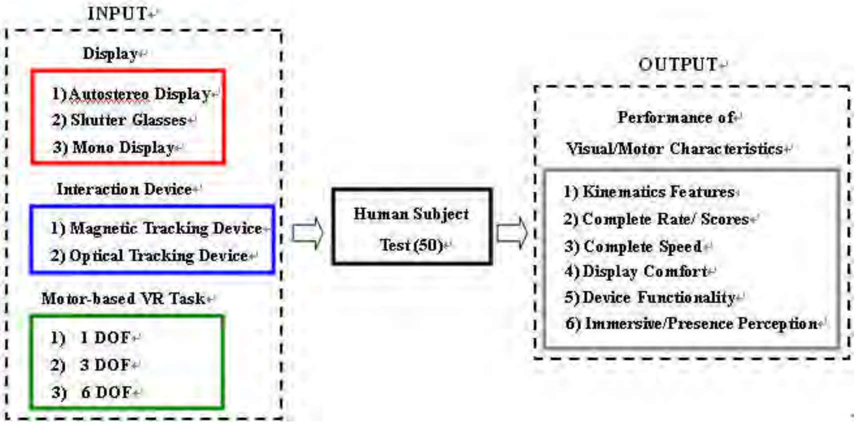


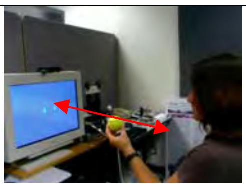
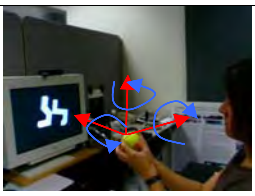
Figure 9.5 Model of our final experiment.

The experiment model is composed of input variables and output measures. The input variables are defined as display type, interaction device and degree of freedom of motion in game task. The display types and interaction devices are already described and discussed in Sec. 9.1.

The output data includes: (1) Kinematical data, measures of participant's motor behavior derived from motion data captured by the interaction devices, was collected to quantitatively describe motor features such as efficiency, discontinuity, oscillation or stability. The kinematics data is not included in this section. (2) Performance data (complete rate, complete time or error rate), are measures of participant's performance while playing each VR game. (3) User-perception data consisted of a series of questionnaires that measured participant’s perception: display comfort, functionality of interaction devices, and presence of VR game and system evaluation.

Four game tasks, developed by the authors [119], are used in this study: ball catching, depth test, reaching test and spatial rotation. Each of these game tasks involved upper limb motion within different degrees of freedom. Detailed descriptions of each game task are shown in table 9.9

Table 9.9 List of game tasks.

Task	**VR Game**	**Description**	**DOF**
Depth Test		Participants compare two virtual objects that appeared on the screen in different geometry (sphere/cube), size and depth. Participants were asked to move one of the objects using the interaction device until it was the same depth as the fixed target object.	1
Ball Catching		Participants are required to catch a ball moving towards them using a virtual hand. The trajectory and velocity of the ball was varied.	3
Reaching Test		Participants are required to reach a series of cubes presented one by one in 3D space at different locations on the screen using a virtual hand.	3
Spatial Rotation		Participants are required to superimpose two identical blocks that appeared with the same configuration but different orientation. One of the blocks was set as target block and was fixed. The second block is moved and rotated by the participant using the interaction device to superpose onto the target block.	6

Detailed description of the final experiments can be found in paper [119].

9.3 Results and Conclusions

Fifty participants (31 females, 19 males), aged 28.72 (± 5.92) years, completed the data collection procedure. Forty seven participants were right handed and 47 participants wore glasses. Only 22 percent of participants had never played video games.

We found [119] that participants were able to complete the ball catching and depth test game tasks using each of the three displays. Participants were able to complete the ball catching and depth test tasks faster when using shutter glasses than the other two displays. This might have been the result of the ability of the participant to see the ball clearly in 3D, allowing them to perceive the depth of the ball and anticipate the trajectory of ball movement. Possible reasons the AS display did not appear to provide participants with the same comfort and effectiveness as the shutter glasses could be the result of the constraints of the AS (auto-stereoscopic) display. When using the AS display, participants could have had difficulty maintaining a good 3D stereo picture when playing VR game tasks due to of the requirement of having to keep head movements within the limited area where 3D stereo can be seen. Participants rated the AS display highest for discomfort and eye strain and least satisfactory overall. However, the shutter glasses display was rated higher than monoscopic and AS displays for comfort, ability to perceive 3D, performance and time to complete.

Foe the tracking devices, both devices (magnetic tracking device and optical tracking device) performed well. But, the optical tracking device, developed by the authors, is less expensive than the magnetic tracking.

Overall, the results of this study provide information (detailed results can be found in [119]) about the use of different display types and interaction devices with four VR games. The results suggest that the use of shutter glasses and magnetic or optical tracking are likely to be the most appropriate display and interaction devices for use in VR environments.

Chapter 10 Future Directions, Summary and Conclusions

Because of the advent of AS displays and other stereoscopic display techniques, stereoscopic-based applications are potential systems for next generation of virtual environment. Since stereoscopic imaging and panorama techniques are still in the early stages of development, there are still many possible areas of improvement. The work described here can be extended and improved in future research efforts. We list some interesting possible topics in this section.

10.1 Stereo Panorama Rendering and Walkthrough

The conventional method for rendering a panorama is by pixel and texture mapping the panorama to a cylindrical surface. The cylinder surface is often implemented by a polygon approximation (usually around 300 faces for entire cylinder), and each face is rendered by projecting the panorama from a suitable viewpoint to the face plane. In work described in chapter 8, we render stereo panoramas generated using a swing panorama capturing structure (or other circular projection) by projecting the left/right panorama patch separately onto each face of the left/right approximation polygon. A distortion correction is usually required for circular projection, but the disparity is almost the same in central projection and circular projection and can be tolerated by the human viewers because we approximate the cylindrical surface by a 300 face polygon (almost 1 degree per face).

From an examination of the rendering results, we feel that users may perceive an effect using a 3D scene model instead of a single circular projection model. The 3D scene model can be calculated from the depth map of the left and right panorama [90]. With this depth information, the stereo panorama can be mapped onto it with image-based rendering methods such as ray tracing, and adjusting the disparity for each viewing direction.

After building a scene model, in principle we can create a virtual environment to let user walk around. To let user feel more immersive and realistic, cameras/camcorders can be used to capture user images and insert images/video into the VR system.

10.2 Subjective Tests

More complete subjective tests as an extension of what described in chapter 9 will be an interesting research topic in the future. Tests can compare the performance of our virtual environment system with the panorama and stereo content with different rendering methods, and equipped with different interaction tools and various displays such as HMD, shutter glasses, AS displays, or even big screen with anaglyph glasses such as pano-chamber. The results can be examined and verified by quantitative analysis with knowledge of human perceptual principles and evaluation with questionnaires developed in chapter 9.

In the future experiments, a 3D concept space as in Figure 10.1 can be defined to exam the influence of display devices, and three taxonomic elements (FOV, Stereo, and Resolution) can also be chosen into consideration due to their influences to user immersion experience and navigation behavior in virtual environments. It will also be an interesting research to design some tests and evaluate some factors (listed in Table 10.1) that may influence VR experiments.

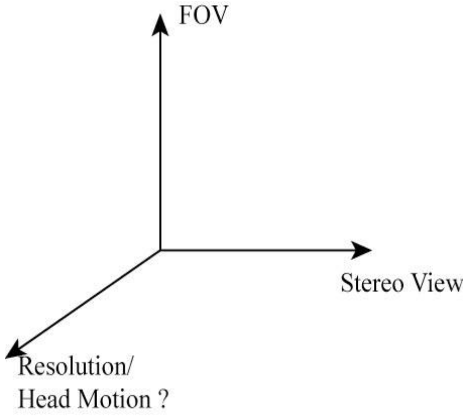


Figure 10.1 Factors of influence of a display device.

Table 10.1 Influence factors of virtual environments

Realism Factors	**Sensing Factor**	**Control Factor**	**Distraction Factor**
Scene Realism	Egocentric/Non Ego centric	Degree of Movement	Isolation[1]
Information Consistency[2]	Display Information (Stereo, FOV)	Degree of Control	Task Focus[3]
Anticipation[4]	Image Quality (Resolution)	Manipulation Method[5]	Perceptual Issue Sweet Spot
Meaning of the Task[6]			

10.3 Disparity Adjusting for General Stereo Images and Video

In the chapter 7, we propose a general disparity adjusting framework for stereo images and video. However, to create a software package which can generate disparity map

[1] HMD provides a virtual space that is isolated from real space.
[2] Minimize confusing of visual cues.
[3] Ease of concentration on a specific task.
[4] Does the system response satisfy user's expectation?
[5] Dimensions of virtual space must match the actual dimensions of user's space.
[6] How well does this task simulate the real world?

accurately and in real time for different kind image sets and video is a challenge issue. A complete survey and evaluation of techniques for disparity map generation methods is needed for practical research usage. Besides, to create a commercial software for stereo images and video editing and a plug-in for current popular commercial image and video editing software (e.g. Adobe Photoshop, Premier) is also a interesting research work can be done in the near future.

10.4 Stereo Panorama Video

For our current camera setup, we can only generate stereo panorama images off-line or for static exhibition purpose. However, users need to feel surrounded by the dynamic scenery (e.g. People can walk around in VR) to get more immersive sense and realistic feeling. Based on my knowledge, there is no stereo panorama camcorder available on the market. How to capture and generate high resolution stereo panorama video in real-time will be another interesting topic.

10.5 Digital Stereoscopic Camera

Recently, 3D World [1] announced its digital stereoscopic camera. However, the camera uses off-axis setup. No stereo camera can be like our human eyes using toe-in axis setup and can adjust the point of fixation automatically or changing disparity automatically according to the distance range. It will be interesting to develop a digital stereoscopic camera has such functions.

10.6 Military Training

Another challenge is to design applications using stereo panorama virtual environment to train war-fighters make decisions and perform actions according to available information and events happened around them. One possible application is “Virtual MOUT (Military Operations in Urban Terrain) Training System”. Cognitive information technology can be used to design the system. The training results can be evaluated to exam if the stereo panorama VR system provides a better and more realistic training environment because of 3D information and 360 degree of view.

10.7 Smart Home

Based on the idea of “Smart Home”, we can expect homes to have multiple wide high-definition screens, panoramas of at least 180 degrees and user controls of perspectives that can free up the human eye to rove and make full use of peripheral vision [78]. Another future research for us is to develop VRs with simulated mixed reality (e.g. VRs that provide city and museum navigation and cooperate with Google Earth) for or TV entertainment application and be interactive accessible for news and journalism. In the 3D live news and journalism, audience can really feel on the spot and the news reporter is just stand in front and talking to them face to face. Also, for commercial advertisement application, we can add 3D information of products that audience can interact with and see from different viewing directions.

10.8 Conclusions and Summary

In this thesis, we describe panoramic stereo impressive image capture and display systems, and techniques for adjusting the perceived stereo effect of 2D regions in the images. Besides, we propose a general framework to extend our disparity morphing method for other stereo images and video editing. A finished stereo panorama in anaglyph is shown in Figure 10.2.

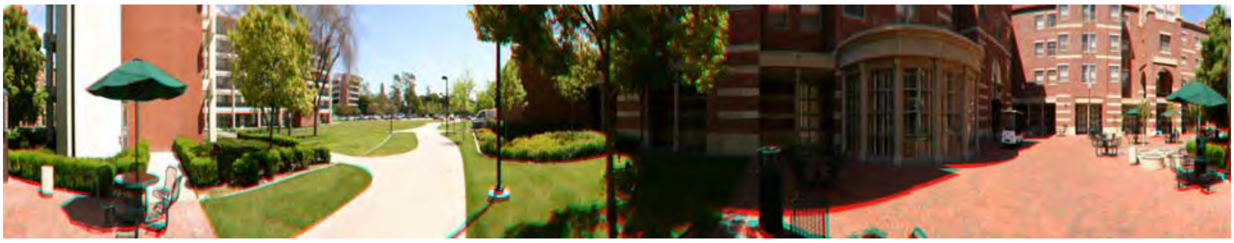


Figure 10.2 Stereo panorama in anaglyph.

We also create a virtual environment using the stereo panorama we generated and equipped with different displays: LCD monitor with anaglyph glasses, CRT monitor with shutter glasses, Sharp AS (autostereoscopic) displays, eMagin HMD; and we allow the user to interact it using keyboard, HMD's head tracker, FOB tracker [2], and optical tracking device [119].

Bibliography

[1] 3D World (2006). *120 Tri-lens Stereo Camera*. http://www.3dworld.cn/.

[2] Ascension Technology Corp. (2004). *Flock of Birds*. http://www.ascension-tech.com/products/flockofbirds.php.

[3] Aizawa, K., K. Sakaue, and Y. Suenaga (2004). *Image Processing Technologies: Algorithms Sensors, and Applications*, pp. 116-141.

[4] Alpaslan, Z.Y. and A.A. Sawchuk (2004). "Three-Dimensional Interaction with Autostereoscopic Displays," *Proc. SPIE Vol. 5291 Stereoscopic Displays and Virtual Reality Systems XI Symposium*, San Jose, CA.

[5] Alpaslan, Z.Y., S.-C. Yeh, A.A. Rizzo, and A.A. Sawchuk (2005). "Quantitative Comparison of Interaction with Shutter Glasses and Auto-stereoscopic Displays," *Proc. SPIE Vol. 5664 Engineering Reality of Virtual Reality 2005 Symposium*, San Jose, CA.

[6] Alpaslan, Z.Y. (2005). "Three-Dimentional Interaction with Autostereoscopic Displays", *PHD theis in Electrical Enginerring of USC*.

[7] Anderson, P. (2005). "Advanced Display Technologies," *JISC Technology and Standards Watch*, Bristol, UK.

[8] Aukstakalnis, S. and D. Blatner (1992). *Silicon Mirage - The Art and Science of Virtual Reality*, Berkeley, CA, Peachpit Press.

[9] Barron, J.F., D.J. Fleet, and S.S. Beauchemin (1994). "Performance of Optical Flow Techniques," *IJCV 12:1*, pp. 43-77.

[10] Banerjee, P., G. Bochenek and J. Ragusa (2002). "Analyzing the Relationship of Presence and Immersive Tendencies on the Conceptual Design Review Process," *Journal of Computing and Information Science in Engineering*, vol. 2, iss. 1, pp. 59-64, 2002.

[11] Bauermann, I. and E. Steinbach (2004). "Low-complexity Image-based 3D Gaming," *Vision, Modeling, and Visualization (VMV '04)*, pp. 101-107. Stanford, USA.

[12] Bendels, G.H. and R. Klein (2003). "Mesh Forging: Editing of 3D-meshes Using Implicitly Defined Occluders", *In Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry processing*, Eurographics Association, pp. 207-217.

[13] Bourke P. (1999). *Autostereoscopic Lenticular Images*, the Swinburne Centre for Astrophysics and Supercomputing: Australia.

[14] Boman, D.K. (1995). “International Survey: Virtual Environment Research,” *IEEE Computer*, vol. 28, pp. 57-65.

[15] Brown, M.Z., D. Burschka, and G.D. Hager (2006). “Advances in Computational Stereo”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993-1008.

[16] Bulthoff, H. and Y. Yuille (1996). “A Bayesian Framework for the Integration of Visual Modules,” *Attention & Performance XVI: Information Integration in Perception and Communication*, MIT Press, Cambridge, 49-70.

[17] Chen, J., S.-K. Wei, and R. Klette (2003). “Authoring and Visualizing Stereo Panoramic Images with Independent Objects,” *IVCNZ.*

[18] Chen, S.E. (1995). “QuickTime VR–An Image-Based Approach to Virtual Environment Navigation,” *Computer Graphics (SIGGRAPH’ 95)*, pp. 29-38.

[19] Choi, C.H., B.H. Kwon, and M.R. Choi (2004). “A Real-Time Field-Sequential Stereoscopic Image Converter”, *IEEE Trans. On Consumer Electronics*.

[20] Chankin, G. (1974). “An Algorithm for High Speed Curve Generation,” *Computer Graphics and Image Processing*, vol. 3, pp.346-349.

[21] Cormen, T.H., C.E. Leiserson, R.L. Rivest, and C. Stein (2006). *Introduction to Algorithms (2nd Edition)*, Prentice Hall.

[22] Comaniciu, D. and P. Meer (2002). “Mean Shift: A robust approach toward feature space analysis,” *IEEE Trans. Pattern Anal. Machine Intelligence*, vol. 24, no. 5, pp. 603–619.

[23] Comaniciu, D. and P. Meer (1999). “Mean-Shift Analysis and Applications,” *Proc. of IEEE Int. Conf. on Computer Vision*, Greece, pp. 1197-1203.

[24] Criminisi, A., J. Shotton, A. Blake, and P. Torr (2004). “Efficient Dense Stereo and Novel-View Synthesis for Gaze Manipulation in One-to-one Teleconferencing”, *[Technical Report] MSR-TR-2003-59*, Microsoft Research.

[25] Cruz-Neira, C., D. Sandin, and T. DeFanti (1993). “Surround-screen Projection-based Virtual Reality: The Design and Implementation of the CAVE,” *Proceedings of the 20th annual conference on Computer Graphics and Interactive Techniques*, p.135-142.

[26] DDD (2003). “Product List”. http//www.ddd.com/products/pro_main_frm.htm.

[27] Eisert, P., Y. Guo, A. Riechers, and J. Rurainsky (2004). "High-Resolution Interactive Panoramas with MPEG-4", *9th Workshop for Vision Modeling and Visualization*, Stanford, CA, USA.

[28] Elder, J.H. and R.M. Goldberg (1998). "Image Editing in the Contour Domain," *IEEE Proc. of Computer Vision and Pattern Recognition Conference*, Santa Barbara, pp. 374-381.

[29] Electronic News (2004). "3D Display Sales to Quadruple by 2010," *Electronic News*, http://www.reed-electronics.com/electronicnews/article/CA442887.html.

[30] Faugeras, O. and O. T. Luong (2001). *The Geometry of Multiple Image*, Cambridge Mass.: The MIT Press.

[31] Faugeras, O. and R. Keriven (1998). "Variational Principles, Surface Evolution, PDE's, Level Set Methods, and the Stereo Problem", *IEEE Trans. Image Processing*, vol. 7, pp. 336-344.

[32] Foley, J.D., A. Van Dam, S.K. Feiner, and J. Hughes (1997). *Computer Graphics: Principles and Practice*. *(2nd Edition)*, Addison-Wesley.

[33] Forsyth, D.A. and J. Ponce (2002). *Computer Vision: A Modern Approach*, Prentice Hall.

[34] Fua, P. and Y.G. Leclerc (1995). "Object-Centered Surface Reconstruction: Combining Multi-Image Stereo and Shading", *International Journal Computer Vision*, vol. 16, pp. 33-56.

[35] Garcia, B.J. (1996). "Approaches to Stereoscopic Video Based on Spatio-temporal Interpolation", *Proc. of SPIE Stereoscopic Displays and Virtual Reality Systems IV*, vol. 2653, pp. 85-95.

[36] Gluckman, J., S. Nayar, and K. Thoresz (1998). "Real-Time Omnidirectional and Panoramic Stereo," *Proc. DARPA Image Understanding Workshop*, pp. 299-303.

[37] Hartley, R. and A. Zisserman (2000). *Multiple View Geometry in Computer Vision*, Cambridge: UK, Cambridge Univ. Press.

[38] Hen, J.-H., M. Jung, C. Lee, and E.-Y Ha (2002). "Panorama Field Rendering with Scene Depth Estimation," *Electronics Letters*, vol. 38, iss. 14, pp.704 -705.

[39] Hirschmüller, H. (2001). "Improvements in Real-time Correlation Based Stereo Vision", *In IEEE Workshop on Stereo and Multi-Baseline Vision, IJCV.*

[40] Holloway, R. and A. Lastra (1993). "Virtual Environments: A Survey of the Technology," *Technical Report*, University of North Carolina, Chapel Hill, NC, Report No. TR93-033.

[41] Holiman, N.S. (2004). "Mapping Perceived Depth to Regions of Interest in Stereoscopic Images," *Proc. Stereoscopic Displays and Virtual Reality Systems XI Symposium*, vol. 5291, San Jose, CA.

[42] Huang, Y.P., C.S. Chen, G.Y. Tang, Y.P. Tsai, C.F. Huang, S.W. Lin, and C.H. Yu (2001). “A Purely Image-based Approach to Augmenting Panoramas with Object Movies,” *Proc. Intl Workshop on Computer Graphics and Virtual Reality*, pp. 80-88.

[43] Intel (2005). “Intel Open Source Computer Vision Library (OpenCV)”. http://www.sourceforge.net/projects/opencvlibrary.

[44] Iinuma, T., H. Murata, S. Yamashita, and K. Oyamada (2000). “Natural Stereo Depth Creation Methodology for a Real-time 2D-to-3D”, *SID 00 DIGEST*.

[45] Jones, G., D. Lee, N. Holliman, and D. Ezra (2001). “Controlling the Perceived Depth in Stereoscopic Images,” *Proc. SPIE Vol. 4297A: Stereoscopic Displays and Virtual Reality Systems VIII*, San Jose, California.

[46] Kawanishi, T., K. Yamazawa, H. Iwasa, H. Takemura, and N. Yokoya (1998). “Generation of High-Resolution Stereo Panoramic Images by Omnidirectional Imaging Sensor using Hexagonal Pyramidal Mirrors,” *Proc. 14th IAPR Int. Conf. on Pattern Recognition (14ICPR)*, Australia, vol. I, pp.485-489.

[47] Kalawsky, R.S. (1999). “VRUSE - A Computerized Diagnostic Tool: for Usability Evaluation of Virtual/Synthetic Environment Systems”, *Applied Ergonomics*, vol. 30, pp. 11-25.

[48] Kang, S.B. and R. Weiss (1997). “Characterization of error in compositing panoramic images,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’ 97)*, pp. 103-109.

[49] Kang, E., I. Cohen, and G. Medioni (2000). “A Graph-Based Global Registration for 2D Mosaics,” *15th International Conference on Pattern Recognition (ICPR 00)*, vol. 1, pp. 257-260.

[50] Kass, M., A. Witkin, and D. Terzopoulos (1987). “Snakes: Active Contour Models,” *First International Conference on Computer Vision*, pp.259-268.

[51] Kim, D.-H. and J.-S. Choi (2004). “View Transition Algorithm for the Walkthrough on the Panorama Based Navigation,” *ACCV*, vol.1, pp1109-1114.

[52] Kim, S., E. Chang, C. Ahn, and W. Woo (2003). “Image-based Panoramic 3D Virtual Environment Using Rotating two Multi-view Cameras,” *IEEE Proc. ICIP 2003*, vol. 1, pp. 917-920.

[53] Kolmogorov, V. and R. Zabih (2002). “Multi-Camera Scene Reconstruction via Graph Cuts”, *In Proc. Europ. Conf. Computer Vision*, Copenhagen, Denmark, pp. 82-96.

[54] Kutulakos, K.N. and S.M. Seitz (2000). “A Theory of Shape by Space Carving”, *Int’l J. Computer Vision,* vol. 38, no. 3, pp. 199-218.

[55] Lampton, D.R., B. W. Knerr, S. L. Goldberg, J. P. Bliss, J. M. Moshell, and B.S. Blau (1994). “The Virtual Environment Performance Assessment Battery (VEPAB): Development and Evaluation”, *Presence: Teleoperators and Virtual Environments,* vol. 3(2), pp.145–157.

[56] Lucas, B.D. and T. Kanade (1981). “An Iterative Image Registration Technique with an Application to Stereo Vision,” *Proceedings IJCAI81*, pp. 674-679.

[57] Li, Y., H.-Y. Shum, C.-K. Tang, and R. Szeliski (2004). “Stereo Reconstruction from Multi-Perspective Panoramas,” *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 26, iss. 1, pp. 45- 62.

[58] Li, Y., J. Sun, C.K. Tang, and H.Y. Shum (2004). “Lazy Snapping,” *ACM Trans. Graph.*, vol. 23, pp. 303-308.

[59] Marr, D. (1982). “Vision: A Computational Investigation into the Human Representation and Processing of Visual Information,” Freeman, San Francisco.

[60] Matsumoto, Y., H. Terasaki, K. Sugimoto, and T. Arakawa (1997). “Conversion System of Monocular Image Sequence to Stereo Using Motion Parallax”, *Proc. of SPIE Vol. 3012 Stereoscopic Displays and Virtual Reality Systems IV Symposium*, pp. 108-112.

[61] Manber, U. (1989). *Introduction to Algorithms-A Creative Approach*, pp. 208-212, 1989.

[62] Mcallister, D. (2005), “Display Technology: Stereo and 3D Display Technologies,” *Wiley Encyclopedia on Imaging Science and Technology*, UK, pp. 1327-1344.

[63] Mortensen, E.N. and W.A. Barrett (1995). “Intelligent Scissors for Image Composition,” *Proc. of the ACM SIGGRAPH: Computer Graphics and Interactive Techniques*, Los Angeles, pp. 191-198.

[64] Mortensen, E.N. and W.A. Barrett (1998). “Interactive Segmentation with Intelligent Scissors,” *Graphic Models and Image Processing,* vol. 60, pp.349-384.

[65] Montgomery, D., G.J. Woodgate, A. Jacobs, J. Harrold, and D. Ezra (2001). “Analysis of The Performance of a Flat Panel Display System Convertible between 2-D and Autostereoscopic 3-D modes,” *Proc. SPIE, Stereoscopic Displays and Applications*, vol. 4297, pp. 148.

[66] Morar, S.S., R.D. Macredie, and T. Cribbin (2002). “An Investigation of Visual Cues used to Create and Support Frames of Reference and Visual Search Tasks in Desktop Virtual Environments,” *Virtual Reality 6*, vol. 3, 140-150.

[67] Nash, E.B., G.W. Edwards, J.A. Thompson, and W. Barfield (2000). “A Review of Presence and Performance in Virtual Environments,” *International Journal of Human-Computer Interaction*, vol. 12**,** pp.1-41.

[68] Nagahara, H., Y. Yagi, and M. Yachida (2003). “Super-resolution Modeling Using an Omnidirectional Image Sensor,” *IEEE Trans. on Systems, Man and Cybernetics part B*, vol. 33, pp. 607-615.

[69] Nayar, S.K. (1997). “Catadioptric Omnidirectional Camera,” *Proc. of IEEE Computer Vision and Pattern Recognition (CVPR).*

[70] Nayar, S.K. and A. Karmarkar (2000). “360 x 360 Mosaics,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 388-395.

[71] Okoshi, T. (1976). *Three-Dimensional Imaging Techniques*, Academic Press, New York, 1976.

[72] Panoram Technologies Inc. (2001). http://www.panoramtech.com/.

[73] Pauly, M., R. Keiser, L. P. Kobbelt, and M. Gross (2003). “Shape Modeling with Point-Sampled Geometry”, *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 641-650.

[74] Peleg, S. and J. Herman (1997). “Panoramic Mosaics by Manifold Projection,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 338-343.

[75] Peleg, S., M. Ben-Ezra, and Y. Pritch (2001). “Omnistereo: Panoramic Stereo Imaging,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23**,** no.3, pp. 279-290.

[76] Peleg, S., Y. Pritch, and M. Ben-Ezra (2000). “Cameras for Stereo Panoramic Imaging”, *CVPR’ 00*, Hilton Head Island, vol. I, pp. 208-214.

[77] Piegl, L. and W. Tiller (1997). *The NURBS Book (2nd Edition)*, Springer-Verlag 1995–1997.

[78] Pryor, L. (2007). "Journalism can be Welcome in 'Smart Homes'," http://www.ojr.org/ojr/stories/071026pryor/, *USC Annenberg On line Journalism Review*.

[79] Reder, A., E. Hendriks, and J. Biemond (1997). "Synthesis of Multi-Viewpoint at Non-Intermediate Positions," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP' 97)*, vol. 4, pp. 2749-2752.

[80] Reese, L.J. and W.A. Barrett (2002). "Image Editing with Intelligent Paint," *Proc. of Eurographics*, vol. 21.

[81] Rizzo, A.A., K. Ghahremani, L. Pryor, and S. Gardner (2003). "Immersive 360-Degree Panoramic Video Environments: Research on Creating Useful and Usable Applications," *Human-Computer Interaction: Theory and Practice*, vol. 1, pp. 1233-1237.

[82] Rolland, J.P., L.D. Davis, and Y. Baillot (2000). "A Survey of Tracking Technology for Virtual Environments," *Augmented Reality and Wearable Computers*.

[83] Saxena, A., S. H. Chung, and A. Y. Ng (2006). "Learning Depth from Single Monocular Images", *In NIPS*, vol. 18.

[84] Sawhney, H.S., S. Hsu, and R. Kumar (1998). "Robust Video Mosaicing through Topology Inference and Local to Global Alignment," *Proceedings of the 5th European Conference on Computer Vision-Volume II*, pp. 103-119.

[85] Scharstein, D. and R. Szeliski (2002). "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms", *International Journal of Computer Vision*, vol. 47, pp. 7-42.

[86] Scharstein, D. and R Szeliski (2002). *Stereo Data Sets with Ground Truth*. http://cat.middlebury.edu/stereo/data.html.

[87] Shimamura, J., H. Takemura, N. Yokoya, and K. Yamazawa (2000). "Construction and Presentation of a Virtual Environment Using Panoramic Stereo Images of a Real Scene and Computer Graphics Models, " *Proc. 15th IAPR Int. Conf. on Pattern Recognition*, vol. IV, pp. 463 – 467.

[88] Shi, J. and C. Tomasi (1994). "Good Features to Track," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94).*

[89] Shum, H.-Y. and L.-W. He (1999). "Rendering with Concentric Mosaics," *SIGGRAPH*, pp. 299-306.

[90] Shum, H.-Y. and R. Szeliski (1999). "Stereo Reconstruction from Multiperspective Panoramas," *Seventh International Conference on Computer Vision (ICCV'99)*, pp. 14-21, Kerkyra, Greece.

[91] Shao, M., T. Simchony, and R. Chellappa (1988). "New Algorithms for Reconstruction of a 3D Depth Map from One or More Images", *In Proc IEEE CVPR*, pp. 530-535.

[92] Simon, A. and S. Beckhaus (2004). "Ominidirectional Stereo Surround for Panorama Virtual Environments," *Proc. of IEEE Virtual Reality*, pp. 67-279.

[93] Sorkine, O., Y. Lipman, D. Cohen, M. Alexa, C. Räossl, and H. P. Seidel (2004). "Laplacian Surface Editing", *In Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pp. 179-188.

[94] StereoGraphics (1997). *Developer's Handbook.*

[95] StereoGraphics (June 2003). "SynthaGram website", http//stereographics.com/products/synthagram/synthagram.htm.

[96] StereoGraphics (June 2003). *The SynthaGram Handbook.* http//www.stereographics.com/products/synthagram/The%20SynthaGram%20Handbook%20v71.pdf.

[97] Stereoscopic 3D Virtual Reality. http://www.stereo3d.com.

[98] Sun, C. and S. Peleg (2004). "Fast Panoramic Stereo Matching Using Cylindrical Maximum Surfaces," *IEEE Transactions on Systems, Man, and Cybernetics*, part B, vol. 34(1), pp.760-765.

[99] Sun, J., Y. Li, S. B. Kang, and H.-Y. Shum (2005) "Symmetric Stereo Matching for Occlusion Handling," *Computer Vision and Pattern Recognition*, vol. 2, pp. 399-606.

[100] Szeliski, R. and H.-Y. Shum (1997). "Creating Full View Panoramic Image Mosaics and Environment Maps," *Proceedings of the 24th Annual Conference on Computer graphics and interactive techniques*, p.251-258.

[101] Tan, K.-H. and N. Ahuja (2001). "Selecting Objects with Freehand Sketches," *IEEE Proc. of Int. Conference on Computer Vision*, vol. 1, pp.337-344.

[102] Tam, W.J. and L. Zhang (2006). "3D-TV Content Generation: 2D-to-3D Conversion**",** *IEEE International Conference on Multimedia and Exposition*.

[103] Tzavidas, S. and A.K. Katsaggelos (2002). "Multicamera Setup for Generating Stereo Panoramic Video," *Proc. 2002 SPIE Conference on VCIP*, San Jose, CA.

[104] Uyttendaele, M., A. Eden, and R. Szeliski (2001) "Eliminating Ghosting and Exposure Artifacts in Image Mosaics," *Proc. IEEE Int. Conf. Computer. Vision and Pattern Recognition (CVPR)*, vol. II, pp. 509–516.

[105] Virtual Realities Inc. (2008). "Head Mounted Displays 1995-2008". http://www.vrealities.com/hmd.html.

[106] Van Berkel, C. and J. Clarke (1997). "Autostereoscopic Display Apparatus," *United States Patent. 6,064,424*, panted May 26, 2000, filed February 12 , 1997.

[107] Van. Berkel, C. (1999). "Image Preparation for 3D-LCD," Proceedings of SPIE Vol. 3639 Stereoscopic Displays and Virtual Reality Systems VI Symposium.

[108] Wang, C., J. Garagate, and A.A. Sawchuk (2005). "Stereo Panorama Personal Virtual Environment," Frontiers in Optics, Optical Society of America Annual Meeting, Tucson, AZ, October 2005; *OSA Annual Meeting Program 2005 Technical Digest*, (Optical Society of America, Washington, DC, 2005).

[109] Wang, C. and A.A. Sawchuk (2006). "Adaptive Parallax Control for Multiple View Stereo Panorama," *Proc. of SPIE Vol. 6005 Stereoscopic Displays and Virtual Reality Systems XIII Symposium*, San Jose, CA.

[110] Wang, C. and A.A. Sawchuk (2006). "Region Based Disparity Adjusting for Stereo Panorama," *International Workshop on Multimedia Signal Processing*, BC, Canada.

[111] Wang, C. and A.A. Sawchuk (2006). "Object-Based Stereo Panorama Disparity Adjusting," Frontiers in Optics, Optical Society of America Annual Meeting, New York.

[112] Wang, C. and A.A. Sawchuk (2007). "Object-Based Disparity Adjusting Tool for Stereo Panoramas," *Proc. of SPIE Vol. 6490 Stereoscopic Displays and Virtual Reality Systems XIV Symposium*, San Jose, CA.

[113] Wang, C. and A.A. Sawchuk (2008). "Disparity Manipulation for Stereo Images and Video," *Proc. of SPIE Vol. 6803 Stereoscopic Displays and Virtual Reality Systems XIX Symposium*, San Jose, CA.

[114] Wei, S.K., F. Huang, and R. Klette (2002). "Specification of Image Acquisition Parameters for Stereo Panoramas," *ICPR (3)*, pp. 603-606.

[115] Weerasinghe, C., P. Ogunbona, and W. Li (2001). "2D to Pseudo-3D conversion of "Head and Shoulder" Images Using Feature Based Parametric Display maps", *ICIP*, vol. 3, pp. 963-966.

[116] Wolberg, G. (1994). *Digital Image Warping*, IEEE Computer Society Press, CA, USA.

[117] Woodgate, G., J. Harrold, A. Jacobs, R. Mosley, and D. Ezra (2000). “Flat Panel Autostereoscopic Displays Characterization and Enhancement,” *Proc. SPIE, Stereoscopic Displays and Virtual Reality Systems VII*, vol. 3957, pp. 153.

[118] Yamazawa, K., Y. Yagi, and M. Yachida (1993). “Omnidirectional Imaging with Hyperboloidal Projection”, *Proceedings of the International Conference on Robots and Systems*.

[119] Yeh, S.-C., B. Lange, C.Y. Chang, C. Wang, A.A. Sawchuk, and A. Rizzo (2008). “Effect of Stereoscopic Displays and Interaction Devices on Human Motor Behavior,” *Proc. SPIE Vol. 6804 The Engineering Reality of Virtual Reality 2008 Symposium,* San Jose, CA.

[120] Zhang, L., W.J. Tam, and D. Wang (2004). “Stereoscopic Image Generation Based on Depth Images,” *International Conference on Image Processing (ICIP)*, pp. 2993-2996.

[121] Zomet, A. and S. Peleg (2000). “Efficient Super-Resolution and Applications to Mosaics,” International Conference on Pattern Recognition, vol. 1, pp. 579-583.

Appendix A– Stereoscopic Image Display Toolkits

Current software for generating the stereoscopic images can be classified as two groups. One operates on real stereoscopic photograph data, another is working on stereo images generated based on the computer graphic model. In Table A.1, we list some stereo imaging development toolkits that display the stereoscopic images or output the stereo result of the graphic library (OpenGL/DirectX) on AS related displays.

Table A.1 Stereo image display preparation toolkits

Software	**Function**	**Note**
TriDefVisualizer OpenGL SDK	Allows the OpenGL graphics based program to be displayed on TriDef supported displays in Table A.2.	
NewSight OpenGL Enhancer	Allows the OpenGL graphics protocol to output to NewSight AS displays	
NewSight 3D SDK	Allows the OpenGL, Driectx based graphics codes to NewSight AS displays	
NewSight Maya Plug-in	Calculation of multiple views of a scene with multi-view selectable cameras	
NewSight 3D MAX Plug-in	Calculation of multiple views of a scene with multi-view selectable cameras	3D Studio MAX
TriDefControl Panel	Configure TriDef supported displays Table A.2. easy, such as scene depth and focal point	

Table A.2 TriDef supported displays

AS PC Displays	Sharp Actius AL3D 2D/3D Switchable Notebook Computer
	Sharp Actius RD3D 2D/3D Switchable Notebook Compute
	Sharp LL-151-3D 2D/3D Switchable Desktop Display
	StereoGraphics SG204 20” LCD
	StereoGraphics SG404 40” LCD
	VR21 45" 3D LCD
	SeeReal Cn
AS Handheld Displays	Sharp 2.2” 2D/3D Switchable qVGA
	Sanyo 2.2” 2D/3D Switchable qVGA
	Ocuity 2.2” 2D/3D Switchable qVGA
Television with shutter glasses	Arisawa Dimen 17" 3D LCD