The whole of science is nothing more than a refinement of everyday thinking.

– Albert Einstein, 1879-1955.

University of Alberta

A Solution to the Eye Contact Correction in Tele-presence Systems

by

Xiaozhou Zhou

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

Fall 2013
Edmonton, Alberta

To all people who love me and support me these years.

Abstract

With the development of globalization, numerous tele-conferencing systems have been developed to shorten the distance between people. Tele-presence is one of these systems which can broadcast high-quality audio and video to remote sites trying to give the illusion to the participants that they are co-located in a common virtual meeting room. In order to provide this illusion, one main criterion is to maintain eye contact. This is important because eye contact can convey more information than languages sometimes and correct eye contact is one of the essential factors which make the participants feel like they are talking face-to-face. However, some of the successful methods to eye contact correction problem in tele-presence systems need special equipment which is either large or expensive. In this thesis, we propose a software-based solution to the eye contact correction problem. This solution uses depth-based view interpolation to create virtual views from a network of cameras located around the display screen. Using this method, users have the freedom to adjust their viewpoints to allow eye contact or to focus their attention on important regions.

To improve the illusion of being in a common meeting room, we propose a new image matting algorithm to extract a participant from the background that can deal with hair and small details. This is essential as any small imperfections in the matte will destroy the illusion of being in a same virtual meeting room.

In the thesis, we also explore how to overcome the uneven illumination found in tele-presence rooms. We present solutions to the illumination invariant stereo matching problem to create better disparity maps. Taking the advantage of the parallel structure of local stereo matching, the GPU implementation of the proposed solution can run in real time. Intensive experiments demonstrate the effectiveness and efficiency.

In general, our main goal in this thesis is to develop core technologies that will eventually be used in future tele-presence systems.

Acknowledgements

It is a pleasure to thank all the people who made this thesis possible. First of all, I would like to express my sincere gratitude to my supervisor Prof. Pierre Boulanger. Thanks for enrolling me in your group and enlightening me with you knowledge and scientific experience. Without your insightful advice and patient guidance, my research work would have been much more difficult. Not only your vision, but also your shared life experience, is invaluable to me in the future. I can never thank you enough.

I am also indebted to my committee members. Thanks Prof. Ehab Elmallah for chairing my defense and thanks Prof. Panos Nasiopoulos, Prof. Vicky Zhao, Prof. Nilanjan Ray, and Prof. Martin Jagersand for spending time in reading my thesis and giving constructive suggestions.

I am grateful to the support staffs in the department, especially Edith Drummond, Fran Moore, Sharon Bell, Deborah Choi, and Louise Whyte. You always patiently help me to solve all kinds of problems, about courses, program, and finance.

I am thankful to my lab mates and friends Xing-Dong Yang, Robyn Taylor, Ying Xu, Hui Wang, Xida Chen, Jichuan Shi, Feng Chen, Rui Shen, and Zhijie Wang. Thanks for your encouragement and company. I am not lonely in the long journey of my Ph.D. program. I will always remember the days we worked together for the homeworks and shared research experiences. You cheered me up when I was down and filled me with happiness. You make me feel so warm in a foreign country, especially in the extremely cold winters in Edmonton. Wish you all have a wonderful future after graduation. Because of the page limitation, I cannot list all the names of people who helped me. However, I want you to know that your friendship and help are precious treasure in my whole life.

Finally, I wish to take this opportunity to express my deepest appreciation to my beloved parents. This thesis cannot be done without your endless love, support, and understanding. To them, I dedicate this thesis.

Table of Contents

1 Introduction 1

1.1 A Brief History of Tele-presence Systems . . . 3

1.2 A Brief History of Free Viewpoint Video Systems . . . 6

1.3 Motivations and Contributions . . . 8

1.3.1 Solving the Eye Contact Problem for Large Tele-presence Screens . . . 8

1.3.2 Foreground Extraction by Mutual-Information-based Image Matting . . . 11

1.3.3 Real-time Illumination Invariant Stereo Matching . . . 12

1.4 Organization . . . 12

2 Literature Review 15

2.1 Eye Contact Correction in Tele-presence Systems . . . 15

2.1.1 Hardware-based Solutions for Eye Contact Correction . . . 15

2.1.2 Software-based Solutions of Eye Contact Correction . . . 17

2.1.3 Comparison of Eye Contact Correction Methods . . . 22

2.2 Stereo Matching . . . 22

2.2.1 Local Stereo Matching . . . 23

2.2.2 Global Stereo Matching . . . 26

2.2.3 Comparison of Stereo Matching Algorithms . . . 29

2.3 View Interpolation . . . 29

2.3.1 View Interpolation with Dense Correspondence Information . . . 30

2.3.2 View Interpolation with Sparse Correspondence Information . . . 34

2.3.3 Ray-based View Interpolation . . . 35

2.3.4 The Comparision of View Interpolation Algorithms . . . 36

3 A New Eye Contact Correction Algorithm for Large Baseline Tele-presence System 37

3.1 Introduction . . . 37

3.2 System Layout . . . 38

3.3 Eye Contact Correction Solution . . . 39

3.3.1 Stereo Matching . . . 39

3.3.2 Pixel Coloring . . . 40

3.3.3 Foreground Extraction . . . 41

3.3.4 Foreground Interpolation . . . 41

3.3.5 Re-rendering . . . 42

3.4 Extension to Regular Stereo Imaging Problems . . . 44

3.5 Experimental Results . . . 46

3.5.1 Tests for Tele-presence Applications . . . 46

3.5.2 Tests for Regular Stereo Images . . . 46

3.6 Conclusion . . . 52

4 Foreground and Background Separation Using Disparity Map and Image Matting 53

4.1 Foreground and Background Segmentation and the Problem . . . 53

4.2 Literature Review of Image Matting . . . 54

4.3 Proposed Method . . . 58

4.3.1 Automatic Generation of the Trimap . . . 58

4.3.2 Trimap Refinement . . . 58

4.3.3 Estimation of Foreground and Background Colors . . . 59

4.3.4 Energy Function and Global Optimization . . . 62

4.4 Experimental Results . . . 64

4.4.1 Visual Comparisons . . . 64

4.4.2 Quantitative Evaluations . . . 68

4.5 Conclusion . . . 69

5 Illumination Invariant Stereo Matching **70**

5.1 Introduction 70

5.2 Illumination Invariant Stereo Matching 73

5.2.1 Local Radiometric Invariance Stereo Matching 73

5.2.2 Global Illumination Invariance Stereo Matching 74

5.3 Hierarchical Stereo Matching Based on Improved Census Algorithm 75

5.3.1 Improved Census Algorithm 75

5.3.2 Holes filling: Interpolation vs. Image In-painting 77

5.3.3 Main Steps of the Proposed Method 77

5.3.4 Experiments Results 78

5.4 Illumination Invariant Stereo Matching Based on Relative Gradients 83

5.4.1 RG - The Illumination Invariant Feature 83

5.4.2 Matching Cost Function of the Proposed Method 83

5.4.3 Post Processing 84

5.4.4 Experimental Results 84

5.5 Real-time Implementation of the RG Algorithm Using GPU 88

5.6 Conclusion 90

6 Conclusion **92**

6.1 Conclusion 92

6.2 Future Work 95

Bibliography **97**

List of Tables

2.1 The category of eye contact correction solutions 22
2.2 Various types of stereo matching algorithms 29
2.3 The category of view interpolation algorithms 36

4.1 Categories of image matting techniques 56
4.2 Quantitative evaluation of the proposed method 68

5.1 Categories of illumination invariant stereo matching methods 75
5.2 Quantitative analysis using MAE measure excluding the occlusion areas 82
5.3 Error ratio on Middlebury 82
5.4 Processing time (times) 82
5.5 Error ratio (%) comparison of local illumination invariant stereo matching methods(without occlusions) 85
5.6 Processing time of proposed method compared to GPU implementation (unit: second) 90

List of Figures

1.1 CISCO series 3000 tele-presence system[11] 1
1.2 Architecture of the CISCO series 3000 tele-presence system [7] 2
1.3 CISCO tele-presence camera system [2] 2
1.4 System architecture of the MAJIC system [97] 3
1.5 A scene of TELEPORT system [49] 3
1.6 A scene of TELECUBICLES system [33] 4
1.7 A demonstration of VIRTUE system [112] 5
1.8 System structure of Coliseum [22] 5
1.9 Typical architecture of FVV system 6
1.10 Dome and computer system [73][34] 7
1.11 The frame of 8-camera capture system [152] 7
1.12 Camera array of Tanimoto system for scene capture [123] 8
1.13 Conceptual design of the tele-presence system based on modified 3dMD camera technology 10
1.14 Proposed architecture of the eye gaze corrected tele-presence System 10
1.15 Corrected eye-gaze using the proposed algorithm 11
1.16 The input and output of image matting (images from [36]) 12
1.17 Color inconsistencies for corresponding pixels with the same illumination 13

2.1 An example of eye tracker and camera selection in Vertegaal's system [128] 16
2.2 Illustration of how the tent-shaped surface works [69] 16
2.3 A demo of Eye-to-Eye systems: TPT 22 desktop 17
2.4 Simulated eye contact after eye replacement, texture mapping, and head rotation [48] 17
2.5 The block diagram of Yang's tele-presence system [143] 18
2.6 Face segmentation and result demonstration [147] 19
2.7 Ott's systems work [98] 20
2.8 Block diagram of Lei's multi-step view reconstruction system [81] 21
2.9 Yip's systems [144] 21
2.10 The illustration of ordering constraint 23
2.11 Illustration of how to find the local optimization 23
2.12 The illustration of the boundary problem 25
2.13 A failure example of Euclidean distance weighting functions 25
2.14 Stereo matching using Graph Cuts algorithm 27
2.15 Matching two scan lines using dynamic programming 28
2.16 Classification of techniques in image based rendering according to the availability of geometric information [116] 30
2.17 Shape distortion caused by image morphing [113] 30
2.18 Illustration of view morphing [113] 31
2.19 Pre-warping in Huang's method [67] 32
2.20 The procedure of Park's algorithm [99] 34
2.21 Camera array for a Light Field system [83] 35
2.22 st-plane and uv-plane 36

3.1 Image taken by cameras with wide stereo baseline 38
3.2 Modified 3dMD camera system 38
3.3 Tele-presence images digitized by the 3dMD cameras 39
3.4 Flow chart of the proposed method for a tele-presence system with stereo cameras on one side 40
3.5 The relationship of coordinates among three cameras 40
3.6 Interpolation from the left side 43
3.7 Flow chart of the proposed method for general stereo matching 44

3.8 Results of disparity extraction using our algorithm for the tele-presence system . . 47
3.9 Example of view interpolation along the baseline of the two color cameras 48
3.10 A series of interpolated virtual images using the proposed algorithm 49
3.11 Disparity result of the “baby” stereo pair 50
3.12 Disparity result of the “teddy” stereo pair 51
3.13 Comparison of our method with other methods using Middlebury database 51

4.1 Interactions in image matting (images from [134]) 55
4.2 Automatic generation of the trimap using disparity 58
4.3 Trimap refinement 59
4.4 An example of fine trimap refinement 59
4.5 An example of coarse trimap refinement 60
4.6 The distance of the real color and inferred color from the linear relationship [134] . 60
4.7 Illustration of distance from a point to a line in 3D space 61
4.8 The matte and the extracted foreground of the tele-presence system 64
4.9 Mattes of T_2 in trimap levels 0, 4, and 9 65
4.10 Mattes of T_8 in trimap levels 0, 4, and 9 66
4.11 The comparison of our results with state-of-the-arts on partial matte T_2 (from trimap 0) 67
4.12 The comparison of our results with state-of-the-arts on partial matte T_8 (from trimap 0) 68

5.1 Real-world image formation 71
5.2 Spectral response of Sony ICX098BQ CCD used in many firewire cameras (image from [16]) 72
5.3 Similarity comparison: HD. *vs.* MI 76
5.4 Holes filling by interpolation and in-painting 78
5.5 The flowchart for the proposed method 79
5.6 Test images with different illumination conditions (images from [4]) 80
5.7 Result comparison of the “Aloe” image in different illuminations 80
5.8 Result comparison of the “Art” image in different illuminations 81
5.9 Result comparison of the “Book” image in different illuminations 81
5.10 Visual comparison of ANCC [57] with the proposed method 85
5.11 The visual results of the “Art” image for different illuminations 86
5.12 The visual results for the “Book” image for different illuminations 87
5.13 The Middlebury rank of the proposed method 88
5.14 Thread batching [12] 89
5.15 Memory model in CUDA [12] 89
5.16 Data processing between CPU and GPU 90

List of Algorithms

3.1 Pseudo code of the region growing algorithm 45
4.1 Pseudo code of $alpha$-expansion 64

Chapter 1

Introduction

In the last century, when people were excited about having their voice transmitted to thousands of miles away, they couldn't imagine that someday a technology would allow them to communicate as if they were present in the same room. This technology is called tele-presence. Tele-presence is a set of technologies, combined advanced networking, video/audio acquisition/processing systems, and full-size high-resolution displays, that are capable of creating the illusion that all participants are co-located in a same room. Tele-presence systems have been deployed in a wide range of areas, such as education, business, and entertainment. Large-scale commercial versions of tele-presence systems (see Figure 1.1) were introduced by CISCO, Polycom, and HP around 2006, with high-quality cameras located near the display screen allowing participants to view each other at life-size and with some level of eye contact. Tele-presence is becoming a common choice of Fortune 500 companies, allowing them to conduct interviews and high-level business negotiations where non-verbal communication is critical. In most cases, the high cost of these systems can be mitigated by reduced traveling time, fatigue, and the impact of traveling on family life.

Figure 1.2 shows the CISCO tele-presence system's architecture. In this system, each user sends the view on his/her side to all the other participants and at the same time the system decodes the

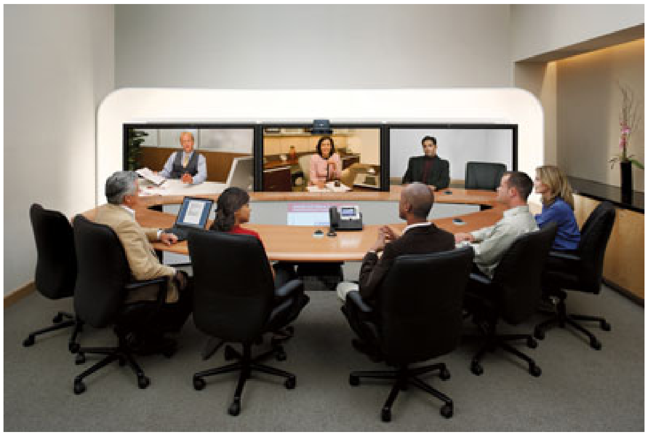


Figure 1.1: CISCO series 3000 tele-presence system[11]

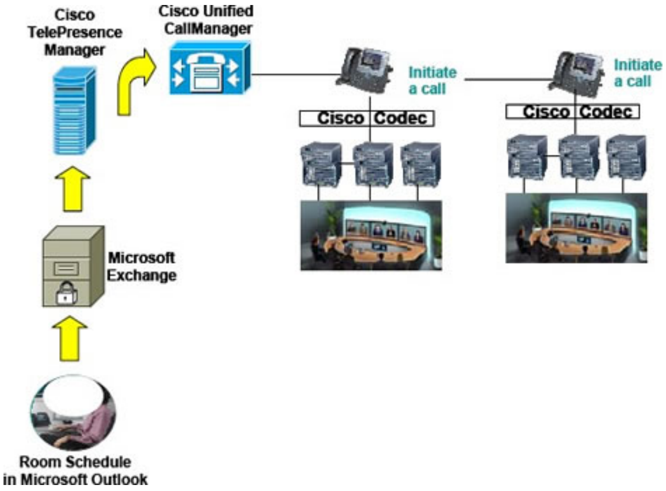

Cisco TelePresence Manager
Cisco Unified CallManager
Initiate a call
Cisco Codec
Initiate a call
Cisco Codec
Microsoft Exchange
Room Schedule in Microsoft Outlook

Figure 1.2: Architecture of the CISCO series 3000 tele-presence system [7]

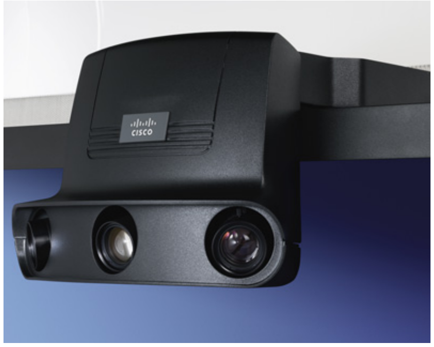


Figure 1.3: CISCO tele-presence camera system [2]

incoming views and displays them on large high-resolution displays.

At the base of a tele-presence, there is the immersive requirement which give the illusion that you are having a face-to-face meeting. In order to create this illusion, the system must be able to maintain eye contact, transmit body gestures, display participants at life-size, and create the illusion of a similar room at all sites. Images taken by cameras directly do not guarantee correct eye contact as usually cameras are located around the display screen and are not at the participant's fixation points (see Figure 1.3 the typical camera mount of the CISCO tele-presence system). Ideally, a true tele-presence system should be able to maintain eye contact by tracking the fixation point of each participant and generating a virtual viewpoint that matches the fixation point. Solving this problem is essential for the future development of tele-presence systems. In the following sections, we will give a short historical review of tele-presence systems and then discuss in more details the motivations and contributions of the thesis.

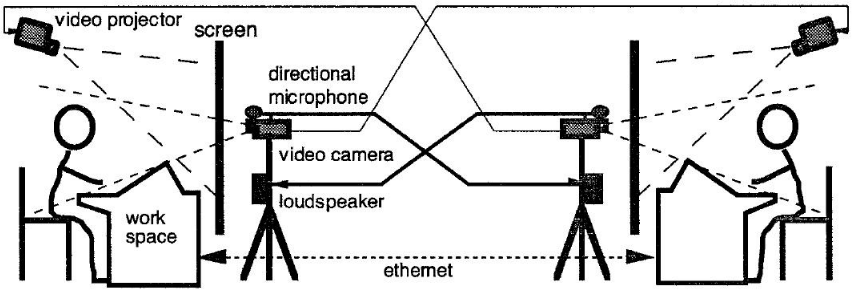


Figure 1.4: System architecture of the MAJIC system [97]

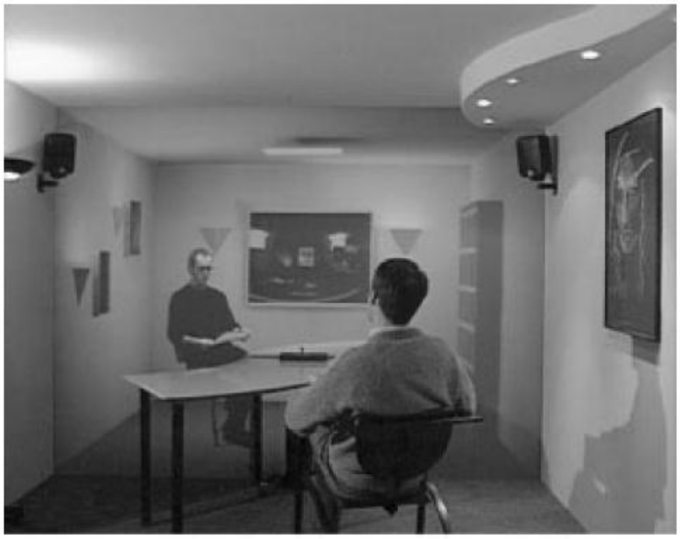


Figure 1.5: A scene of TELEPORT system [49]

1.1 A Brief History of Tele-presence Systems

The word “tele-presence” was first coined by Marvin Minsky in 1980 [94]. Following this pioneering concept, many researchers focused their attention on developing real implementation of tele-presence systems. The system Multi-Attendance Joint Interface for Collaboration (MAJIC) [97] is an early attempt of the concept of tele-presence. This system is composed of a large semi-transparent film made of black and white hexagons that allows the participants to see an image of the other participant projected onto the screen but not the camera aiming directly towards them (see Figure 1.4).

In 1999, another system, MONJUnoCHIE [17], was proposed. The system is able to support eye contact and gaze awareness with the help of a special hologram display called “Glass Vision”. The TELEPORT system developed in 1999, is the first to set all the participants around a virtual table [49]. The outlines of the participants are segmented and located inside a virtual 3D environment projected onto a wall (Figure 1.5).

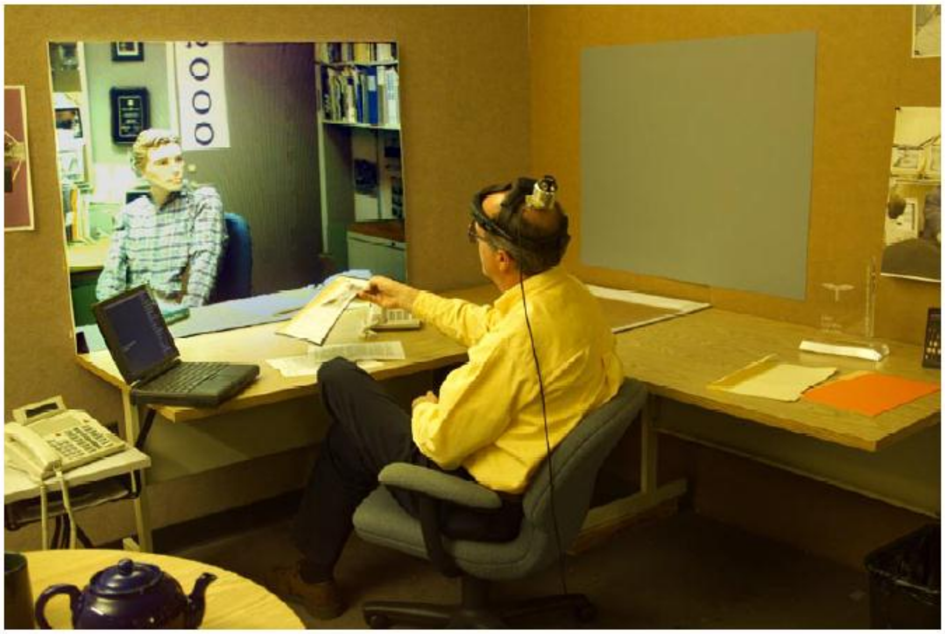


Figure 1.6: A scene of TELECUBICLES system [33]

Although the TELEPORT system is much closer to a real face-to-face meeting, it is still in two-Dimension. In 2002, the National Tele-Immersive Initiative (NTII) in the United States developed a true stereo tele-presence system called TELECUBICLES (see Figure 1.6) [33, 126]. In TELECUBICLES, the background scene is scanned using a laser scanner and rendered at each location. A 3D magnetic tracker is used to determine the participant's head position. A stereo rendering algorithm is used to help with eye contact, immersion, and gaze awareness.

However, TELEPORT and TELECUBICLES require special tracking equipment, and therefore, their practical usage is limited. The VIRtual Team User Environment (VIRTUE) system [112] was developed in 2002 at Delft University of Technology. As shown in Figure 1.7, the main elements of the system are frame integrating cameras, directional speakers, and a 61-inch plasma display. VIRTUE also includes a semi-circular table which is extended as a virtual table into the virtual meeting room to help improve illusion of presence.

Coliseum [22] was proposed in 2002 at Hewlett-Packard (HP) research laboratory. Coliseum constructs a 3D model from the extracted foregrounds of five cameras with a 30° angle between two neighbors (see Figure 1.8). The silhouettes of the foreground are extracted to construct a 3D model using an image-based visual hull algorithm. Then, the 3D model is inserted into a virtual meeting room with a novel viewpoint.

Current solutions to eye contact correction problem in tele-presence systems could be classified as hardware-based and software-based methods. The details of each category will be discussed in Chapter 2. A subcategory of software-based methods, called image-based methods, are mostly

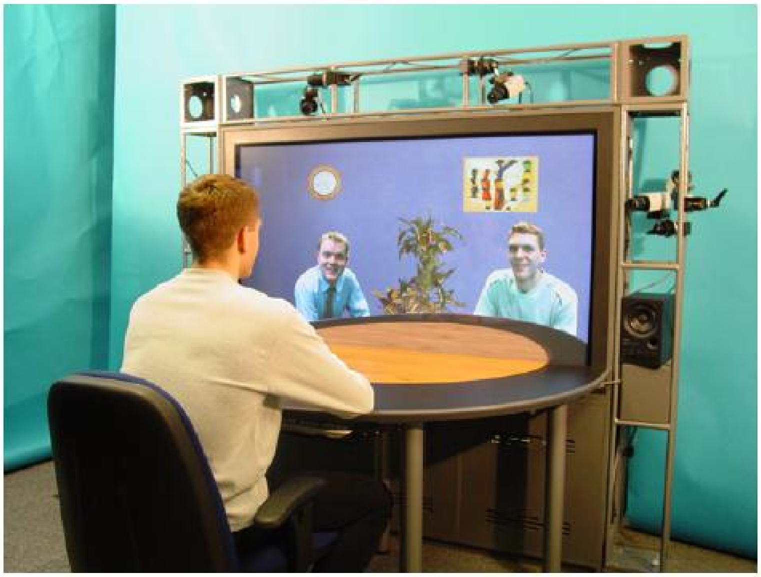


Figure 1.7: A demonstration of VIRTUE system [112]

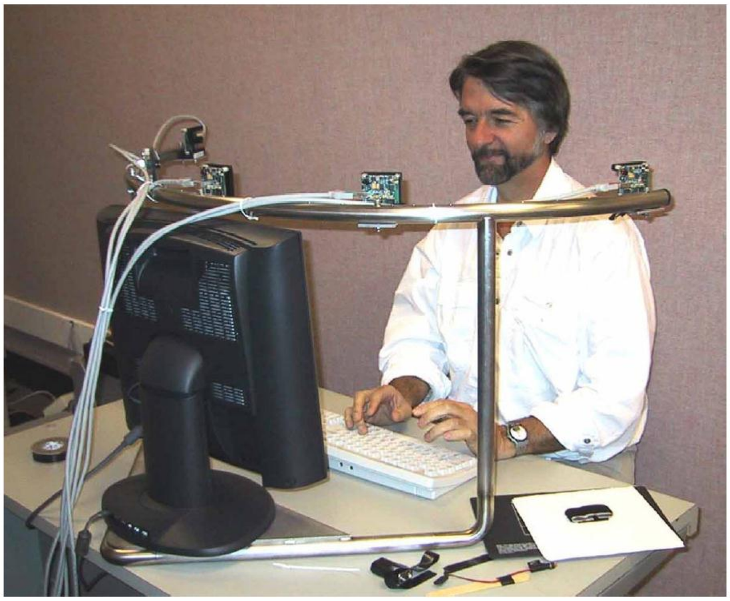


Figure 1.8: System structure of Coliseum [22]

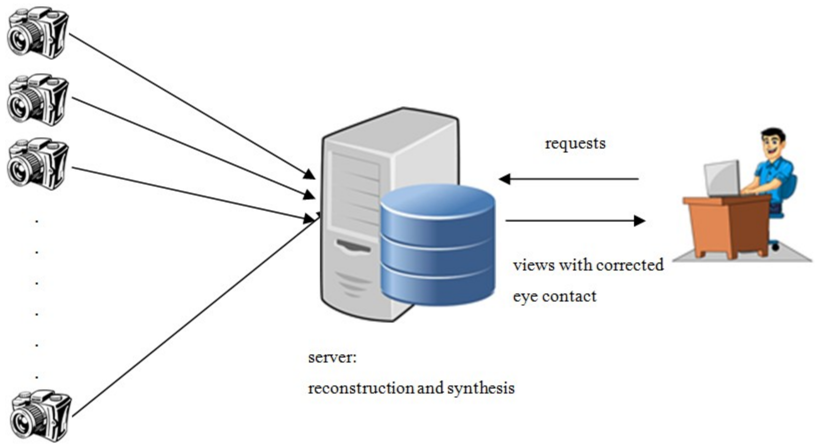


Figure 1.9: Typical architecture of FVV system

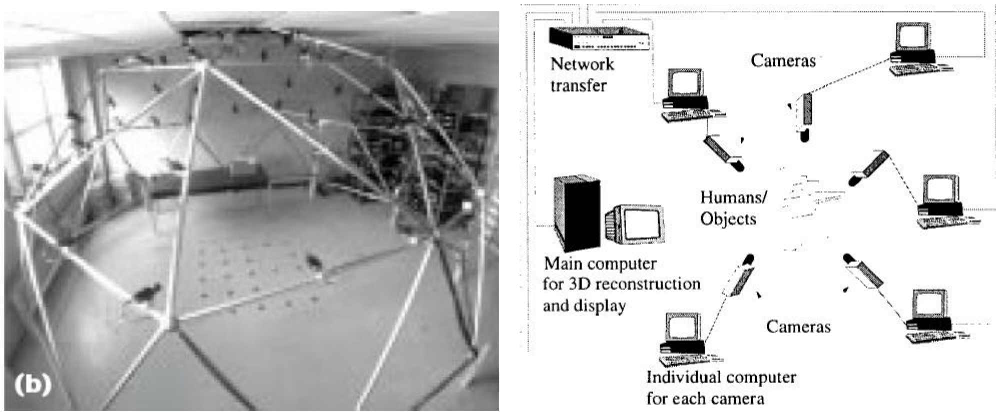
similar to depth-based methods adopted in Free-Viewpoint Video (FVV). In the following section, we will give a short historic introduction of FVV systems.

1.2 A Brief History of Free Viewpoint Video Systems

In FVV systems, multiple cameras capture the scene and send images to a server. The server reconstructs the scene and synthesizes the views to users according to their requests. The ways to reconstruct and represent the scene could be roughly categorized as depth-based, object-based, and ray-based methods. End users can explore a scene from any viewpoint by sending the desired viewpoint to the server. The server then returns the corresponding video stream from the requested viewpoint. Figure 1.9 shows a typical architecture of FVV system.

QuickTime VR [32] may be considered the predecessor of FVV. Several still images are stitched together into a 360-degree static panorama, where a simple texture-based rendering algorithm is used to re-render the views from a new viewpoint. In 1997, Carnegie Mellon University (CMU) built the first version of a true FVV system called the Virtualized Reality system [73]. As illustrated in Figure 1.10 (a), a total of 51 cameras are mounted on a dome to capture a dynamic scene. Stereo matching algorithms are used to calculate the depth information for free viewpoint reconstruction. In 2000, Cheng *et.al* [34] at CMU proposed another system consisting of six PCs with five cameras as illustrated in Figure 1.10(b). One PC works as the master PC and each of the other five PCs is connected to a camera. Each camera takes the silhouette of the moving person from one viewpoint. Five silhouettes are sent to the master PC to reconstruct a 3D model offline.

Microsoft Research proposed an eight-camera system arranged along a 1D arc frame as illustrated in Figure 1.11 [152]. Point Grey Research designs two special concentrator units for Microsoft Research to synchronize the cameras and to store all video streams into a group of hard disks in real time. The multi-view scene reconstruction is based on depth information extracted by multi-view stereo matching. The scene is separated into the main layer and the boundary layer to improve the



(a) Multi-view CMU video dome

(b) Computer configuration of the video dome system

Figure 1.10: Dome and computer system [73][34]

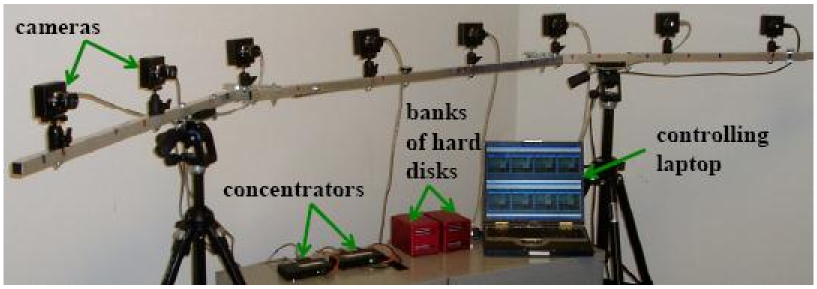


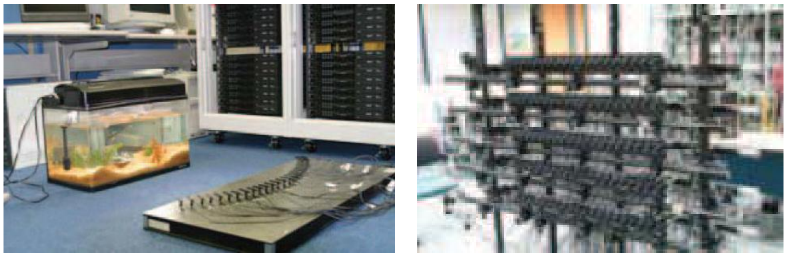
Figure 1.11: The frame of 8-camera capture system [152]

rendering performance.

Free-viewpoint TV (FTV) is an application of FVV for the television transmission of a multi-camera system. Because of its increased complexity, the techniques necessary to create working FTV are still in the research phase. Tanimoto [123] is the first researcher to propose the ray-based FTV with real-time performance in 2001, as illustrated in Figure 1.12. Ray-space records each ray into a 4D-parameter unit. Each ray is a line from the camera optical center and goes through a pixel in the image plane. The line intersects with a 3D voxel whose intensity is the intensity of the pixel in the image plane of the 3D voxel. Ray-based FTV must collect as many rays as possible. Therefore, generally, a dense camera array is used to capture the scene. The quality of the images depends on the density of the camera array.

Based on the ray-space interpolation technique, KDDI Corporation, developed the world's first free-viewpoint video technology for the Web in 2008 [68]. Unfortunately, the size of the cameras limits the density of the camera array. In order to obtain highly sampled ray space, virtual images are interpolated to fill up the gaps between neighboring cameras.

Systems with free viewpoint abilities have two main advantages. First, users have the freedom



(a) 1D arrangement

(b) 2D arrangement

Figure 1.12: Camera array of Tanimoto system for scene capture [123]

to change their viewpoints while traditional systems only provide videos from a fixed viewpoint. Second, free viewpoint choice allows to create views where it is impossible to set real-physical cameras, such as the center of the display screen.

1.3 Motivations and Contributions

As mentioned previously, there are key parameters in tele-presence systems that are essential in order to be able to create the illusion of presence to all participants. In this thesis, we decide to address a subset of those parameters by solving some of these issues automatically using new image processing and computer vision algorithms. They include:

1) Solving the eye contact problem for large tele-presence screens;

2) Developing a new foreground and background separation technology to robustly extract participants and integrate them into a common virtual meeting room;

3) Solving in real time the stereo correspondence from a network of cameras with different illuminations.

1.3.1 Solving the Eye Contact Problem for Large Tele-presence Screens

Eye contact has been considered by many researchers to be the most important factor in immersive communications. Sometimes, eye contact can deliver more information than languages. When we talk to people face-to-face, we always look into their eyes. However, in traditional video conferencing systems, the cameras cannot capture the views with direct correct eye contact. This is because people sitting in front of computers are always staring at the screens where the message windows with remote views show up, but the cameras are mounted on the frame of the display screens, not at the exact positions of the message windows.

As mentioned previously, most successful immersive tele-presence systems use various hardware to solve the eye contact problem, but in some cases the hardware is either too expensive or too large. In addition, hardware-based solutions capture views from only one viewpoint, which is not enough for 3D display. Therefore, in this thesis, we develop a software solution to the eye contact correction problem for tele-presence. A real-time stereo matching algorithm is used to generate a disparity map and the virtual view with correct eye contact can be synthesized by disparity-based interpolation. To improve the accuracy of the disparity maps in low-textured areas, for example for the skin and the hair, a pattern with random dots is projected onto the scene. The hardware involved in this system is easily accessible and can be made in small size. Hence, the proposed system is independent of the location and can even be applied to hand-held mobile devices. The large baseline is also a challenge which may impact on the quality of eye contact correction. Previous systems do not need to worry about this problem because the size of the monitor is small, such as a PC monitor. However, in order to display a life-size image or video, the tele-presence monitors are now quite large. The distance between cameras on the left and right sides is too large to create dense disparity maps. In this thesis, we suggest to set an assistant camera besides each installed stereo camera. With the help of the assistant camera, the cameras on the same side are responsible for creating a partial disparity map of the scene and interpolating the virtual views on the desired viewpoint. All partial virtual views are then blended together to compose a whole view. One can see in Figure 1.13 the proposed camera system and Figure 1.14 shows the processing pipeline developed in this thesis. This solution consists of eight main stages (**S**):

S-1: At least two units of cameras are mounted on the oppsite sides of the monitor (up and down, or left and right). Each unit includes two infrared (IR) cameras, one IR projector, and one color camera.

S-2: The disparity maps are calculated by IR cameras on the same side first.

S-3: The foreground is extracted from the scene for each unit of cameras.

S-4: The disparity map of the foreground is refined and then the disparity maps between two color cameras are inferred.

S-5: Both the color images and their corresponding disparity maps are encoded and sent to all the other remote users through the Internet. Receivers then decodes what they receive.

S-6: The scenes are re-projected to the desired position to correct eye contact, and a novel view is rendered.

S-7: All the corrected foregrounds are merged into a virtual environment, and rendered on the end monitors. We suggest the virtual environment is the scene in front of each user and hidden by the screen.

S-8: During the meeting, users can adjust the viewpoints to watch the people they are talking with through a keyboard or a mouse. The virtual views can be displayed in 2D or 3D. It is awkward to wear glasses during the meeting, so an auto-stereo screen is recommended.

Figure 1.15 shows the multiple images captured by the system and the corrected eye gaze computed using the proposed algorithm.

The proposed solutions have the following advantages:

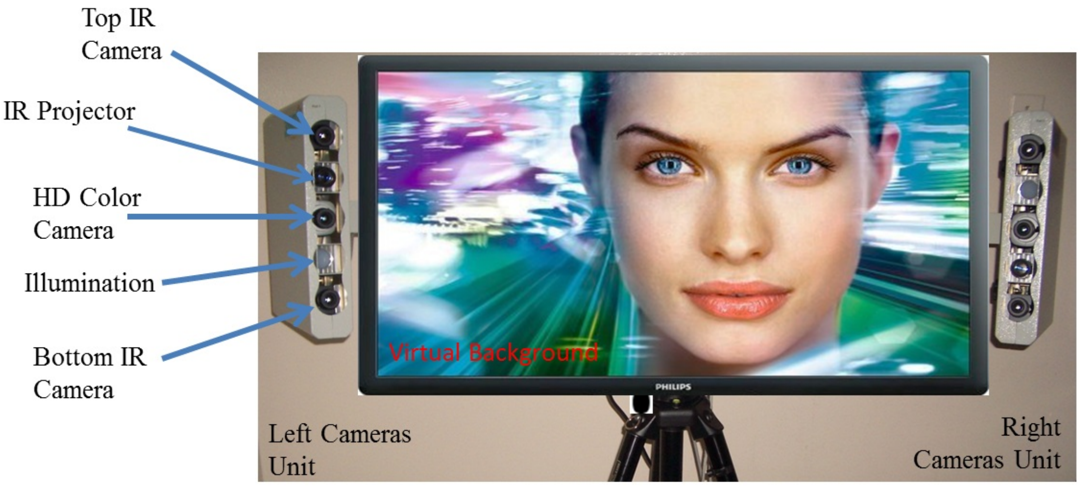


Figure 1.13: Conceptual design of the tele-presence system based on modified 3dMD camera technology

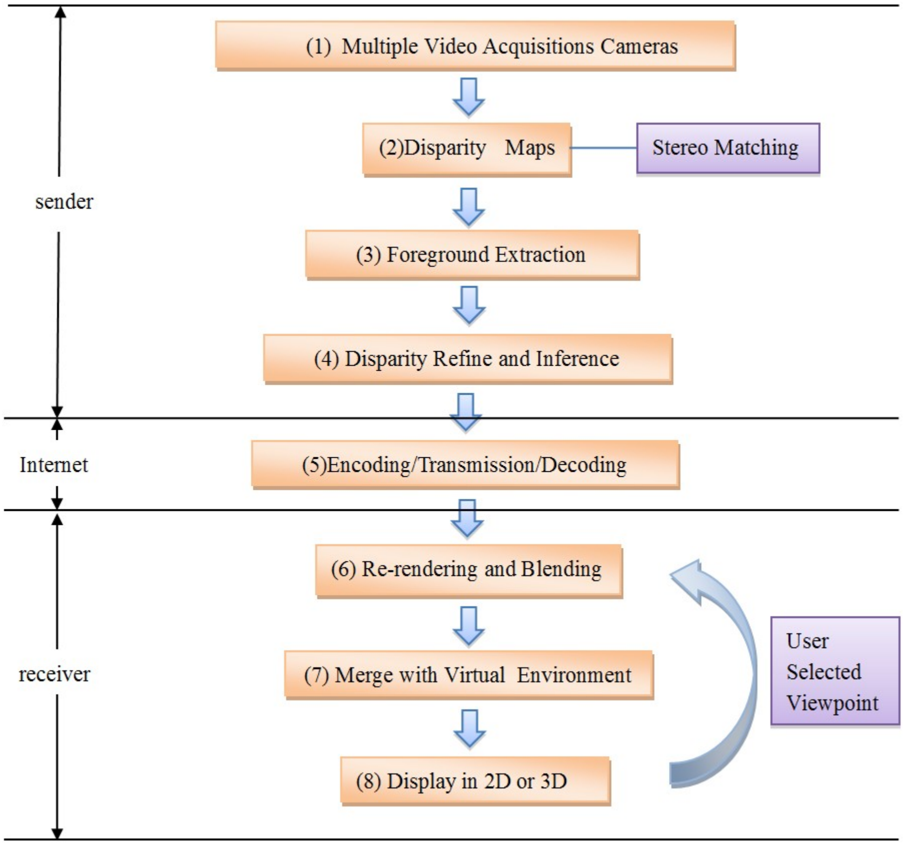
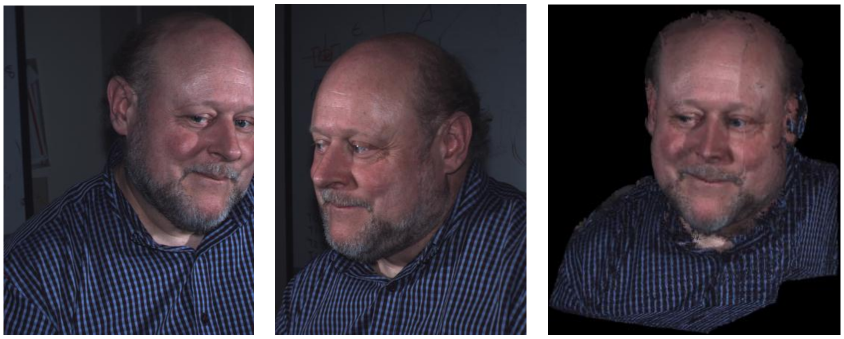

sender
Internet
receiver
(1) Multiple Video Acquisitions Cameras
(2)Disparity Maps
Stereo Matching
(3) Foreground Extraction
(4) Disparity Refine and Inference
(5)Encoding/Transmission/Decoding
(6) Re-rendering and Blending
(7) Merge with Virtual Environment
(8) Display in 2D or 3D
User Selected Viewpoint

Figure 1.14: Proposed architecture of the eye gaze corrected tele-presence System



(a) Real left view (b) Real right view (c) Virtual front view

Figure 1.15: Corrected eye-gaze using the proposed algorithm

1) This system does not require any special hardware, so it is economical to manufacture;

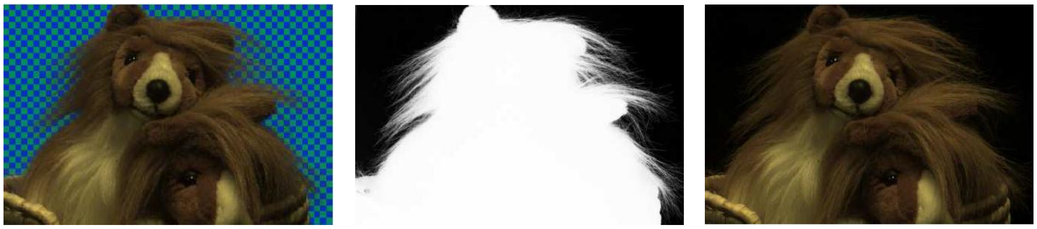
2) Existing systems only send the corrected views to other end users, but our proposed system sends the uncorrected views along with the disparity maps. It allows people to select eye contact which they are comfortable with;

3) This system is easy to implement with real-time performance;

4) This system can handle large baselines generated by large tele-presence displays.

1.3.2 Foreground Extraction by Mutual-Information-based Image Matting

The ability to create the illusion of being in a common meeting room is critical to presence performance. Commercial systems solve this problem by building great-cost similar rooms at all participating sites. A solution where participants could be extracted from their backgrounds and then integrated in a common virtual room could contribute greatly to the cost reduction of tele-presence systems. Previous systems use image segmentation to extract the foreground from its background. However, segmentation cannot completely prevent the tiny or semi-transparent elements, such as hair, from being deleted. The pixel colors of those elements partially belong to the foreground and partially belong to the background. Therefore, we propose using an novel image matting algorithm instead of image segmentation to solve this problem. The result of image matting is called the matte, a grayscale image, which indicates the percentage of a pixel color belongs to the foreground or background. One can see in Figure 1.16 the process of image matting. Figure 1.16(c) shows the extracted foreground. One can see that the hair on the bears are well preserved. Our algorithm starts from a trimap refinement. For better color estimation, we first explore the pixels with unknown alpha values, which can be either foreground or background color samples. Non-parametric color estimation is followed by the trimap refinement to estimate the pair of foreground and background colors best fitting each unknown pixel. An energy function is built on the estimated foreground and background colors. Graph Cuts is used to minimize the energy function, while mutual information



(a) Input color image (b) Matte (c) Extracted foreground

Figure 1.16: The input and output of image matting (images from [36])

[115] works as the metric in the data term.

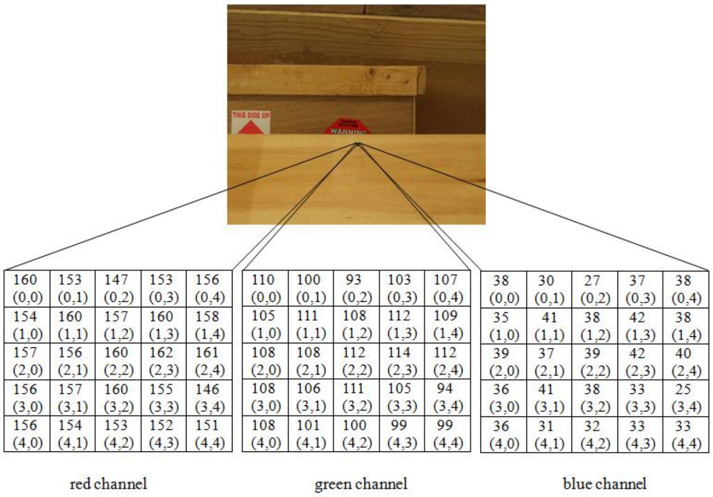
1.3.3 Real-time Illumination Invariant Stereo Matching

The disparity map is one of the most important keys to generate virtual views. Although many stereo matching methods are proposed to improve the disparity map, most are dependent on the colors. In real tele-presence environment, the uneven lighting condition creates a variance between the corresponding pixels, hence the need for an illumination invariant stereo matching algorithm. Later on, we find that even if the stereo images are taken under the same illumination, the corresponding pixels would not be the same. Figure 1.17 shows one example of this situation. The values listed in (a) and (b) are colors from pairs of corresponding pixels, and as one can see they are not the same. There are many reasons for this: reflection from a non-Lambertian surface, camera calibration, the complex lighting distribution, and different camera settings.

In order to improve the accuracy of disparity maps, we propose two radiometric invariant stereo matching methods. One works for a particular non-Lambertian reflection model, while the other is for general illumination changes. Both try to convert the color from the RGB space to another space, in which the values are the same for corresponding pixels. The test images are digitized in different illumination environments. Moreover, in order to integrate the proposed stereo matching to a real tele-presence system, we choose local optimization and Graphic Processing Units (GPU) implementation to make the system run in real time.

1.4 Organization

This thesis is organized as following. Chapter 2 reviews the existing eye contact correction methods in tele-presence systems, as well as two key techniques involved in eye contact correction models: stereo matching and view interpolation. Chapter 3 explains the proposed strategy for handling the eye contact correction problem in the tele-presence systems and how it deals with the large baseline problem. Chapter 4 first reviews the history of image matting and then introduces our proposed method based on mutual information and how it can be used in the tele-presence context. Chapter 5



160 (0,0)	153 (0,1)	147 (0,2)	153 (0,3)	156 (0,4)
154 (1,0)	160 (1,1)	157 (1,2)	160 (1,3)	158 (1,4)
157 (2,0)	156 (2,1)	160 (2,2)	162 (2,3)	161 (2,4)
156 (3,0)	157 (3,1)	160 (3,2)	155 (3,3)	146 (3,4)
156 (4,0)	154 (4,1)	153 (4,2)	152 (4,3)	151 (4,4)

red channel

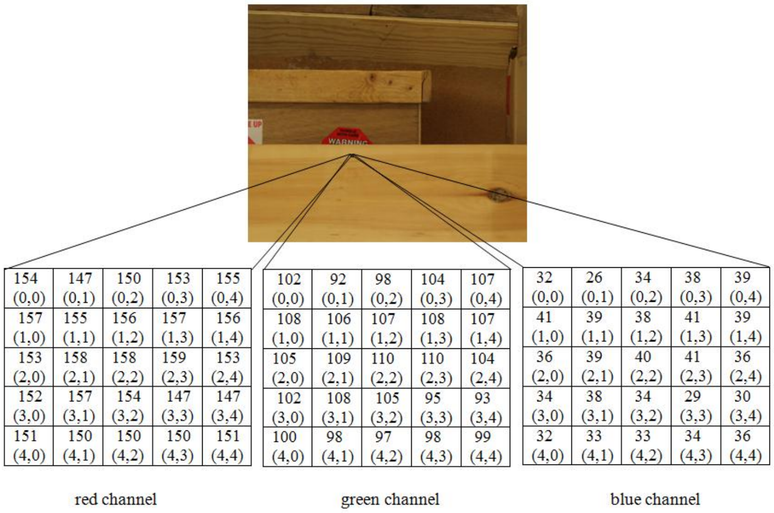
110 (0,0)	100 (0,1)	93 (0,2)	103 (0,3)	107 (0,4)
105 (1,0)	111 (1,1)	108 (1,2)	112 (1,3)	109 (1,4)
108 (2,0)	108 (2,1)	112 (2,2)	114 (2,3)	112 (2,4)
108 (3,0)	106 (3,1)	111 (3,2)	105 (3,3)	94 (3,4)
108 (4,0)	101 (4,1)	100 (4,2)	99 (4,3)	99 (4,4)

green channel

38 (0,0)	30 (0,1)	27 (0,2)	37 (0,3)	38 (0,4)
35 (1,0)	41 (1,1)	38 (1,2)	42 (1,3)	38 (1,4)
39 (2,0)	37 (2,1)	39 (2,2)	42 (2,3)	40 (2,4)
36 (3,0)	41 (3,1)	38 (3,2)	33 (3,3)	25 (3,4)
36 (4,0)	31 (4,1)	32 (4,2)	33 (4,3)	33 (4,4)

blue channel

(a) Colors in a 5 x 5 window in the left image



154 (0,0)	147 (0,1)	150 (0,2)	153 (0,3)	155 (0,4)
157 (1,0)	155 (1,1)	156 (1,2)	157 (1,3)	156 (1,4)
153 (2,0)	158 (2,1)	158 (2,2)	159 (2,3)	153 (2,4)
152 (3,0)	157 (3,1)	154 (3,2)	147 (3,3)	147 (3,4)
151 (4,0)	150 (4,1)	150 (4,2)	150 (4,3)	151 (4,4)

red channel

102 (0,0)	92 (0,1)	98 (0,2)	104 (0,3)	107 (0,4)
108 (1,0)	106 (1,1)	107 (1,2)	108 (1,3)	107 (1,4)
105 (2,0)	109 (2,1)	110 (2,2)	110 (2,3)	104 (2,4)
102 (3,0)	108 (3,1)	105 (3,2)	95 (3,3)	93 (3,4)
100 (4,0)	98 (4,1)	97 (4,2)	98 (4,3)	99 (4,4)

green channel

32 (0,0)	26 (0,1)	34 (0,2)	38 (0,3)	39 (0,4)
41 (1,0)	39 (1,1)	38 (1,2)	41 (1,3)	39 (1,4)
36 (2,0)	39 (2,1)	40 (2,2)	41 (2,3)	36 (2,4)
34 (3,0)	38 (3,1)	34 (3,2)	29 (3,3)	30 (3,4)
32 (4,0)	33 (4,1)	33 (4,2)	34 (4,3)	36 (4,4)

blue channel

(b) Colors in a 5 x 5 window in the right image

Figure 1.17: Color inconsistencies for corresponding pixels with the same illumination

introduces two new illumination invariant stereo matching techniques and the real-time implementation using GPU. Chapter 6 concludes the thesis and discusses some possible future research work and improvements.

Chapter 2

Literature Review

This chapter reviews the literature on eye contact correction in tele-presence system. We also review the state-of-the-art of the technologies involved to solve this problem: stereo matching and view interpolation.

2.1 Eye Contact Correction in Tele-presence Systems

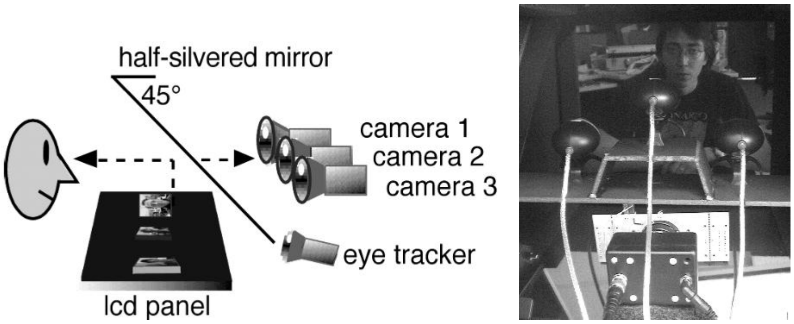
One of the important components of tele-presence systems is to create the illusion that the participants are having a virtual face-to-face meeting. In order to do so, a virtual image must be generated from the virtual cameras located around the screen where the participant is looking. This process is called gaze correction or eye contact correction in the literature. Solutions for the eye contact correction problem can be categorized as hardware-based and software-based.

2.1.1 Hardware-based Solutions for Eye Contact Correction

In the early days of tele-presence, researchers solved the eye contact problem by using some special optical hardware, such as half-silvered mirrors or semi-transparent mirrors, to split the optical paths between the display screens and the cameras [27, 96, 97]. To avoid losing eye contact, Vertegaal *et al.* [128] set a group of cameras behind a half-silvered mirror and used an eye tracker to detect the direction of the eye gaze (see Figure 2.1). The camera closest to the gaze direction is selected and broadcast the views. However, this system can only provide approximate eye contact as there is no way to extract view in-between the discrete cameras.

Jones *et al.* [Jones 09] scan the face of a participant using a real-time 3D range sensor and send the 3D range data with the associate color texture to the other participants. At each terminal, a high-speed optical system projects the 3D data onto an auto stereoscopic display where a rotating two-sided reflective surface in the shape of a tent (see Figure 2.2(a)) is used to create the illusion of an holographic display. The images that appear on the surface (see Figure 2.2(b)) allow for 180-degree field-of-view visualization without glasses at full resolution.

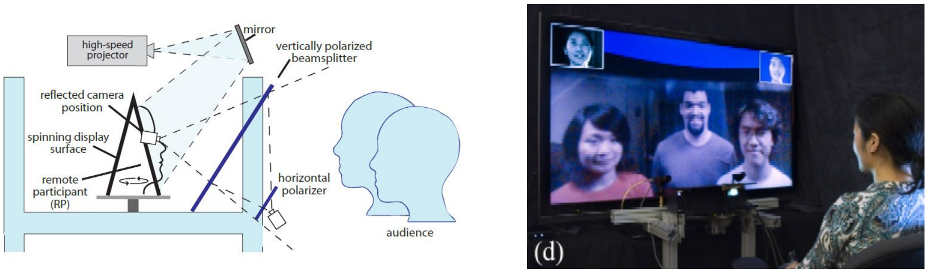
Hardware-based solutions can provide results very quickly, because they are generally based



(a) The model of hardware setting

(b) A video chat scene

Figure 2.1: An example of eye tracker and camera selection in Vertegaal's system [128]



(a) Tent-shaped surface

(b) The scene of two-part chatting

Figure 2.2: Illustration of how the tent-shaped surface works [69]

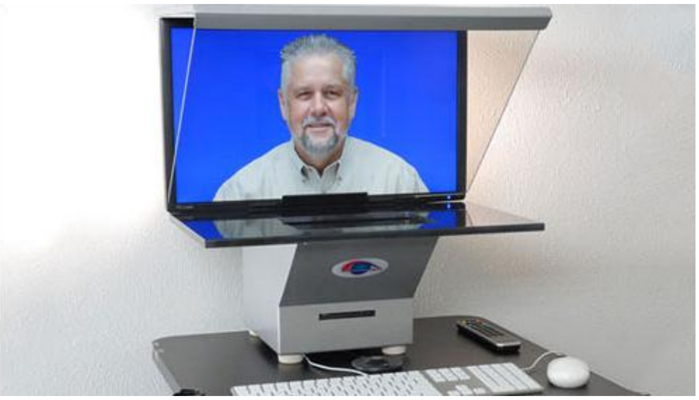


Figure 2.3: A demo of Eye-to-Eye systems: TPT 22 desktop
(from: http://www.telepresencetech.com/tpt22-desktop)

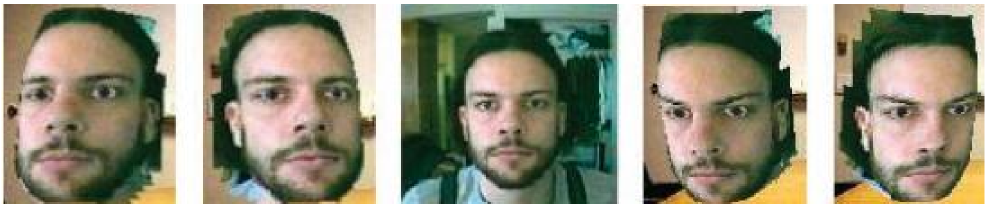


Figure 2.4: Simulated eye contact after eye replacement, texture mapping, and head rotation [48]

on light-splitting techniques. Most successful commercial products use this technique to correct the eye contact, since it can provide the most robust results. For example, the Eye-to-Eye systems (see Figure 2.3) sold by TelePresence Tech use a half-silvered mirror in front of the display screen. However, the hardware used is not easily accessible, and sometimes expensive, bulky and needs a large set-up space.

2.1.2 Software-based Solutions of Eye Contact Correction

Other research has focused on software-based solutions. These solutions can be classified as model-based methods and image-based methods. Model-based methods map a participant's face onto a pre-built face model while image-based methods create virtual views with depth information.

Model-based Eye Contact Correction

Gemmell *et al.* [48] think that the positions of the eyelids and pupils, as well as the head orientation, are sufficient to correct facial expressions. In their system, the eyes are first segmented from the video frame and the original eyes are replaced by the synthesized eyes in the desired gaze direction. Then, all the video frames with the correct eye gaze are mapped onto a face model and the model is rotated to the desired orientation. Figure 2.4 shows some examples of simulated faces looking up, down, left, and right.

The head position is detected by a face model and stereo cameras in [143]. The parts other than

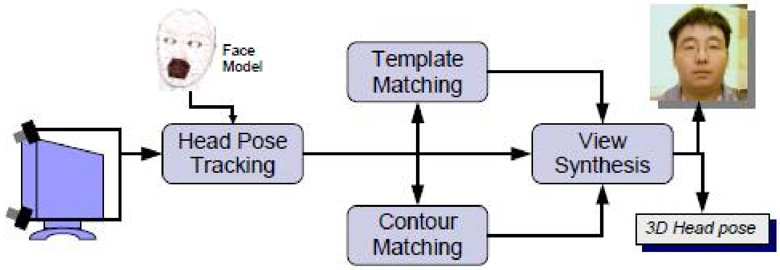


Figure 2.5: The block diagram of Yang's tele-presence system [143]

faces in the images are matched by a feature-and-silhouette-based stereo algorithm. The virtual view is synthesized through either view morphing or hardware-assisted rendering. Figure 2.5 shows the block diagram of Yang's system. Since the stereo pairs improve the accuracy of the head tracking, this method produces better results than pure model-based method. However, it requires human interactions in the calibration and model acquisition steps.

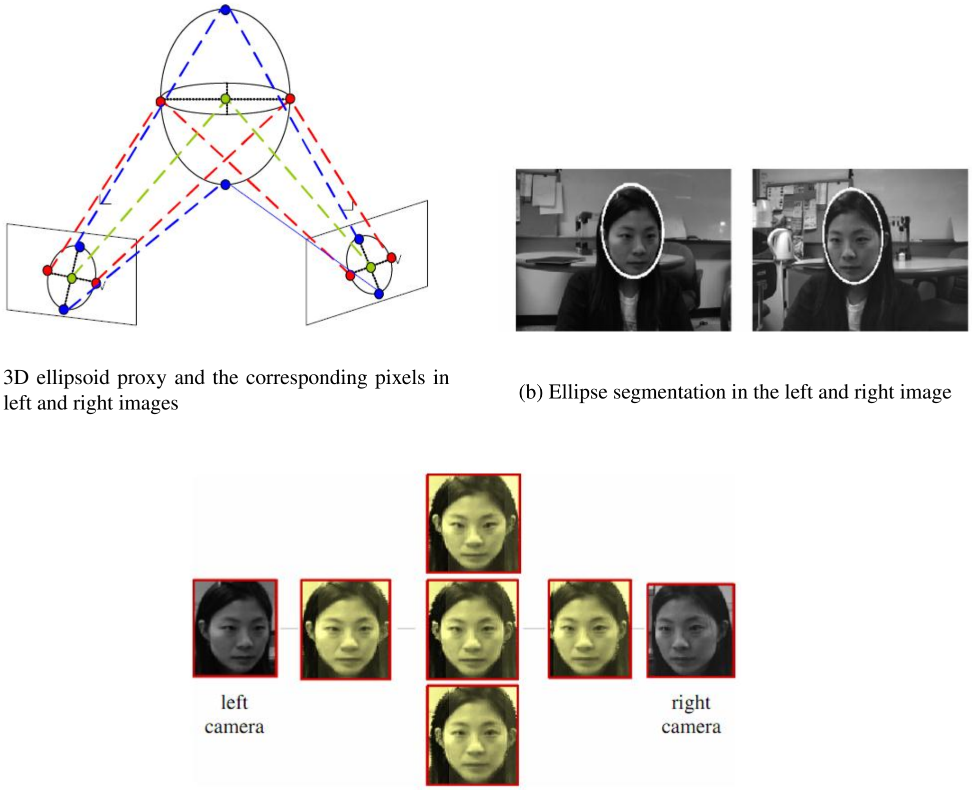
Cham *et al.* [30] register the user's head with a reference face and then map the face parameters to a face image with the correct eye gaze. Yoon *et al.* [147] use a 3D ellipsoid proxy to simulate the head model. The participant's face in each frame is segmented within a 2D ellipse template (Figure 2.6(b)) and mapped to the 3D ellipsoid by matching the minor and major axis of the 2D ellipses (see Figure 2.6(a)). Then, rays are casted from the optical center of the virtual camera to each pixel on the virtual image plane. If the ray intersects the ellipsoid model, then the algorithm checks the corresponding pixels in the left and right images. The color assigned to this pixel in the virtual image is a blend of the colors of two corresponding pixels (Figure 2.6(c)).

Model-based methods usually create an explicit face model and merge multiple images onto the model. However, it is difficult to represent complicated facial expressions with parameterized face models as each real face has distinct features. The distortion is the most obvious artifact. One more reason that model-based methods are not always successful is that in addition, to the face, the view of a participant also includes the neck, upper body and hands, and these body parts are also important in communication as well.

Image-Based Eye Contact Correction

Ott *et al.* [98] suggest mounting two cameras on the top and bottom sides of a monitor. The system diagram is shown in Figure 2.7(a). By computing a disparity map from the two cameras, a virtual image can be interpolated as if a camera was at the center of the screen. Figure 2.7(b) shows a sample result of Ott's system.

Liu *et al.* [87] also compute the disparity map of images first, but from three cameras instead of two. They use a feature-based multistage tri-ocular algorithm to compute the disparity map. The virtual view is synthesized from the images mapped from the left and right reference images by

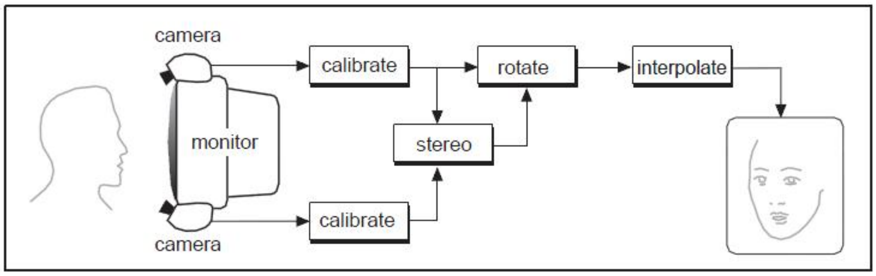


(a) 3D ellipsoid proxy and the corresponding pixels in the left and right images

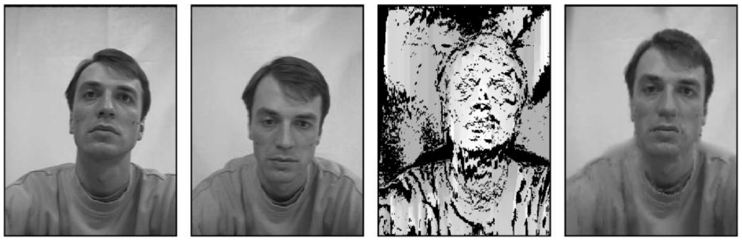
(b) Ellipse segmentation in the left and right image

(c) Simulated results

Figure 2.6: Face segmentation and result demonstration [147]



(a) Ott's tele-presence system



(b) An example of the result. From left to right: bottom view, top view, disparity map, and virtual image

Figure 2.7: Ott's systems work [98]

interpolation. The third camera is only used to generate the disparity map.

Lei *et al.* [81] proposed a multi-step view reconstruction system with known camera geometry. According to the disparity information, the new view is interpolated by two rectified images along the X-axis in order to move the virtual camera to the location $[L_x, 0, 0]$. Then, the new view is extrapolated in the Y-axis to simulate a virtual camera in $[L_x, L_y, 0]$. At last, the Z value is adjusted to move the virtual camera to the desired position $[L_x, L_y, L_z]$. The final view is ready after de-rectification. Figure 2.8 shows a block diagram of Lei's system.

Criminisi *et al.*[42] use an improved dynamic programming stereo matching algorithm to obtain a dense disparity map of the head. Once the disparity map is ready, the corresponding pixels are projected back onto the location of the virtual camera. Yip *et al.* [144] set one camera on top of the monitor and the other one at the desired place (usually the center) on the monitor and calculate the affine transform matrix from the top camera to the center camera (see Figure 2.9). During the video conference, only the top camera is used. The images from the top camera are transformed to the position of the front camera by an affine transform matrix and the eyes are rectified by an eye model. A large occlusion areas appear after transformation because only one camera is used to capture the scene.

Image-based methods are more general as they work not only for faces, but also for the body and other objects in the scenes. These methods interpolate virtual images directly from color images

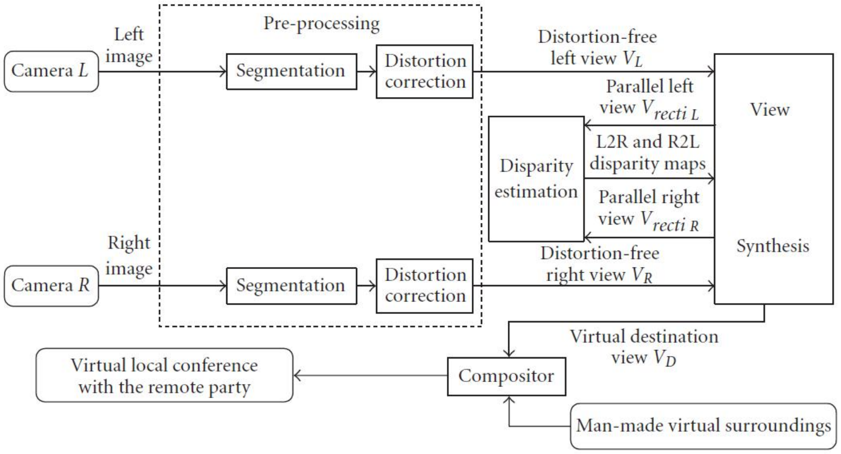

Pre-processing
Camera L
Left image
Segmentation
Distortion correction
Camera R
Right image
Segmentation
Distortion correction
Distortion-free left view V_L
Parallel left view $V_{recti\ L}$
Disparity estimation
L2R and R2L disparity maps
Parallel right view $V_{recti\ R}$
Distortion-free right view V_R
View Synthesis
Virtual destination view V_D
Compositor
Virtual local conference with the remote party
Man-made virtual surroundings

Figure 2.8: Block diagram of Lei's multi-step view reconstruction system [81]

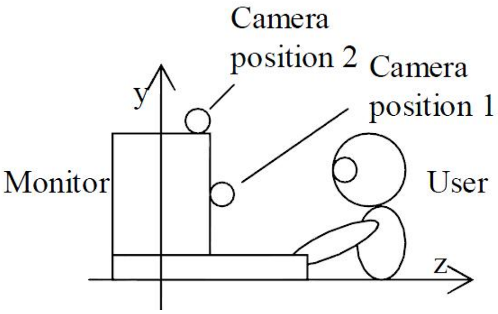


Figure 2.9: Yip's systems [144]

Table 2.1: The category of eye contact correction solutions

Hardware-based	Software-based	
	Model-based	Image-based
Buxton 92[27] Nakazawa 93 [96] Okada 94[97] Jones 09 [69]	Gemmell 00 [48] Yang 02 [143] Cham 02[30] Yoon 05 [147]	Ott 93[98] Liu 95 [87] Lei 02[81] Criminisi 03 [42] Yip 05 [144]

guided by correspondence information. No explicit parametric model is required. Therefore, image-based methods usually need fewer reference images than model-based methods. However, they suffer from occlusion problems created by large baselines between the cameras. The key to image-based methods for tele-presence is how to generate high-quality disparity maps in real time.

2.1.3 Comparison of Eye Contact Correction Methods

Table 2.1 lists the classification of methods discussed previously. Although hardware-based methods are still the mainstream in the commercial products, their high cost favors software solutions. Compared to model-based methods, image-based methods do not require complicated hardware and are computationally lighter. Image-based methods are the most general among the three categories. These are the reasons we decided to choose an image-based method to solve the eye contact correction problem in our system. The general steps of image-based eye contact correction methods are image rectification, stereo matching, and view interpolation. Image rectification is a geometry transform which rectifies multiple images so that the corresponding pixels are on the same scan-line. Stereo matching is the technique which calculates the disparity map from rectified images. The disparity maps provide the correspondence information. View interpolation is concerned with how to create a virtual image from a desired viewpoint. Let's now review the state-of-the-art of stereo matching and view interpolation.

2.2 Stereo Matching

Stereo matching looks for corresponding pixels from two images and then calculates the depth to a reference point, *i.e.*, cameras, using the triangulation rule [55]. It is a key technique in 3D model reconstruction, view synthesis, 3D modeling, and some other applications where depth information is necessary. Depending on which optimization methods are used, stereo matching algorithms can be classified as local and global [111].

There are five main constraints in stereo matching:

1) ***Similarity*** [52] : the corresponding pixels share the similar intensities in all images.

2) ***Ordering*** [21] :on the same epipolar line, if the projection of a pixel **P** (**P**$_{\mathbf{1}}$) is on the left/right of the projection of the other pixel **Q** (**Q**$_{\mathbf{1}}$) in one image, the corresponding projection of **P** (**P**$_{\mathbf{2}}$) also appears on the left/right of the projection of **Q** (**Q**$_{\mathbf{2}}$) in all the other images (see Figure 2.10).

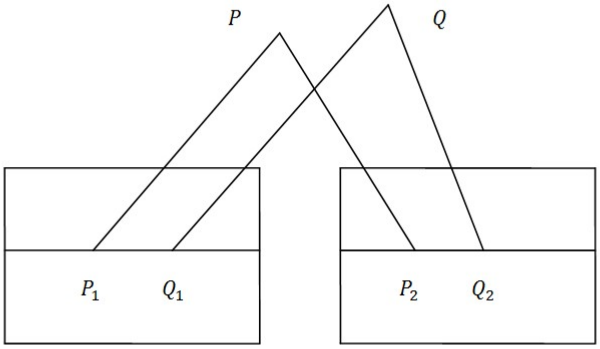


Figure 2.10: The illustration of ordering constraint

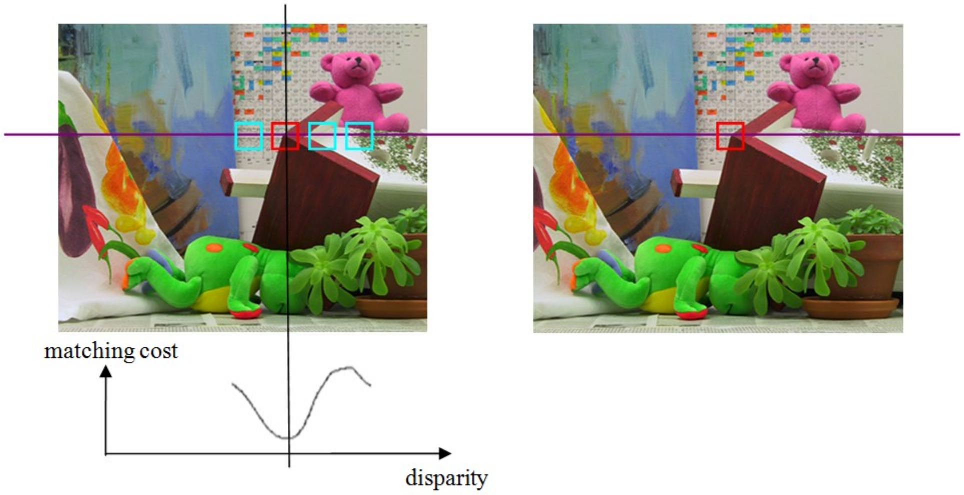


Figure 2.11: Illustration of how to find the local optimization

3) ***Epipolar*** [55] : the corresponding pixels should on the same epipolar line.

4) ***Uniqueness*** [88] : "each item from each image may be assigned at most one disparity value".

5) ***Continuity*** [88] : "disparity varies smoothly almost everywhere".

In the following sections, we will review the existing stereo matching algorithms and analyze how these assumptions affect the design.

2.2.1 Local Stereo Matching

Local stereo matching methods use two windows: one in the reference image and the other in the target image. Thus, local stereo matching is also called window/area/block-based stereo matching. It assumes that the disparity of a pixel is similar to that of neighbor pixels in the same surrounding window. For each window $W_L(i, j)$ in the reference (left) image ((i, j) is the coordinate of the central pixel of a window located at column i and row j), we compute the matching cost of the window with a series of windows centered at $(i - d, j)$ on the same scan-line in the target image, where d is an integer between 0 and the maximum disparity value (see Figure 2.11). If a window

$W_R(i-d,j)$ has the minimal matching cost with $W_L(i,j)$, $|d|$ is assigned as the disparity of (i,j) in the reference image. The optimization of local stereo matching is usually based on the Winner-Takes-All (WTA) strategy [111].

Local stereo matching methods rely on two factors: the matching cost functions and the size of the local window. The matching cost functions decide the similarity of two windows. Some commonly used matching cost functions [60] are:

Sum of Absolute Differences (ASD):

$$SAD = \sum_{\mathbf{q}\in W_{\mathbf{p}}} |\mathbf{I_L}(\mathbf{q}) - \mathbf{I_R}(\mathbf{q}+\mathbf{d})|. \tag{2.1}$$

Sum of Squared Differences (SSD):

$$SSD = \sum_{\mathbf{q}\in W_{\mathbf{p}}} (\mathbf{I_L}(\mathbf{q}) - \mathbf{I_R}(\mathbf{q}+\mathbf{d}))^2. \tag{2.2}$$

Normalized Cross Correlation (NCC):

$$NCC(\mathbf{p},\mathbf{d}) = \frac{\sum_{\mathbf{q}\in W_{\mathbf{p}}}(\mathbf{I_L}(\mathbf{q}))(\mathbf{I_R}(\mathbf{q}-\mathbf{d}))}{\sqrt{\sum_{\mathbf{q}\in W_{\mathbf{p}}}((\mathbf{I_L}(\mathbf{q}))^2 \sum_{\mathbf{q}\in W_{\mathbf{p}}}(\mathbf{I_R}(\mathbf{q}-\mathbf{d}))^2)}}. \tag{2.3}$$

Zero-mean Normalized Cross Correlation (ZNCC):

$$ZNCC(\mathbf{p},\mathbf{d}) = \frac{\sum_{\mathbf{q}\in W_{\mathbf{p}}}(\mathbf{I_L}(\mathbf{q})-\mathbf{I_L}(\mathbf{p}))(\mathbf{I_R}(\mathbf{q}-\mathbf{d})-\mathbf{I_R}(\mathbf{p}-\mathbf{d}))}{\sqrt{\sum_{\mathbf{q}\in W_{\mathbf{p}}}((\mathbf{I_L}(\mathbf{q})-\mathbf{I_L}(\mathbf{p}))^2 \sum_{\mathbf{q}\in W_{\mathbf{p}}}(\mathbf{I_R}(\mathbf{q}-\mathbf{d})-\mathbf{I_R}(\mathbf{p}-\mathbf{d}))^2)}}. \tag{2.4}$$

The parameter $W_{\mathbf{p}}$ defines a local window centered at $\mathbf{p}$. $\mathbf{I_L}(\mathbf{p})$ and $\mathbf{I_R}(\mathbf{p})$ are the colors of the pixel $\mathbf{p}$ in the left and right images. In addition, the size of the local window is also a big issue. If the window is too small, the signal-to-noise ratio will be low hence degrading the results [72]. In general, large windows should be a good choice, but large windows always include pixels on other disparity planes, especially along the boundaries. For example, in Figure 2.12, the red windows contain the disparity planes of both the roof and the background, and the central pixels are located on the roof. Since only one disparity value is assigned to each pair of windows, even if the disparity of the roof is assigned to this pair of windows, the mismatches of the background pixels will increase the matching cost. This problem is called the boundary problem or fattening effect.

Therefore, several adaptive window algorithms have been proposed by adjusting the shape and size of windows according to local textures. Kanade *et al.* [72] developed a statistical model to calculate the uncertainty of a disparity value for a pixel. The disparity is estimated by searching the window with minimun uncertainty. Veksler *et al.* [127] proposed a novel matching cost function which consists of the average error in a window, a bias to smaller variance of the errors in a window, and a bias term for larger windows. The integral image technique [43, 130] is used to speed up the search. Yoon *et al.* [145] assign a supporting weight for each pixel in a window based on the Gestalt principle [1]. In Tombari's work [125], the weighting function is similar to the one in the

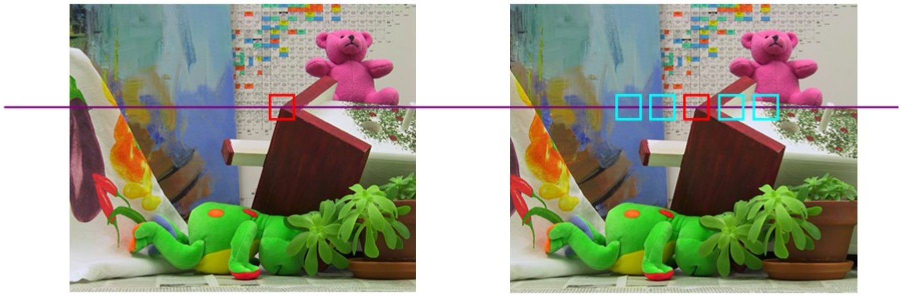


Figure 2.12: The illustration of the boundary problem

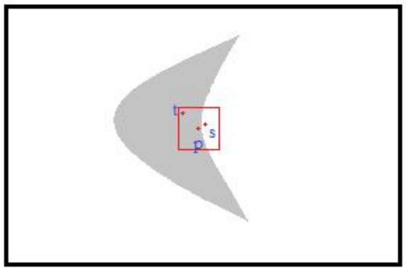


Figure 2.13: A failure example of Euclidean distance weighting functions

Yoon's paper [145], but the weights are assigned as "1.0" if the pixels are in the same segment with the central pixel. Yoon and Kweon [146] design a new matching cost function called the "Distinctive Similarity Measure" (DSM) to compare the similarity of two windows. Brockers *et al.* [25] combine both color distance and spatial distance in the weighting function to select a adaptive weight for cooperative optimization. The color distance is measured in CIELab color space [139]. Tombari *et al.* [15] use segmentation to assist the build of matching cost function which includes two parts: C_s and C_w. C_s is the sum of the Truncated Absolute Differences (TAD) over a segment and C_w is the sum of TAD over a window. Hosni *et al.* [64] proposed to use geodesic distance to replace the spatial distance in the weighting functions.

All the proposed weighting functions could be categorized into three categories according to the measurement of the pixel distance: space distance, color distance, or both. Space-distance-based weighting functions assume that the further a pixel is from the window center, the less likely it is for that pixel to exist on the same disparity plane with the central pixel. However, this is not always true. For example, in Figure 2.13, **p** and **t** are on the same disparity plane and should be assigned a large weight, while **s** should be ignored. However, space-distance-based weighting functions give a large weight to **s** but small weight to **t**. Color-based weighting functions assume that the discontinuities of the color are the potential discontinuities of the disparity plane. Such weighting functions work in most cases, but fail in highly textured areas. In highly textured areas, only a small amount of pixels are chosen in the calculation of the matching cost so that the signal-to-noise ratio is low. Weighting functions based on both the color and the space distance perform better than a single measurement, but are still not a perfect solution. Moreover, local stereo matching fails when there are regions with less textures. This is due to the fact that more than one pair of windows have the same matching cost. This problem can be solved by increasing the window size or using color segmentation [145].

2.2.2 Global Stereo Matching

Global stereo matching tries to find a disparity assignment for each pixel which will minimize a global matching cost function. Cooperative Stereo [88] is the first global algorithm proposed to solve the stereo problem and defines two constraints: uniqueness and continuity. A 3D disparity space of dimensions $(image\ width\ u) * (image\ height\ v) * (disparity\ range\ d)$ is built, where each element (u, v, d) corresponds to the pixel (u, v) in the left image and pixel $(u + d, v)$ in the right image. After setting initial values for the 3D disparity space, each pixel is iteratively updated (u, v, d) according to the match values until convergence. For each pixel (u, v), the disparity d is the element (u, v, d) with the maximum match value.

In a similar framework, the stereo matching problem can also be casted as a probabilistic function, such as the Bayesian probability formulation [80, 23]:

$$P(d_L \mid \mathbf{I_L}, \mathbf{I_R}) \propto P(\mathbf{I_L}, \mathbf{I_R} \mid d_L)P(d_L), \tag{2.5}$$

where the posterior probability $P(d_L \mid \mathbf{I_L}, \mathbf{I_R})$ is defined by the likelihood of a pixel (i, j) on image $\mathbf{I_L}$ to predict a corresponding pixel in image $\mathbf{I_R}$ with a shift $d_L(i, j)$ along the same epipolar line. In some papers, this optimization problem is solved using simulated annealing techniques [23]. Another way to solve the global optimization problem is to use Markov Random Field (MRF) methods [85]. Pixels are formatted as the nodes in the MRF where an energy function is defined. The energy function contains two terms - the data term E_{data} and the smoothness E_{smooth} term [24]:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d). \tag{2.6}$$

The data term tests how appropriate disparities d fit the similarity assumption:

$$E_{data}(d) = \sum_{(u,v) \in I} \mathbf{C}(u, v, d). \tag{2.7}$$

For example, $\mathbf{C}(u, v, d)$ can be any matching cost functions SSD or SAD.

The smoothness term is one more constraint added to ensure the continuity of the disparity map. It is based on the constraint that the adjacent pixels should move smoothly, thus solving some of the issues about occlusions and low texture areas. Some typical smoothness terms are:

1) Quadratic: $(d_\mathbf{p} - d_\mathbf{q})^2$;

2) $L1$: $|d_\mathbf{p} - d_\mathbf{q}|$;

3) Potts model: $V(d_\mathbf{p}, d_\mathbf{q}) = K\delta(d_\mathbf{p} \neq d_\mathbf{q})$.

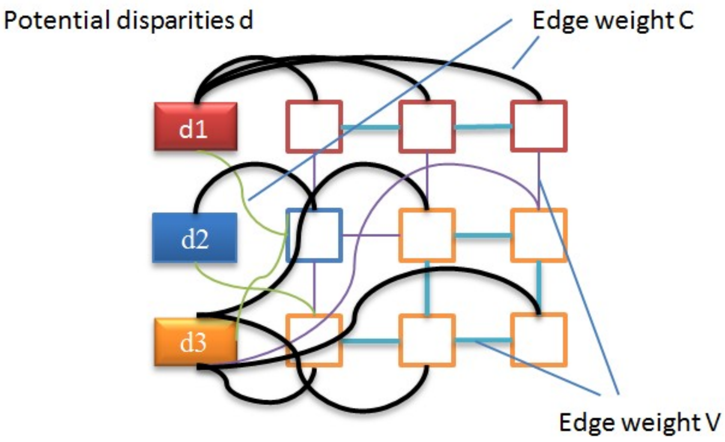


Figure 2.14: Stereo matching using Graph Cuts algorithm

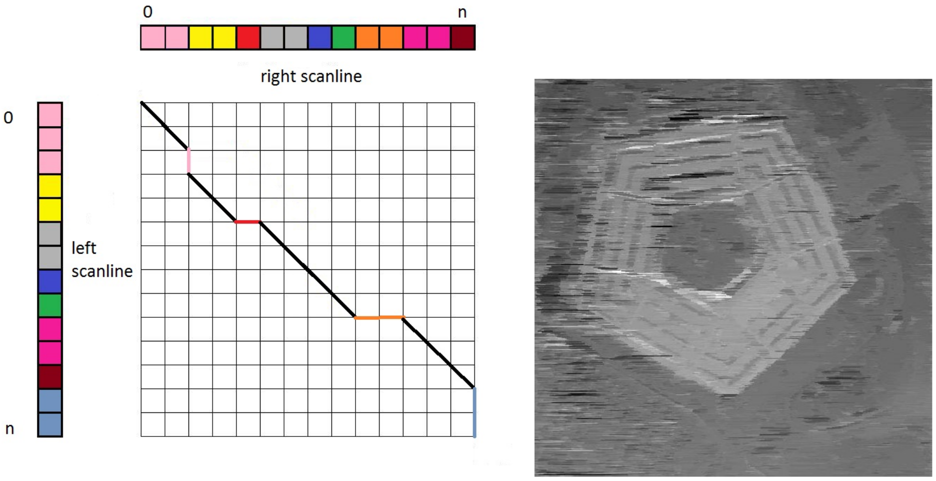
The solution to the global optimization problem is reached when the energy function is at a global minimum. Numerous optimization algorithms including Graph Cuts and belief propagation have been proposed to solve the function. Graph Cuts algorithms [108, 24, 76] consider stereo matching as a labeling problem. Graph Cuts assigns a pre-defined label to each pixel, as shown in Figure 2.14. In stereo matching, each pixel corresponds to a node in a graph and the labels are the potential disparity values. Then, the likelihood of the disparity are set to the edges between each label and each node.

Belief propagation [122] is a multi-class optimization algorithm. The number of nodes in belief propagation is $(image\ width) \times (image\ height) \times (disparity\ range)$. Each node (u, v, d) sends messages to the neighbor nodes and receives messages from all neighbor nodes at the same time. After receiving the messages, each pixel computes its belief. After several iterations, when the belief functions converge, the disparity of pixel (u, v) is the d value of the node (u, v, d) with the largest belief. Sun *et al.* [122] include an occlusion term to eliminate the negative effect caused by occlusions.

One can use dynamic programming method [40] to solve the multi-class optimization problem by strictly following the ordering constraint. It works on the comparison of two scan-lines $\mathbf{l_L}$ and $\mathbf{l_R}$. The parameter $|\mathbf{l}|$ is the number of pixels in one scan-line. Pixels on each scan-line are assigned different matching costs according to match or unmatch. The minimum cost of matching two scan lines $\mathbf{C}(|\mathbf{l_L}|, |\mathbf{l_R}|)$ yields an optimal path. One can see in Figure 2.15(a) the optimal path chosen for two scan-lines and in Figure 2.15(b) the corresponding disparity map.

Dynamic Programming based stereo matching methods have been proven to be very fast and can even reach global optimization in real time. However, since the local errors may be propagated along a scan-line as in Figure 2.15(b), streak marks along the scan lines degrade the results.

Recently, segment-based algorithms have attracted a lot of attention. These algorithms have the best ranking on the Middlebury stereo evaluation website [5]. Segment-based stereo matching



(a) Optimal path chosen

(b) A result of dynamic programming with streaks [40]

Figure 2.15: Matching two scan lines using dynamic programming

algorithms assume that the depth boundary is correlated to the color discontinuity and the scene can be approximated by piecewise planar surfaces. Most of these algorithms follow a four-step procedure: over-segmentation to the reference image, stereo matching to get an initial disparity map, plane-fitting for each segment, and global optimization for the final disparity map. Tao *et al.* [124] iteratively warp the segments from the reference view to the second view and update the depth by minimizing the global image similarity energy. Hong *et al.* [63] fit a plane for each single segment and groups the neighbor segments if they are on the same depth plane. At last, each segment is assigned to the disparity plane with the largest possibility. Wei *et al.* [137] start from computing the Ground Control Points (GCP), which are supposed to be pixels with correct matches. The images are divided into regions by color segmentation. Reliable regions are matched by GCPs on them and marked as MATCHED. The regions that contain more than one disparity are split into smaller regions. All the UNMATCHED regions are matched by a progressive framework instead of the time-consuming global optimization. Klaus *et al.* [75] assume all points on the same segment are on the same disparity plane. Then, segments are treated as nodes in the global optimization which is solved as a labeling problem. The labels are disparity planes found in plane fitting step. Wang *et al.* [Wang 08] calculate the coefficients of the disparity planes by voting strategy and use a inter-regional cooperative optimization algorithm to achieve global minimization. Yáng *et al.* [142] first initialize a disparity map by hierarchical belief propagation with a data term based on color-weighted correlation. They then classify pixels as stable, unstable, or occluded. The third step is to use a belief propagation framework with segmentation and plane-fitting constraints to iteratively refine disparities. Pixels labeled as stable are used to help find the disparities of unstable pixels. Xu

Table 2.2: Various types of stereo matching algorithms

Local		Global	
without weight	with weight	without segmentation	with segmentation
ASD	Kanade 94[72]	Marr 76[88]	Tao 01[124]
SSD	Yoon 06 [145]	Roy 98 [108]	Hong 04 [63]
NCC	Tombari 07 [125]	Cox 96[40]	Wei 04 [137]
ZNCC	Hosni 09 [64]	Boykov 01[24]	Klaus 06 [75]
Egnal 00 [44]	Veksler 03 [127]	Kolmogorov 01 [76]	Yáng 06 [142]
	Tombari 08[15]	Sun 02 [122]	Wang 08 [136]
			Xu 08 [141]

et al. [141] introduce Outlier Confidence values to each pixel after initializing the disparity map. The following global optimization use the outlier confidence maps to build the data term.

2.2.3 Comparison of Stereo Matching Algorithms

The stereo matching algorithms discussed previously are classified in Table 2.2. The main steps of stereo matching can be summarized as camera calibration, image rectification, and disparity computation. Since the entire test images from the Middlebury website [4] are rectified, the first two steps can be ignored. Generally, local methods are faster than the global ones, but have more artifacts on boundaries, low-textured areas, and occlusions. Global methods integrate the smoothness term in the energy function to solve the occlusion problem. And global methods perform better than local methods because of the global optimization [60]. Weighted local methods are effective at removing the pixels on different disparity planes from the calculation of the matching cost. However, they fail in the high texture areas because only a few pixels are thought to be on the same disparity plane with the central pixel. Among all global methods, segmentation-based methods have the best results according to the Middlebury evaluation [5].

From the speed perspective, global stereo matching methods are much slower than local methods. Furthermore, the computational structure of local stereo matching is parallel as each pixel's is independent to each other when looking for its best match. Such a parallel computational structure makes it easy to implement local stereo matching on Graphics Processing Unit (GPU) for real-time performance.

2.3 View Interpolation

Figure 2.16 is a classification of various Image Based Rendering (IBR) techniques found in the literature. View interpolation is a technique used to synthesize new views from a discrete set of cameras. It creates novel images from virtual viewpoints by either re-sampling the pixels or interpolating the parameterized rays. Unlike traditional rendering, view interpolation does not need to reconstruct a 3D model and the speed of the rendering process is independent of the scene complexity. A virtual image is interpolated from real images, and as a result it is more realistic than rendered images gen-

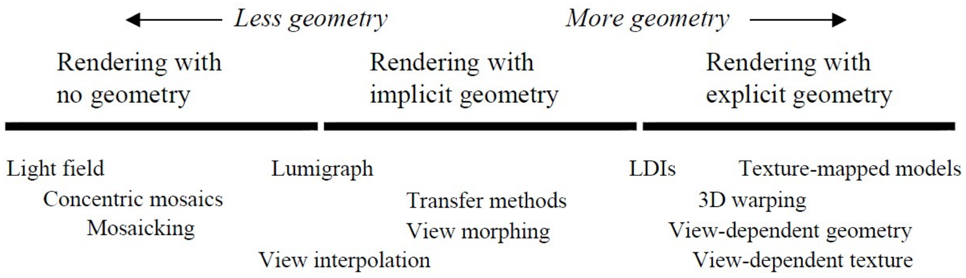


Figure 2.16: Classification of techniques in image based rendering according to the availability of geometric information [116]

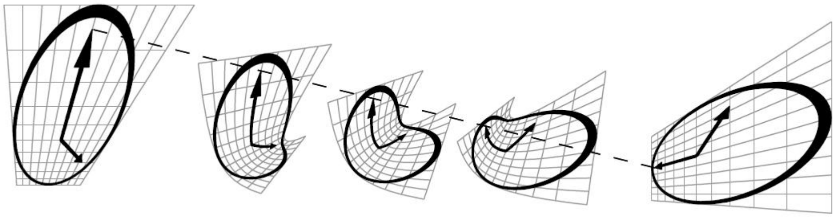


Figure 2.17: Shape distortion caused by image morphing [113]

erated by the traditional computer graphics pipeline. Several real images of a scene are captured by cameras and warped to the position of the virtual viewpoint guided by pixel correspondence. The color of the final virtual image is a weighted sum of the multiple warped images.

The View interpolation methods we discuss in this section use implicit geometry (pixel correspondence information) or no geometry (ray-based). Based on the density of the pixel correspondence, the category called “interpolation with pixel correspondence” can be further divided into two subcategories as being “with dense correspondence information” or “with sparse correspondence information”.

2.3.1 View Interpolation with Dense Correspondence Information

In 1993, Chen and Williams are the pioneers to synthesize virtual images from real images without 3D models [31]. They obtain the pixel correspondence from the range data and camera transform. Following this process, they use image morphing techniques to transform two neighbor images into an in-between one. However, simple image morphing does not work for perspective views as it causes obvious shape distortion, as shown in Figure 2.17.

Laveau and Faugeras [79] predict views at new viewpoints by exploring the relations among reference images instead of 3D scene model. Only two cameras are needed to be fully calibrated in case of more than two reference viewpoints. In 1996, Seitz and Dyer proposed “view morphing” [113] to correct the shape distortion appeared in Chen’s paper [31]. They think that the shape can be preserved during image morphing if the two views are parallel. Figure 2.18 shows a result of view morphing.

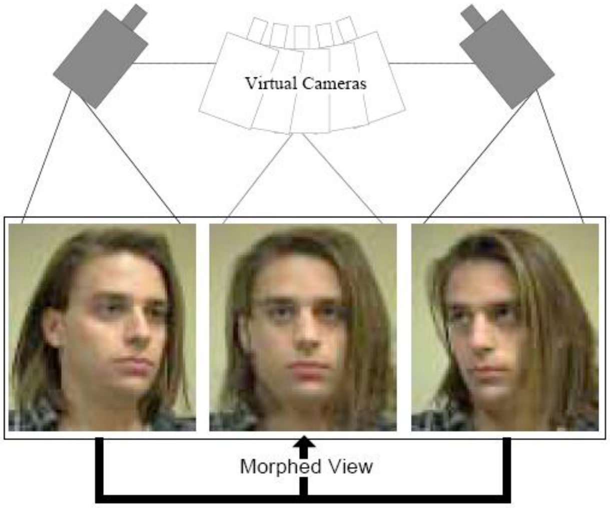


Figure 2.18: Illustration of view morphing [113]

Let's assume two parallel images are $\mathbf{I_0}$ and $\mathbf{I_1}$ and the centers of the camera are $\mathbf{L_0} = (0, 0, 0)$ and $\mathbf{L_1} = (L_x, L_y, 0)$. Let $\mathbf{M_P^0}$ and $\mathbf{M_P^1}$ be the perspective projection matrices of both cameras respectively, with focal length f_0 and f_1. A 3D point is projected onto the 2D image by:

$$\mathbf{p_0} = \mathbf{M_P^0 P}, \tag{2.8}$$

where $\mathbf{P} = (x, y, z)$ is a point in the 3D world coordinate and $\mathbf{p_0} = (u_0, v_0)$ is a pixel in $\mathbf{I_0}$ projected from $\mathbf{P}$. There is a similar projection in $\mathbf{I_1}$, $\mathbf{p_1} = \mathbf{M_P^1 P}$. Then the pixel $\mathbf{p_s}$ projected from $\mathbf{P}$ on image $\mathbf{I_s}$ is a linear interpolation of $\mathbf{p_0}$ and $\mathbf{P_1}$ [113]:

$$\mathbf{p_s} = (1 - s)\mathbf{p_0} + s\mathbf{p_1}, \tag{2.9}$$

where s is the normalized location between $\mathbf{I_0}$ and $\mathbf{I_1}$.

If two views are not parallel, then two more steps are necessary to avoid image distortion: pre-warping and post-warping [113]:

1) Pre-warping: $\mathbf{I_0}$ is pre-warped to $\widehat{\mathbf{I_0}}$ and $\mathbf{I_1}$ is pre-warped to $\widehat{\mathbf{I_1}}$. $\widehat{\mathbf{I_0}}$ and $\widehat{\mathbf{I_1}}$ are parallel views;

2) View morphing: interpolate $\widehat{\mathbf{I_0}}$ and $\widehat{\mathbf{I_1}}$ into $\widehat{\mathbf{I_s}}$ by image morphing;

3) Post-warping: $\widehat{\mathbf{I_s}}$ is post-warped into $\mathbf{I_s}$.

However, Seitz's method does not work very well if the optical axis is parallel to the transition baseline or just close. Huang *et al.* [67] proposed a novel pre-warping technique to correct this problem. Unlike the pre-warping in View Morphing, Huang's pre-warping procedure warps two optical axes onto the same line with the baseline as illustrated in Figure 2.19. After pre-warping, the next step in this approach is to apply image morphing to the warped images based on the computed correspondences. The result of image morphing is then post-warped to the desired camera

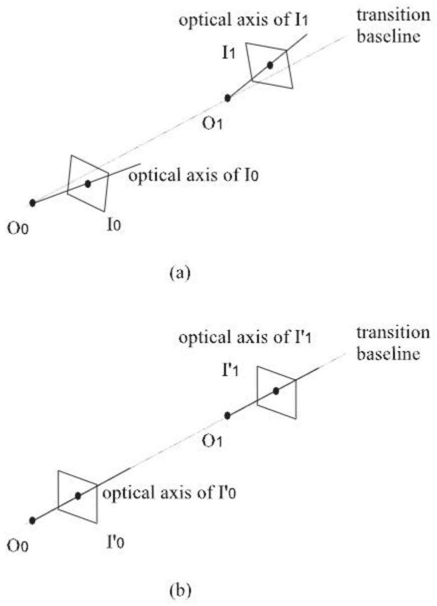


Figure 2.19: Pre-warping in Huang's method [67]

orientation.

According to McVeigh [91], the relation of un-occluded pixels in the stereoscopic rectified images is:

$$\mathbf{p}_r = \begin{bmatrix} u_r \\ v_r \end{bmatrix} = \begin{bmatrix} u_l + d_{lr}(u_l, v_l) \\ v_l \end{bmatrix} = \mathbf{p}_l + \begin{bmatrix} d_{lr}(u_l, v_l) \\ 0 \end{bmatrix}. \tag{2.10}$$

For occluded areas, disparities can be inferred from un-occluded regions because the depth in the occluded regions is supposed to be constant. After completing the disparity map, the virtual view is mapped from left or right images and merged by a linear interpolation:

$$\mathbf{p}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} u_l + \nu d_{lr}(u_l, v_l) \\ v_l \end{bmatrix}, \tag{2.11}$$

where $\nu \in [0, 1]$ is the relative location between the left and right image.

Scharstein *et al.* [110] proposed a method to interpolate views from an incomplete disparity map. First, they look for the correspondence by stereo matching from rectified images. Then, they map left and right images to the new viewpoint and combine them into the new image. Occlusion areas are filled by texture synthesis. At last, the generated view image is de-rectified by an inverse transform.

Lhuillier and Quan [84] construct a joint view triangulation based on a quasi-dense disparity map. In this way, there is a one-to-one correspondence between the triangular vertices and a one-to-one correspondence between the boundaries of matched regions. This interpolation is more complicated than simple linear interpolation. It contains three steps: first, linearly interpolate of vertices of each triangle. Then, map all triangles in $\mathbf{I_0}$ and $\mathbf{I_1}$ to $\tilde{\mathbf{I}}_\mathbf{0}$ and $\tilde{\mathbf{I}}_\mathbf{1}$. Finally, blend the pixel values from $\tilde{\mathbf{I}}_\mathbf{0}$ and $\tilde{\mathbf{I}}_\mathbf{1}$ by a weighting function.

Song *et al.* [118] proposed a bi-directional interpolation on a dense correspondence. In order to illuminate the holes in the in-between image, especially when the corresponding pixels are both invisible in the reference images, the pixel values are calculated by the following rule:

$$\mathbf{I_s}(\mathbf{P}) = \begin{cases} (1-s)\mathbf{I_0}(\mathbf{P}) + s\mathbf{I_1}(\mathbf{P}), & \text{if } \mathbf{I_0}(\mathbf{P}) \neq 0 \text{ and } \mathbf{I_1}(\mathbf{P}) \neq 0 \\ \mathbf{I_0}(\mathbf{P}), & \text{if } \mathbf{I_0}(\mathbf{P}) \neq 0 \text{ and } \mathbf{I_1}(\mathbf{P}) = 0 \\ \mathbf{I_1}(\mathbf{P}), & \text{if } \mathbf{I_0}(\mathbf{P}) = 0 \text{ and } \mathbf{I_1}(\mathbf{P}) \neq 0. \\ \text{nearest neighbor pixel}, & \text{if } \mathbf{I_0}(\mathbf{P}) = 0 \text{ and } \mathbf{I_1}(\mathbf{P}) = 0 \end{cases} \tag{2.12}$$

where $\mathbf{I_0}(\mathbf{P})$ is the pixel value of the pixel projected from 3D point $\mathbf{P}$ onto $\mathbf{I_0}$. In addition, a Z- buffer technique is used to handle ambiguity when multiple points project to the same pixel position.

Besides stereo cameras, dense disparity maps can be obtained from multiple cameras. Kanade *et al.* [71] set a virtualized reality studio with cameras placed on a dome. Then, the 3D structure is extracted by multi-baseline stereo and known camera geometry. The new views are rendered by both depth map and intensity map. Avidan *et al.* [19] utilize the tri-linear tensor to represent the correspondence among three viewpoints. A seed tensor is first constructed from three reference images. When there are two reference images available, the third image coincides with one of the two. The tensor of the new image and another two reference images are modified by applying the user- specified position of virtual viewpoint. This new tensor and dense correspondence between two selected reference images are used to render the new image. Saito *et al.* [66] explore the projection geometry among multiple cameras using the fundamental matrices. The new viewpoint should be inside the triangle whose vertices are on the three viewpoints. The new image is interpolated as:

$$\mathbf{p} = w_1\mathbf{p_1} + w_2\mathbf{p_2} + w_3\mathbf{p_3}. \tag{2.13}$$

$\mathbf{p_1}$, $\mathbf{p_2}$, and $\mathbf{p_3}$ are the coordinates of corresponding pixels on reference images. w_1, w_2, and w_3 are weighting factors based on the distance between $\mathbf{P}$ and a viewpoint, and satisfy $w_1 + w_2 + w_3 = 1$. Xiao and Shah [140] first re-build a tri-focal plane among three reference cameras. Then, the disparity map is computed between each rectified image pair. The new pixels $\mathbf{p_s}$ in the virtual image are linearly blended by three corresponding pixels $\mathbf{p_0}$, $\mathbf{p_1}$, and $\mathbf{p_2}$:

$$\mathbf{M_P^s} = \eta_1\mathbf{M_P^1} + \eta_2\mathbf{M_P^2} + \eta_3\mathbf{M_P^3}, \tag{2.14}$$

$$\mathbf{p_s} = \frac{1}{z}\mathbf{M_P^s}\mathbf{P_s}. \tag{2.15}$$

$\mathbf{M_P}$ is the projection matrix, f is the focal length and $\mathbf{L}$ is the camera optical center.

$$f_s = \eta_1 f_1 + \eta_2 f_2 + \eta_3 f_3, \tag{2.16}$$

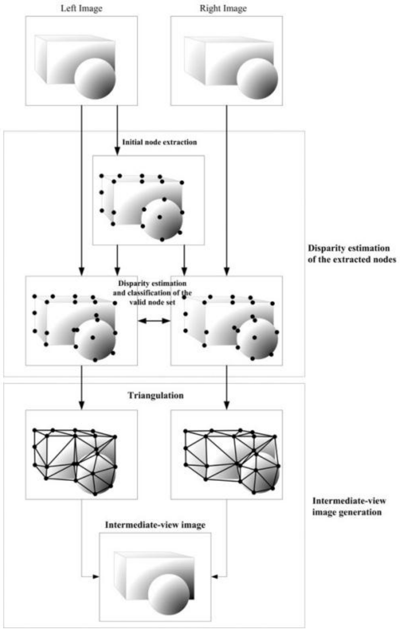


Figure 2.20: The procedure of Park's algorithm [99]

$$\mathbf{L}_s = \eta_1 \mathbf{L}_1 + \eta_2 \mathbf{L}_2 + \eta_3 \mathbf{L}_3, \tag{2.17}$$

The parameters η_1, η_2, and η_3 are linear combination coefficients which must satisfy $\eta_1 + \eta_2 + \eta_3 = 1$.

Rana *et al.* [104] suggest warping multiple depth images (more than two) to the virtual viewpoint in order to examine the spatial consistency. The pixels with valid connection hypotheses in the virtual image are filled by the average of the connected pixel colors.

2.3.2 View Interpolation with Sparse Correspondence Information

Pollard's paper [101] only matches edges in three reference images. Then, the disparities on the edges are projected onto the virtual image and the rest of the virtual image is rendered by the raster-based technique.

Park *et al.* [99] look for the disparities of the feature pixels in the left image. The left image is divided into multiple m-by-m blocks and then the pixels with the highest gradient in each block are extracted as feature pixels. A mesh of the whole scene is built on those feature pixels using Delaunay Triangulation [54]. The two reference images are warped to the novel image, patch by patch, according to the disparities on the vertices. Figure 2.20 shows the diagram of Park's algorithm.

Choi *et al.* [35] use a mesh-based method as an intermediate view interpolation from a rectangular multi-view camera system. The feature points are detected by edge detection and the triangular meshes are composed by Delaunay triangulation [54]. Meshes are then divided as foreground and

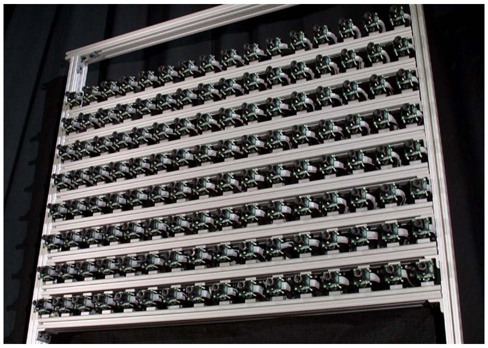


Figure 2.21: Camera array for a Light Field system [83]

background meshes according to disparity values of feature points on them. The intermediate view is remapped by applying an affine transform on the reference views.

2.3.3 Ray-based View Interpolation

Basically, this kind of view interpolation, which includes Plenoptic modeling, Light Field Rendering, and Lumigraphs, is based on ray space. McMillan and Bishop [90] define a 5D plenoptic function to represent a scene by simplifying the original 7D plenoptic function $\mathbf{P}(\theta, \phi, \gamma, V_x, V_y, V_z, t)$ [14] without wavelength γ and time t terms. For each viewpoint, they create a cylindrical projection from all reference images taken from the same viewpoint [90]. The new viewpoint is projected from other viewpoints by cylindrical-to-cylindrical mapping. Finally, the panorama is warped back to a planar image. It is necessary to mention that in some literatures, Plenoptic modeling [90] is classified into the category of "View Interpolation with Dense Correspondence Information" because the creation of images from novel viewpoint is also dependent on disparity information.

The main idea of Light Field Rendering [83] is the construction of a light field from a dense camera array. A new virtual image is interpolated by the parameterized lights without any geometry or correspondence information. Figure 2.21 shows the equipment required to create a light field. Two parallel planes are defined as uv-plane and st-plane. Each light ray is parameterized as a 4D function $\mathbf{L}(u, v, s, t)$. Each pixel in the uv-plane is a camera. As shown in Figure 2.22, the st-plane is the image viewed by the camera (u, v), while each pixel in the st-plane is a collection of all the lights of image (s, t) from all the cameras in the uv-plane. The light rays in a new image are then interpolated from adjacent lights.

Similar to Light Field Rendering, Gortler *et al.* [50] sample a 4D function called Lumigraph. As we all know, the plenoptic function [90] is 5D consisting of the position and direction of each light: $\mathbf{P}(x, y, z, \theta, \phi)$. They believe that a bounding box of an object can hold all the light information; thus, the surfaces of the bounding box are enough to define a ray. In the Lumigraph algorithm, the rays from the new viewpoint are interpolated from nearby rays too. One advantage that the Lumigraph has over Light Field is that there is no need to construct a camera array. A hand-held

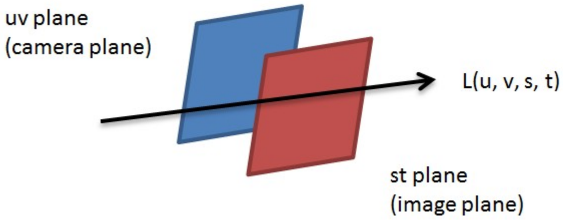


Figure 2.22: st-plane and uv-plane

Table 2.3: The category of view interpolation algorithms

ray-based	with pixel correspondence	
	dense	sparse
McMillan 95 [90] Gortler 96 [50] Levoy 96 [83]	Chen 93 [31] Laveau 94 [79] Seitz 96 [113] Huang 98 [67] McVeigh 96 [91] Scharstein 96 [110] Lhuillier 97 [84] Song 04 [118] Kanade 95 [71] Avidan 98 [19] Saito 02 [66] Xiao 03 [140] Rana 10 [104]	Pollard 98 [101] Park 03 [99] Choi 10 [35]

camera is enough to take numerous images along the camera plane.

2.3.4 The Comparision of View Interpolation Algorithms

Table 2.3 shows the category of view interpolation algorithms we introduced before. Ray-based interpolation is the fastest algorithm in view interpolation since the interpolation is only between neighboring parameterized light rays. It is the best choice for the real-time performance, but ray-based view interpolation also requires a dense camera array and a large storage memory, which are not easily accessible.

Sparse-correspondence-based methods only compute the disparities of feature pixels, and then construct meshes by connecting those feature pixels. Interpolation is performed within each triangle or polygon. It is faster than the dense-correspondence-based method, but the artifacts may happen when the vertices are on different disparity levels.

Compared to sparse-correspondence-based methods, dense-correspondence-based methods deal with interpolations pixel by pixel. Therefore, the disparity of each pixel has to be computed, which is time-consuming. Moreover, because of the occlusion and low-texture problem, some of the pixels cannot find their correct corresponding pixels by stereo matching. Then, wrong disparity values have to be found out, for example, using cross-checking [46], and a post-processing, such as filtering or interpolation, is necessary to fill up those missing disparity values.

Chapter 3

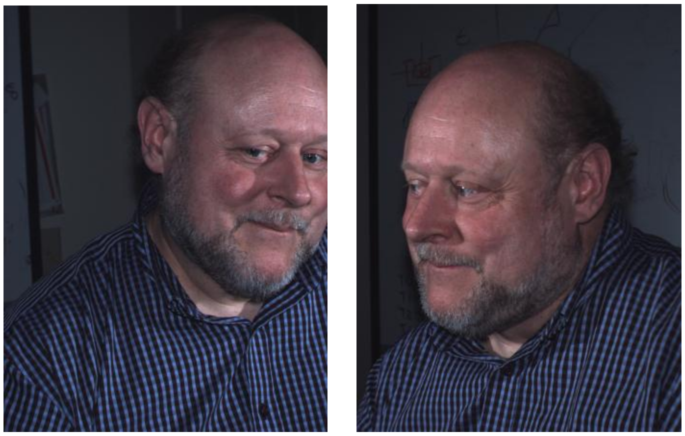
A New Eye Contact Correction Algorithm for Large Baseline Tele-presence System

In this chapter, we describe how our solution to eye contact correction problem works. Specifically, how to correct eye contact with cameras located on the opposite sides of a large tele-presence screen which creates a large baseline problem.

3.1 Introduction

As discussed in Chapter 1, correct eye contact, is essential for emotional communication because eye contact carries non-verbal clues. In commercial products, eye contact is approximated by using either a large semi-transparent beam splitter mirror to project an image of the remote participant and capture the local participant image by using a video camera aiming the center of the screen or by using low profile cameras that on the top of the display screen. These systems are very cumbersome and require complex hardware and software setups. A low-cost solution is to digitize the participants by locating a group of cameras around the display screen and calculating a disparity map from those cameras using stereo matching algorithms. Once the disparity map is determined, virtual images are generated from the viewpoint of the participant's gaze direction.

In general, in order to create a good disparity map, the stereo cameras should be located very close to each other to avoid occlusions. At minimum, one camera is set on each side of the participant in order to interpolate virtual views in-between. However, as the size of the display screens becomes larger and larger to display participant at life-size, the stereo image shares little corresponding information, as shown by the example in Figure 3.1. Two cameras are used: one on the left side and one on the right side of the screen. The left-side camera can only capture the left side of the face and a small part of right side. The right-side camera can capture the whole right side of the face and a small part of left side. Only a few pairs of corresponding pixels can be matched if we apply directly a stereo matching algorithm to these images. The disparity map is too sparse to generate



(a) Left image

(b) Right image

Figure 3.1: Image taken by cameras with wide stereo baseline

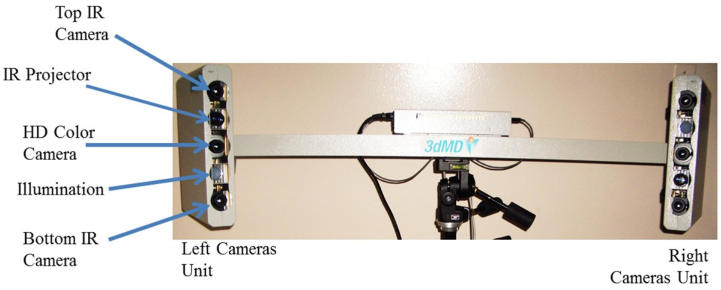


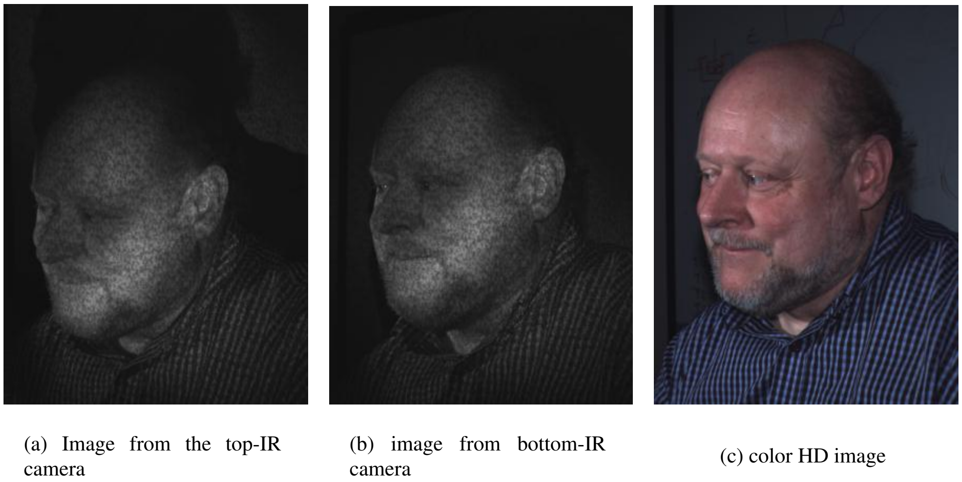
Figure 3.2: Modified 3dMD camera system

new views from any virtual viewpoint.

Therefore, in this chapter, we propose a solution to solve the eye contact correction problem with large baseline. For each pixel in the left/right image, we try to find its corresponding position in the right/left image as if it shows up in the right/left image. Then, using a depth-based interpolation, any virtual views in-between can be rendered. The rest of this chapter describes the system layout and the proposed eye contact correction method.

3.2 System Layout

Figure 3.2 is an illustration of the hardware design installed on the user's terminal. Two sets of modified 3dMD [6] camera units are installed on the opposite sides (left and right) of a large telepresence monitor. The 3dMD system is composed of two camera units. Each unit has two high resolution black and white cameras with IR filters, one HD color camera, one IR projector, and one white light frontal illumination unit. The original system is modified to deal with our application in order to be able to install a large screen between the two units. The two black and white cameras are



(a) Image from the top-IR camera

(b) image from bottom-IR camera

(c) color HD image

Figure 3.3: Tele-presence images digitized by the 3dMD cameras

also modified to deal with infrared illumination by placing a band-limited IR filter in front of their CCDs and the pattern projection unit light is also replaced with IR diodes. In this system, the left and right units independently calculate the 3D world coordinates and their colors.

An invisible IR pattern with random dots is projected to the scene to help derive the disparity map in low texture areas, such as skin and clothes. Because the IR pattern is only visible to the IR camera, one color camera is added to record the scene colors. That is the reason we need two kinds of cameras. IR cameras on each side are responsible for creating the disparity map of their own side using stereo matching.

3.3 Eye Contact Correction Solution

Figure 3.3 is an example of images taken by the three cameras using the right unit. The random dots add invisible texture to low texture areas. We find that the head, neck and upper torso have no sharp disparity changes; thus, the whole foreground in Figure 3.3(a) or 3.3(b) can be recovered by one continuous surface, which is easily obtained by foreground extraction.

Figure 3.4 shows the flow chart of the proposed system. The green boxes represent inputs and outputs. The blue boxes represent intermediate results. The orange boxes are operations performed on original images and intermediate results.

3.3.1 Stereo Matching

According to the evaluation conducted by the Middlebury test bed [5], the best stereo matching results are from plane-fitting based algorithms [124, 63, 137, 75, 136, 142]. These algorithms assume that the scenes can be modeled by a set of planes, and that large disparity discontinuities only happen at the boundaries of homogenous color segments. Hence, all the potential disparity planes should

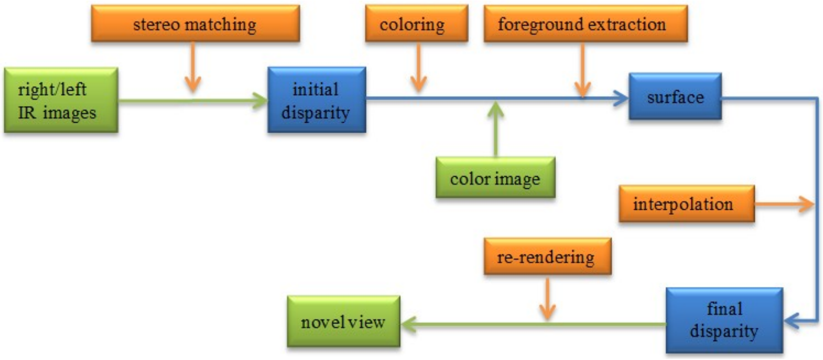


Figure 3.4: Flow chart of the proposed method for a tele-presence system with stereo cameras on one side

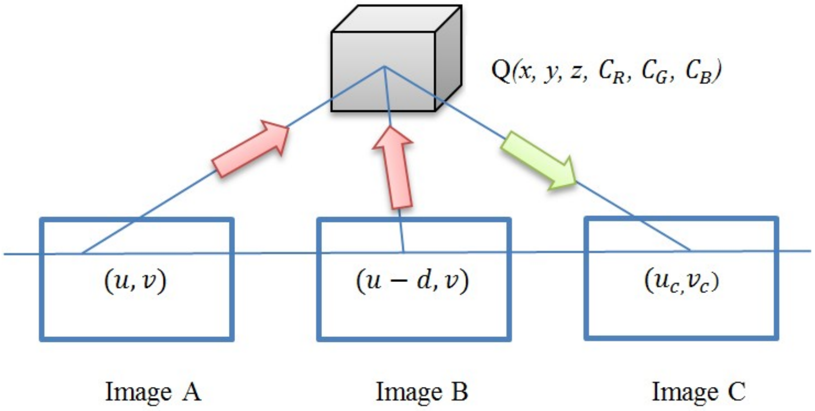


Figure 3.5: The relationship of coordinates among three cameras

be detected from the initial disparity map, but it is very time-consuming to fit thousands of planes caused by over-segmentation and because of our real-time constraint, we decided to use a basic local stereo matching algorithm. The algorithm uses the Sum of the Absolute Differences (SAD) as its matching cost function and cross-checking method [46] is used to detect those incorrect disparity values. Cross-checking is expressed by Equation 3.1, which is to check whether the disparity values of the corresponding pixels inferred by the diaprity maps (D_L and D_R) are the same.

$$\begin{cases} D_L(i,j) = D_R(i - D_L(i,j), j) \Rightarrow \ D_L(i,j) \ is \ correct; \\ D_L(i,j) \neq D_R(i - D_L(i,j), j) \Rightarrow \ D_L(i,j) \ is \ wrong. \end{cases} \tag{3.1}$$

In addition, because tele-presence mainly deals with human faces which are renowned to have very little textures, the proposed projection of IR pattern solves this problem effectively.

3.3.2 Pixel Coloring

Pixel coloring consists of aligning the disparity map with the color image. Figure3.5 shows the coordinates on three images of a 3D point $Q(x, y, z)$ with color (C_R, C_G, C_B). Let's take the case of the left-side cameras for example:

Each pixel (u, v) and its corresponding pixels $(u - d, v)$ are projected from the point **Q** in a 3D world with coordinates (x, y, z) that can be computed by the triangulation rule and projection matrix. In our system, since the color images are acquired from a color camera rather than the IR cameras, coordinates on the color image where the color of (x, y, z) is stored can be calculated by:

$$\begin{bmatrix} u_c \\ v_c \end{bmatrix} = \mathbf{M^c_P} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \mathbf{M^c_{intrinsic}} \begin{bmatrix} \mathbf{M^c_{rotation}} | M^{\mathbf{c}}_{\mathbf{translation}} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{3.2}$$

where $\mathbf{M^c_{intrinsic}}$, $\mathbf{M^c_{rotation}}$ and $\mathbf{M^c_{translation}}$ are intrinsic matrix, the rotation matrix and translation matrix of the color camera, respectively, which are determined by a camera calibration procedure performed previously.

3.3.3 Foreground Extraction

Since the background in our system will be replaced by a virtual scene, we care only about the disparity of the foreground. Since there is an obvious difference of the disparity values between the foreground and background, the foreground is easy to be separated apart from the background with a threshold and the largest connected component is the foreground. The initial foreground boundary is then refined by eroding and dilating to get rid of noises and holes inside the foreground are filled. The initial boundary can also be refined by a fast relaxation labeling algorithm [138] or by a matting method that will be described in the next chapter.

3.3.4 Foreground Interpolation

After the cross-checking, some pixels have been marked as bad pixels (holes in black), which need interpolation in order to be filled. Technically, any interpolation algorithm could be applied. We chose (Radial Basis Function) RBF interpolation and we will extend the RBF interpolation to a general stereo matching algorithm later on.

RBF interpolation [103, 26] is one of the global spatial interpolation techniques which have been shown to work well for scattered point clouds. Hence, it has been widely used in 3D modeling [28, 77] and face modeling [93]. RBF interpolation fits a radial basis function to each individual point so that there is no constraint on the complexity of the surface it can represent. The interpolations function is described as [28]:

$$f(\mathbf{p}) = K(\mathbf{p}) + \sum_{i=1}^{n} \eta_i \phi(\|\mathbf{p} - \mathbf{p}_i\|). \tag{3.3}$$

The variable $\mathbf{p}_i$ represents the image coordinates (u_i, v_i) and $f(\mathbf{p})$ is the disparity value of pixel $\mathbf{p}$. Let's define $\{\mathbf{p}_i\}, (1 \le i \le n)$ to be a sample set of pixels with correct disparity values. The distance $\|\mathbf{p} - \mathbf{p}_i\|$ is the Euclidean distance between $\mathbf{p}$ and $\mathbf{p}_i$. Let's $K(\mathbf{p})$ be a linear polynomial term in the form of $au + bv + c$, where a, b, and c are coefficients. The function $\phi(r)$ is a radial basis function. Any symmetric function with a single and positive variable (the distance from the origin), such as the inverse multiquadric, can be used as the radial basis function. The mathematic expression of the inverse multiquadric [10] is:

$$\phi(r) = (r^2 + c^2)^{-1/2}, \tag{3.4}$$

where c^2 is a smoothness term. Normally, n samples can only build n functions, but there are $n + 3$ parameters in Equation 3.3. The problem cannot be solved without the following constraints [28]:

$$\sum_{i=1}^{n} \eta_i = \sum_{i=1}^{n} \eta_i u_i = \sum_{i=1}^{n} \eta_i v_i = 0. \tag{3.5}$$

Then, Equations 3.3 and 3.5 can be combined as **AX** $=$ **B** and solved linearly where the matrices **A**, **X** and **B** are defined as:

$$\begin{bmatrix} \phi_1 & \phi_2 & ... & \phi_n & u_1 & v_1 & 1 \\ \phi_1 & \phi_2 & ... & \phi_n & u_1 & v_1 & 1 \\ & & ... & & & & \\ & & ... & & & & \\ \phi_1 & \phi_2 & ... & \phi_n & u_1 & v_1 & 1 \\ u_1 & u_2 & ... & u_n & 0 & 0 & 0 \\ v_1 & v_2 & ... & v_n & 0 & 0 & 0 \\ 1 & 1 & ... & 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \eta_1 \\ \eta_2 \\ \cdot \\ \cdot \\ \eta_n \\ a \\ b \\ c \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \cdot \\ \cdot \\ f_n \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{3.6}$$

Based on all the pixels with correct disparity, the $n + 3$ parameters in Equation 3.3 can be determined by solving Equation 3.6. We then apply Equation 3.3 to calculate all missing disparities. The disparity value d of a new point is calculated by giving its coordinates (u, v). Following this step, each pixel in the foreground has a valid disparity.

3.3.5 Re-rendering

After the interpolation, each pixel in the foreground can find its corresponding 3D point in the global coordinate system by the triangulation rule. As we mentioned before, there are two ways to produce the final virtual image: re-projection or interpolation.

Re-projection: With a known re-projection matrix, the 3D points detected by the left camera unit are projected to the new virtual viewpoints one by one where a new image called $\mathbf{I'_L}$ is created. The same is performed for the right camera unit creating image $\mathbf{I'_R}$ and if there were more units the same process would apply. A pixel marked as null in the new images $\mathbf{I'_L}$ and $\mathbf{I'_R}$ means that there are no 3D point that can be projected on this pixel. The final virtual image $\mathbf{I_V}$ is a blending of the two re-projected images $\mathbf{I'_L}$ and $\mathbf{I'_R}$ (similar to [118]) as defined by:

$$\mathbf{I_V}(u,v) = \begin{cases} \mathbf{I'_L}(u,v), & \text{if } \mathbf{I'_L}(u,v) \neq \ null \text{ and } \mathbf{I'_R}(u,v) = \ null \\ \mathbf{I'_R}(u,v), & \text{if } \mathbf{I'_L}(u,v) = \ null \text{ and } \mathbf{I'_R}(u,v) \neq \ null \\ \frac{\mathbf{I'_L}(u,v)+\mathbf{I'_R}(u,v)}{2}, & \text{if } \mathbf{I'_L}(u,v) \neq \ null \text{ and } \mathbf{I'_R}(u,v) \neq \ null \\ 0, & \text{if } \mathbf{I'_L}(u,v) = \ null \text{ and } \mathbf{I'_R}(u,v) = \ null \end{cases} \tag{3.7}$$

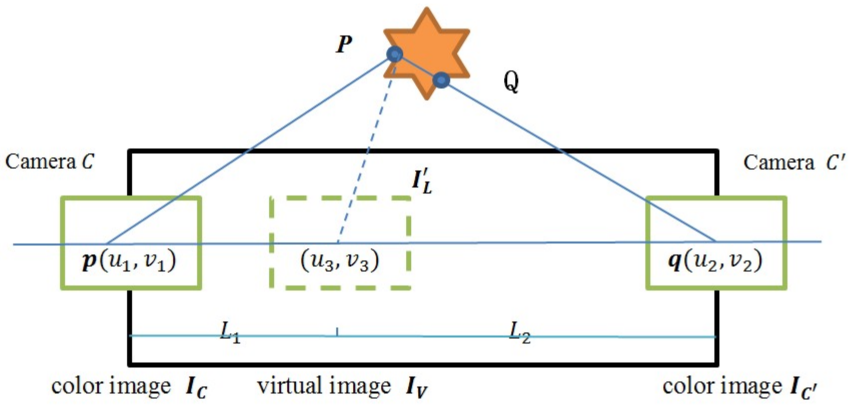


Figure 3.6: Interpolation from the left side

Interpolation: In this case, the re-projection matrix is unknown. Given the color images $\mathbf{I_C}$ and $\mathbf{I_{C^{'}}}$ from the color cameras on both sides and its corresponding disparity maps from left to right $\mathbf{D_C}$ and right to left $\mathbf{D_{C^{'}}}$, virtual images can be generated by interpolation for an image $\mathbf{I_L^{'}}$ on the line between the left to right color cameras and another one $\mathbf{I_R^{'}}$ from right to left.

According to the standard view morphing algorithm [113], the virtual image $\mathbf{I_L^{'}}$ is a linear interpolation of images $\mathbf{I_C}$ and $\mathbf{I_{C^{'}}}$ where the disparities of the corresponding pixels are known and the images $\mathbf{I_C}$ and $\mathbf{I_{C^{'}}}$ are both rectified. Unfortunately, because of the large baseline, most scenes captured by color camera $Camera_C$, such as the point **P** in Figure 3.6, are not necessarily visible in color camera $Camera_{C'}$. The location of point **P** on image $\mathbf{I_C}$ is (u_1, v_1), but because of occlusion one cannot see **P** on $\mathbf{I_{C^{'}}}$ since **P** is hidden by **Q**. The coordinates of **P** on $\mathbf{I_{C^{'}}}$ do exist (the same with **Q**: (u_2, v_2)) and can be calculated using Equation 3.2 with the projection matrix $\mathbf{M_P^{C^{'}}}$. Using the triangulation rule, the 3D world coordinates of **P** can be obtained from the disparity map of two IR cameras on the left side. In theory, each pixel on image $\mathbf{I_C}$ can find the location of its matching pixel on image $\mathbf{I_{C^{'}}}$. Afterwards, we have $v_1 = v_2$ and $d = |u_1 - u_2|$ which is the value of the disparity map from left to right $\mathbf{D_C}(\mathbf{p})$. The same holds true from right left, where $\mathbf{D_C^{'}}$ can be created.

In the proposed eye contact correction solution, the four images ($\mathbf{I_C}$, $\mathbf{I_{C^{'}}}$, $\mathbf{D_C}$, and $\mathbf{D_{C^{'}}}$) are sent to all other participants. Since we do not have the transmission part of the system implemented, we simulate the actions of the receiver and the sender at the same terminal. After the receiver receives these four images, the position of **P** on the interpolated image $\mathbf{I_L^{'}}(u_3, v_3)$ from the left side can be derived by:

$$\begin{cases} u_3 = u_1 - \frac{L_1}{L_1+L_2}d = u_1 - \frac{N}{L}d \\ v_3 = v_1 = v_2. \end{cases} \tag{3.8}$$

The color of the pixel (u_3, v_3) on image $\mathbf{I_L^{'}}$ is given by the color of the pixel (u_1, v_1) on image $\mathbf{I_C}$. In this way, we can obtain the interpolated images $\mathbf{I_L^{'}}$ and $\mathbf{I_R^{'}}$ from both sides, and blend them together using Equation 3.7. In the experiments, we divide the distance L into 10 segments so that L is 10 and N is an integer from 0 to 9. During the tele-conference, the end users could use a mouse to

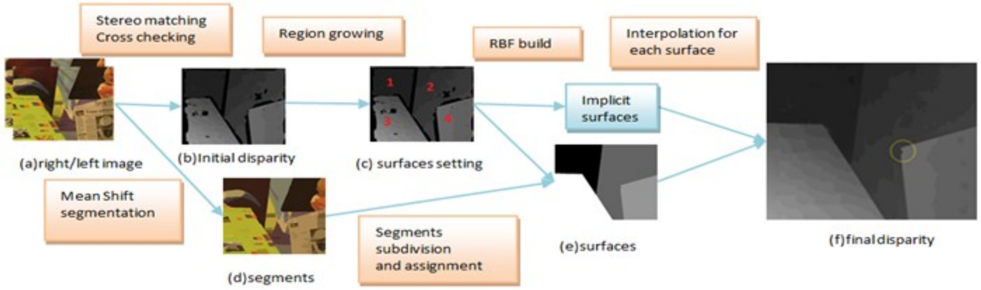


Figure 3.7: Flow chart of the proposed method for general stereo matching

select a remote participant by clicking and then use the keyboard (left and right arrows) to adjust the viewpoints (N value) between the most left ($N = 0$) and the most right ($N = 9$). The parameter L is free to be set as other integers and N is from $[0, L - 1]$.

3.4 Extension to Regular Stereo Imaging Problems

The interpolation of disparity maps in Section 3.3.4 is inspired by problems associated with tele-presence, but it can be extended for general stereo matching applications. Araujo [18] and Carr [29] did similar work to fill up the gaps on the depth map using RBF interpolation, but simply trained one group of RBF parameters for all the pixels in the image. We found in our experiments that RBF interpolation is not good at filling the gaps across several discontinuous depth planes. It is good for a tele-presence application because all the foreground pixels are on a continuous disparity plane. For general images, artifacts appear near the border areas with sharp changes. for extending or algorithm for general stereo matching is how to define the minimum number of non-overlapped surfaces (where each surface we assume that the disparities are continuous) that can cover the whole scene and how to assign each pixel to a certain surface. Each surface is going to train its own RBF parameters. Suppose the initial disparity map is ready, four more steps are designed to solve these problems: mean-shift segmentation, surface setting, segments-assisted pixel assignment, and RBF interpolation (Figure 3.7).

1) Mean-shift segmentation: The reference image is segmented using a mean-shift segmentation algorithm [38]. There is no need to over-segment the color image (just like the plane-fitting based algorithms do), because the segment will be subdivided into smaller pieces if there are more than one disparity planes in this segment.

2) Surface setting: Surfaces can be extracted quickly from the initial disparity map (from Section 3.3.1). Since interpolation is biased toward creating a smooth transition of data, sharp disparity discontinuities are not expected on a surface. Thus, region growing [13] is used to do this task. The region growing performs in this way. First, all the pixels with correct disparity values are marked as “$unvisited$” and black pixels (holes) are marked as “visited” . Then, start check each pixel from the beginning. Once we find an “$unvisited$” pixel i, we mark it as “$visited$” and check its neighbor pixel one by one in the 3-by-3 block. If the absolute difference of i and a neighbor pixel j is less than a pre-set threshold and j is "unvisited", the pixel j is marked as "*visited*". The algorithm then continues to check pixel j's neighbor pixels. The growing process stops when the absolute differences of a pixel with all its neighbor pixels are greater than the threshold or no more "unvisited" pixels in the neighborhood and one surface is found. The algorithm then finds the next pixel marked as "*unvisited*" and repeats the growing to look for the next surface, until all the pixels with correct disparity values are marked as "*visited*". Algorithm 3.1 is a recursive implementation of the region growing we described above. The disparity values in a final surface may cover a wide range, but will change smoothly.

Algorithm 3.1 Pseudo code of the region growing algorithm

```
FOR each pixel in the image: $visit[p] = $ "$unvisited$"
FOR each black pixel in the image: $visit[p] = $ "$visited$"
REGIONGROW(i)
if $visit[i] ==$ "$unvisited$" then
    $visit[i] =$ "$visited$"
    if pixel $j \in$ 3-by-3 neighbor of $i$ then
        if ($visit[j] ==$ "$unvisited$")and $(abs(disparity[i] - disparity[j]) < threshold)$ then
            REGIONGROW(j)
        end if
    end if
end if
```

Since occlusions usually happen on large disparity discontinuities, the black color in occlusion areas creates an artificial large discontinuity in disparity value to stop growing. This is the second line of defense separating two surfaces on different disparity planes.

3) Segments-assisted pixel assignment: In this step, surfaces extracted from "surface setting" are spread to include those pixels with unknown disparities. Overlapping the segmented color image and surfaces detected, one can see there are four kinds of segments: (1) a segment is totally on one surface; (2) a segment partially belongs to a surface, but the rest part are in black areas; (3) a segment spans multiple surfaces; (4) a segment does not overlap with any surface. The segment is assigned to the surface which the segment in on in the first two cases. For the third case, the segment is then further sub-divided into several segments: segments on different surfaces are assigned to those surfaces respectively while segments in the black areas (floating segments) are assigned to the surrounding surface with the lowest average disparity. The segments in case (4) are treated as floating segments. Floating segments are usually small areas and most of the errors are caused by the wrong assignment of the floating segments. All the pixels in the image can be assigned to a unique surface after this step.

4) RBF interpolation: The last step is the "RBF build and interpolation" step described in Section 3.3.4. This step is applied to each surface independently to infer the missing disparities.

3.5 Experimental Results

We test our method on both tele-presence images and standard stereo images. SAD is used as the matching cost function for the initial disparity map. The window size is 25×25 pixels and the threshold of the improved region grow is set to 10 for all the experiments.

3.5.1 Tests for Tele-presence Applications

The first tele-presence images for test are shown in Figure 3.3. Figure 3.8(a) shows the initial disparity derived from Figure 3.3(a) and Figure 3.3(b). Outliers are marked as black. The precision of the disparity map is greatly enhanced by those random dots except for some occlusion areas. In some cases, if there are not too many low-textured areas, even the initial disparity map is good enough to be used. In this case, the interpolation step can be skipped.

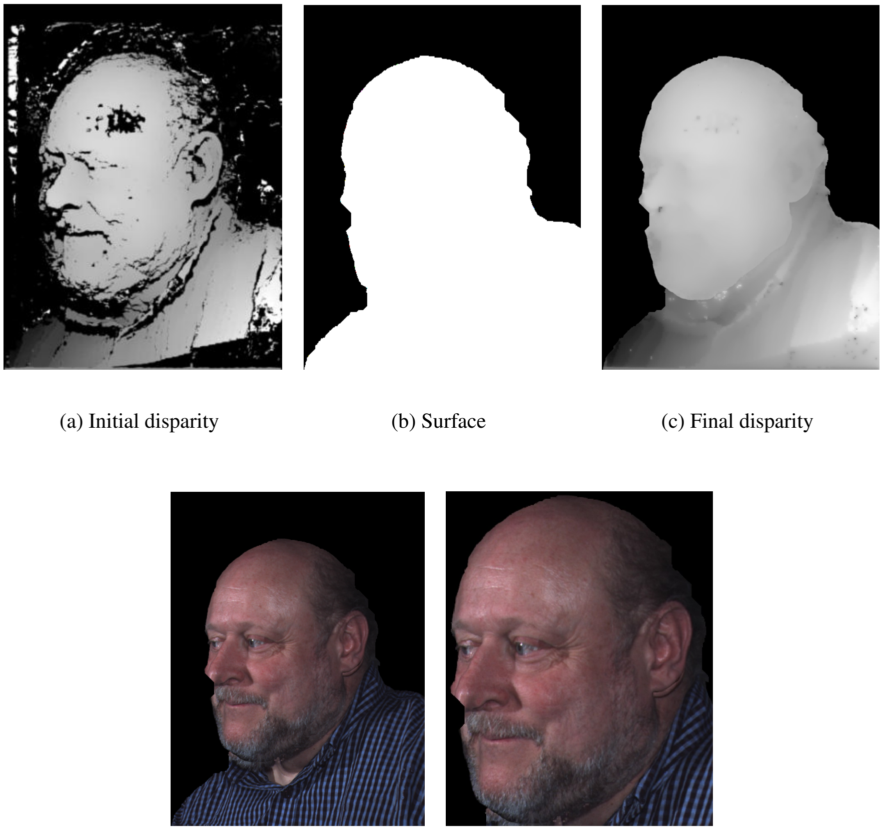
Figure 3.8(b) is the surface representing of the foreground area and Figure 3.8(c) is the final foreground disparity map after the interpolation. Figure 3.8(d) is Figure 3.3(b) after coloring. Figure 3.8(e) shows a zoom-in of the face. Figure 3.9 shows a series of virtual images interpolated. The images are interpolated along the line between two color cameras, but technically the virtual images could be generated anywhere on the screen. Figure 3.10 shows another test case. Both show a smooth viewpoint transition of the real cameras.

Whether this system is real-time or not depends on the algorithms chosen at every stage, especially on the foreground extraction, stereo matching, and interpolation. RBF interpolation introduced in this chapter is not easy to implement in real time. RBF interpolation can be substituted by simpler interpolation algorithms, such as linear interpolation, to make the system run faster. The interpolation step could even be skipped because the use of random dots pattern significantly enhance the accuracy and density of disparity maps.

3.5.2 Tests for Regular Stereo Images

We use the Middlebury dataset to test the extension of RBF interpolation to general stereo matching and to compare its performance with other methods. Figure 3.7 is one example - the “venus” stereo pair. Errors may occur when the floating segments are assigned to the wrong surface, as indicated by the yellow circle in the final disparity map. The “venus” stereo pair uses four surfaces (indicated by red digital numbers in 3.7(c)) to cover the scene. Figures 3.11 and 3.12 show more examples for the “baby” and “teddy” stereo pairs, which need four and eleven surfaces to compute the disparities.

We also compare our method using the Middlebury test bed [5] for quantitative analysis (Figure 3.13). Our approach ranks the 8th out of 123. This is because the proposed method has no strict constraint about the disparity distribution of a surface, which makes it more flexible for surface selection and hence has a greater capacity for error tolerance.



(a) Initial disparity

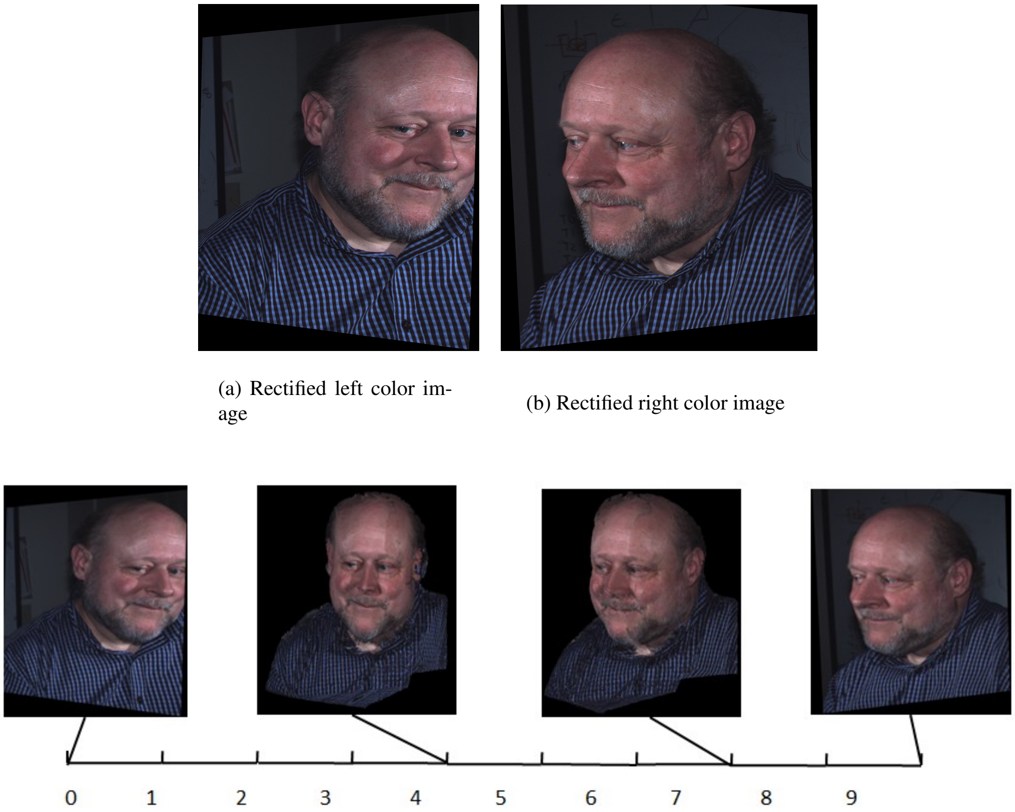
(b) Surface

(c) Final disparity

(d) Colored left IR image

(e) Zoom in on the face

Figure 3.8: Results of disparity extraction using our algorithm for the tele-presence system

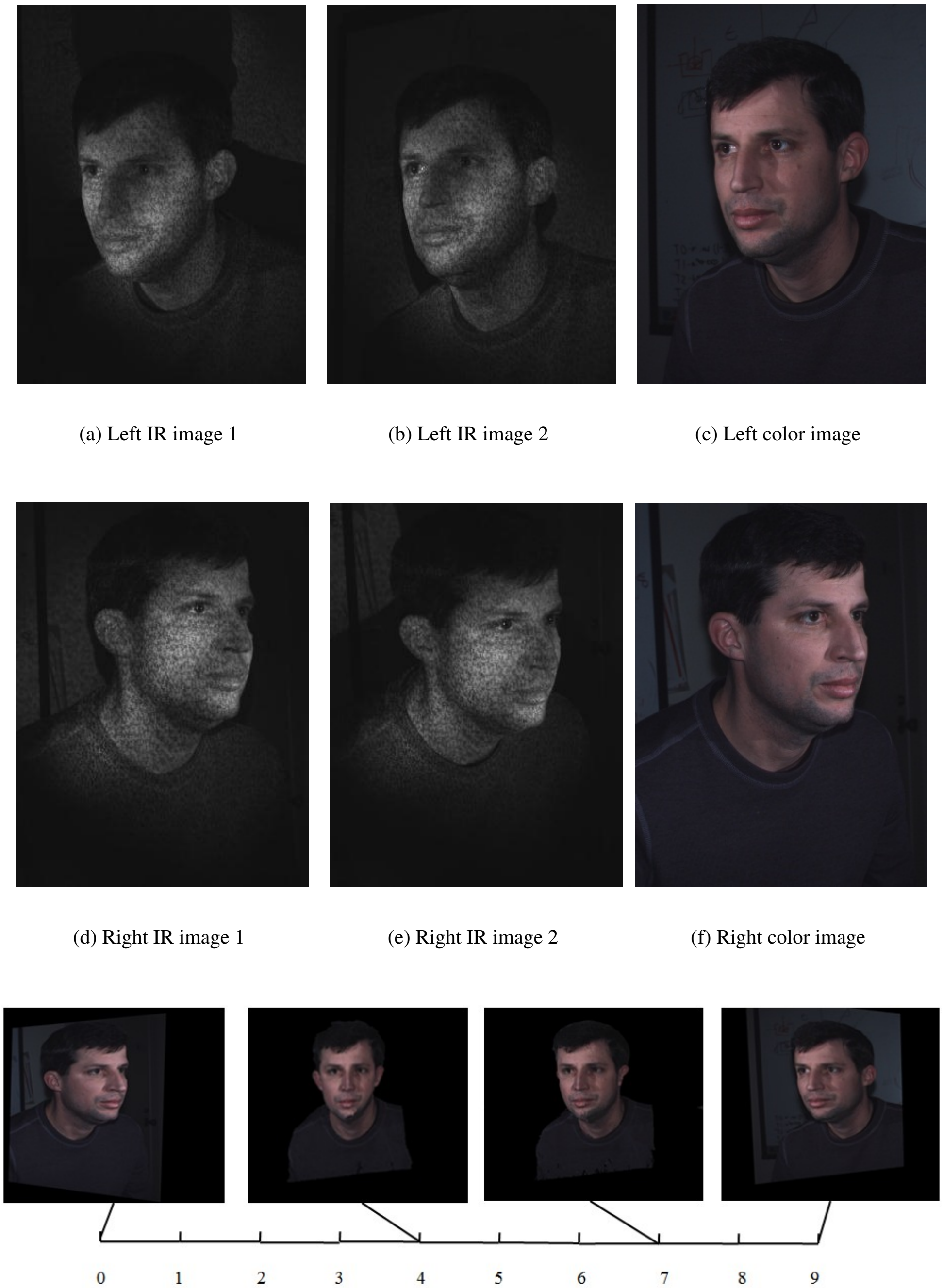


(a) Rectified left color image

(b) Rectified right color image

(c) Interpolated images

Figure 3.9: Example of view interpolation along the baseline of the two color cameras



(a) Left IR image 1

(b) Left IR image 2

(c) Left color image

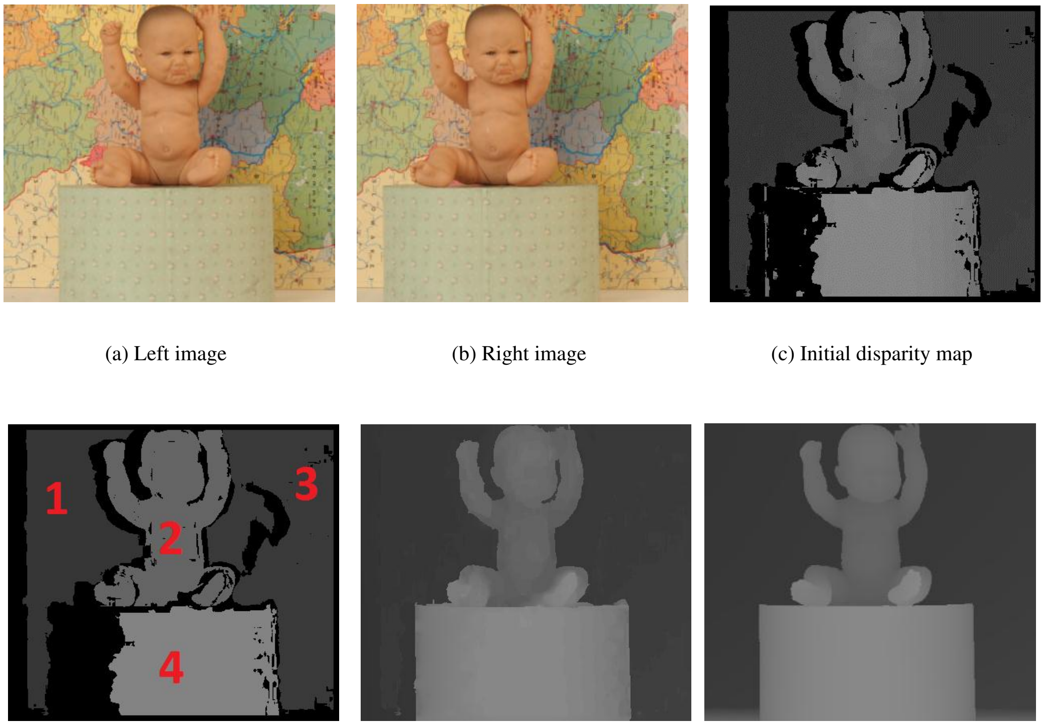
(d) Right IR image 1

(e) Right IR image 2

(f) Right color image

(g) Viewpoints transition from the left and right color image

Figure 3.10: A series of interpolated virtual images using the proposed algorithm



(a) Left image

(b) Right image

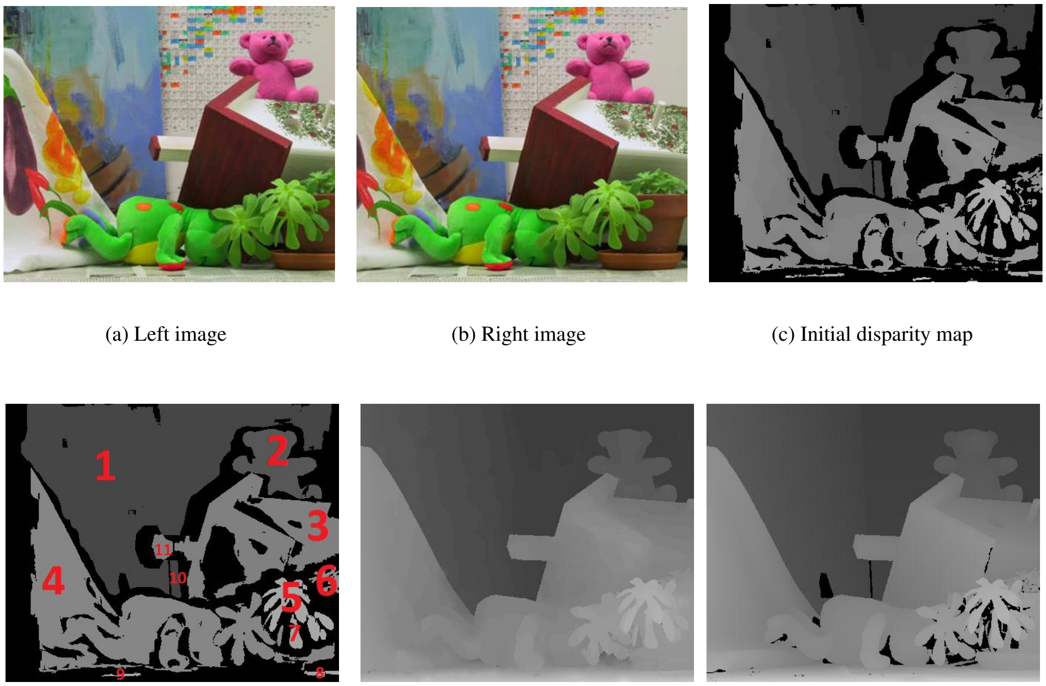
(c) Initial disparity map

(d) Surfaces

(e) Final disparity map

(f) Ground truth

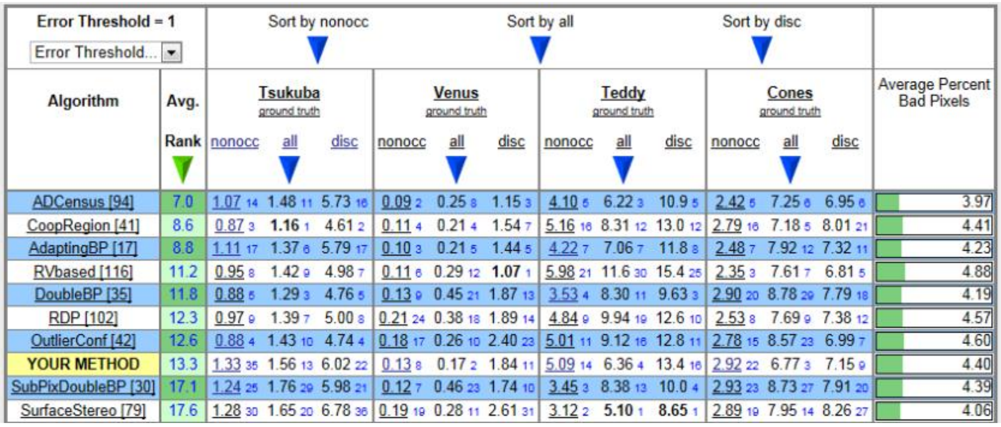
Figure 3.11: Disparity result of the “baby” stereo pair



(a) Left image (b) Right image (c) Initial disparity map

(d) Surfaces (e) Final disparity map (f) Ground truth

Figure 3.12: Disparity result of the “teddy” stereo pair



Error Threshold = 1														
Algorithm	Avg. Rank	Tsukuba ground truth			Venus ground truth			Teddy ground truth			Cones ground truth			Average Percent Bad Pixels
		nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	
ADCensus [94]	7.0	1.07 14	1.48 11	5.73 16	0.09 2	0.25 8	1.15 3	4.10 6	6.22 3	10.9 5	2.42 5	7.25 6	6.95 6	3.97
CoopRegion [41]	8.6	0.87 3	**1.16** 1	4.61 2	0.11 4	0.21 4	1.54 7	5.16 16	8.31 12	13.0 12	2.79 16	7.18 5	8.01 21	4.41
AdaptingBP [17]	8.8	1.11 17	1.37 6	5.79 17	0.10 3	0.21 5	1.44 5	4.22 7	7.06 7	11.8 8	2.48 7	7.92 12	7.32 11	4.23
RVbased [116]	11.2	0.95 8	1.42 9	4.98 7	0.11 6	0.29 12	**1.07** 1	5.98 21	11.6 30	15.4 25	2.35 3	7.61 7	6.81 5	4.88
DoubleBP [35]	11.8	0.88 5	1.29 3	4.76 5	0.13 9	0.45 21	1.87 13	3.53 4	8.30 11	9.63 3	2.90 20	8.78 29	7.79 18	4.19
RDP [102]	12.3	0.97 9	1.39 7	5.00 8	0.21 24	0.38 18	1.89 14	4.84 9	9.94 19	12.6 10	2.53 8	7.69 9	7.38 12	4.57
OutlierConf [42]	12.6	0.88 4	1.43 10	4.74 4	0.18 17	0.26 10	2.40 23	5.01 11	9.12 16	12.8 11	2.78 15	8.57 23	6.99 7	4.60
YOUR METHOD	13.3	1.33 35	1.56 13	6.02 22	0.13 8	0.17 2	1.84 11	5.09 14	6.36 4	13.4 16	2.92 22	6.77 3	7.15 9	4.40
SubPixDoubleBP [30]	17.1	1.24 25	1.76 29	5.98 21	0.12 7	0.46 23	1.74 10	3.45 3	8.38 13	10.0 4	2.93 23	8.73 27	7.91 20	4.39
SurfaceStereo [79]	17.6	1.28 30	1.65 20	6.78 36	0.19 19	0.28 11	2.61 31	3.12 2	**5.10** 1	**8.65** 1	2.89 19	7.95 14	8.26 27	4.06

Figure 3.13: Comparison of our method with other methods using Middlebury database

3.6 Conclusion

This chapter presents a solution to the eye contact correction problem in tele-presence systems with large baselines. At least two camera units are mounted on the monitor. Each set includes two IR cameras and one color camera and is responsible to capture/re-render partial scene. For each pixel in the left color image, our solution aims to find its corresponding position on the right color image, even though this pixel is invisible on the right color image, and vise verse. Therefore, virtual views can be created between this pair of color images by depth-based interpolation.

By sending both the color images and their disparity maps to all the other participants, our solution allows each participant to choose his/her own viewpoint. The viewpoints can be adjusted at any time.

The way to interpolate the initial disparity map of foreground can also be applied to general stereo matching after deciding the numbers of RBF interpolation surfaces. These impressive results demonstrate that the proposed method can generate excellent virtual views with lifelike eye contact, and works well for general stereo images in both visual and quantitative aspects.

Since the solution we propose is software-based, each step has to be within the limitation of computation capacity of computers. With the projection of the pattern with random dots, the initial disparity map of the foreground may be dense enough, thus the interpolation step could be skipped. There are also other ways to improve the quality of disparity maps, such as the methods we propose in Chapter 5.

Chapter 4

Foreground and Background Separation Using Disparity Map and Image Matting

As mentioned in the Chapter 1, one of the key elements to create a sense of tele-presence is to create the illusion that the participants are located in a common meeting room. In some commercial systems, this illusion is achieved by replicating the meeting room at the different sites with the same decoration, lighting, and background. In many ways, this is one of the reasons tele-presence rooms are so expensive. In order to reduce the cost and make tele-presence more accessible, the system should be able to extract each participant (the foreground) from its background environment and place all the participants in a common virtual meeting room as in the HP Coliseum system [22]. In this chapter, we will review various ways to extract the foreground from its background and explore a new way to perform this task by a novel method of image matting that can be used in the tele-presence context.

4.1 Foreground and Background Segmentation and the Problem

Numerous methods have been proposed to extract the foreground from the background automatically. The background subtraction is the simplest idea. The naive description of the background subtraction is that the foreground is extracted by computing the difference between the current frame and a known background. However, this method is affected by many factors, such as the camera noise and the illumination changes. A more sophisticate foreground and background segmentation is to use per-pixel modeling, for example Mixtures of Gaussian [119], non-parametric kernel density estimators [45], and Hidden Markov Model [107]. Those models could model complex backgrounds, but still have some severe limitations. Besides color cameras, there are some auxiliary devices that can capture an initial boundary of the foreground. For example, Wu *et al.* [138] use infrared camera to initialize the foreground boundary as the background is basically black in the infrared image. The rough boundary is easily detected by thresholding. Using various kinds of depth cameras, such as time-of-flight range sensor [135], range information can be used as well to pre-segment the foreground, since the depths of the foreground is much closer to the sensor than the background. In those methods, the initial segmentations are then refined by a relaxation labeling algorithm or other segmentation algorithms.

In tele-presence systems, the foregrounds usually contain images of humans with hair. Image segmentation works fine for clear boundaries, but it fails to handle hair or blurred boundaries. In order to maintain the completeness of foregrounds, we propose using image matting for the foreground extraction.

4.2 Literature Review of Image Matting

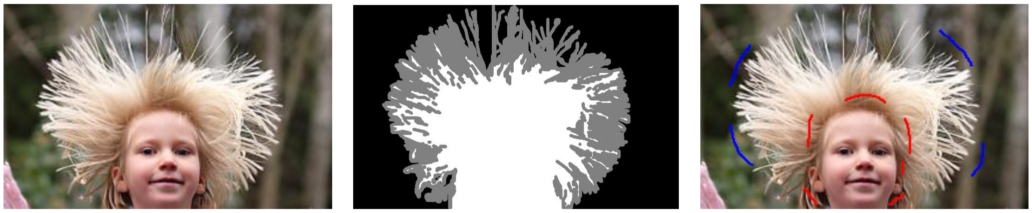
Image matting [102] is a well-known process used to extract foreground from its background and blend it with other backgrounds to create a new image seamlessly. Matting is one of the key steps in image editing and special effects in the film industry today. Image matting, also called soft segmentation, considers each pixel to be a convex combination of foreground color and background color. Image matting can be expressed mathematically as [133] :

$$\mathbf{I}(i,j) = \alpha_{i,j} * \mathbf{F}(i,j) + (1-\alpha_{i,j}) * \mathbf{B}(i,j), \tag{4.1}$$

where (i, j) are the 2D image coordinates of a pixel and $\mathbf{I}(i,j) = (C_R(i,j), C_G(i,j), C_B(i,j))$ represents the color of the pixel (i, j). $\mathbf{F}(i,j)$ and $\mathbf{B}(i,j)$ are the pixel (i, j) 's foreground and background colors, respectively. The parameter $\alpha_{i,j} \in [0, 1]$ measures the contribution of the foreground color and background color to a certain pixel. When $\alpha_{i,j} = 1$, this means that the pixel color is totally from the foreground, while $\alpha_{i,j} = 0$ means that the color is totally from the background. The image, storing the value of $\alpha_{i,j}$ for each pixel, is called the "matte". Image matting is an under-constrained problem, which has three equations with seven unknowns:

$$\begin{cases} I_R(i,j) = \alpha_{i,j} * F_R(i,j) + (1-\alpha_{i,j}) * B_R(i,j) \\ I_G(i,j) = \alpha_{i,j} * F_G(i,j) + (1-\alpha_{i,j}) * B_G(i,j). \\ I_B(i,j) = \alpha_{i,j} * F_B(i,j) + (1-\alpha_{i,j}) * B_B(i,j) \end{cases} \tag{4.2}$$

This is the reason why user intervention is necessary. Image matting starts with the blue screen matting [95, 117] where actors stand in front of a blue (or green) screen so that they can be easily segmented by making sure the foreground does not contain any color of the uniform background. This is very restrictive, but to date, this method is still the most frequently used in the film industry. Smith *et al.* [117] also introduce another approach called "triangulation". In this approach, the foreground is digitized in front of two known backgrounds colors: blue and green. This leads to two unknown parameters and two equations:



(a) Original image

(b) Trimap

(c) Scribbles

Figure 4.1: Interactions in image matting (images from [134])

$$\begin{cases} \mathbf{I_g}(i,j) = \alpha_{i,j} * \mathbf{F_g}(i,j) + (1-\alpha_{i,j}) * \mathbf{B_g}(i,j) \\ \mathbf{I_b}(i,j) = \alpha_{i,j} * \mathbf{F_b}(i,j) + (1-\alpha_{i,j}) * \mathbf{B_b}(i,j). \end{cases} \tag{4.3}$$

Hence, $\alpha_{i,j}$ can be solved by:

$$\alpha_{i,j} = 1 - \frac{\mathbf{I_g}(i,j) - \mathbf{I_b}(i,j)}{\mathbf{B_g}(i,j) - \mathbf{B_b}(i,j)}. \tag{4.4}$$

Matting with known background is easy to implement, but only limited to the studio environment. In recent years, more and more researchers have focused their work on images with natural backgrounds. There are two kinds of interventions worth mentioning: trimaps and scribbles. A trimap gives a fine, three-colored segmentation to indicate the Definite Foreground (DF), Definite Background (DB), and Unknown Region (UR) in white, black, and gray respectively (see Figure 4.1(b)). Alpha values in the DF and DB are set to 1 and 0 respectively. The colors in the DF and DB are used as the color samples to estimate the alpha values in the UR. The manually drawn trimaps can provide competitive mattes, but it is time-consuming for complex scenes and certainly not for cases like tele-presence systems which needs to be automatic and real-time. Some researchers have developed techniques to create the trimap automatically [89, 121, 70], but in many cases, the trimap produced is too coarse to deal with fine details accurately.

Scribble-based methods use scribble marks to indicate the possible color samples of the foreground (in red) and background (in blue). These scribbles, such as those illustrated in Figure 4.1(c), are typically drawn by users. Scribble-based methods save time on drawing fine trimap. Users can add more scribbles to improve the results. However, scribble-based methods are hard for inexperienced users to manipulate as the users must be careful to include all the color samples in the scribbles.

Depending on how the matte is inferred, image matting is classified into two main categories [134]: sampling-based and propagation-based, as shown in Table 4.1.

Sampling-based methods explicitly estimate the foreground color and background color for each pixel in the UR. Pixel colors in the DF and DB regions are used as color samples. The "Knockout" [39] computes the foreground colo as a weighted sum of the known foreground color samples in

Table 4.1: Categories of image matting techniques

	Sampling-based	Propagation-based
Trimap-based	Corporation 02 [39] Ruzon 00 [109] Chuang 01 [36] Rhemann 08 [106] He 01[56]	Sun 04 [120] Wang 07 a[134] Wang 07 b[131] Gastal 10[47] Grady 05[51]
Scribble-based	Wang 05[132]	Guan 06[53] Levin 06[82] Bai 07[20] Zheng 08[151] Zheng 09[150]
Other	Mishima 93[95] Smith 96 [117] Sun 06 [121] McGuire 05 [89] Joshi 06 [70]	

a neighborhood. Sun *et al.* [120] use the closest foreground and background colors as the estimated colors of an unknown pixel. Wang *et al.* [134] suggest a non-parametric optimized color estimation by assigning a confidence value to some closest color sample pairs. Only those pairs with high confidence values are selected. He *et al.* [56] suggest using the global sample set to avoid missing true samples and use a fast randomized search algorithm to decrease the computation burden. Rhemann *et al.* [106] find that the true color samples are not those closest in Euclidean distance but in geodesic distance.

There are also some methods that model the color distribution as the Mixture of Gaussians in order to find the best matched foreground and background colors. Ruzon *et al.* [109] first divide the unknown region into multiple sub-regions. Next, they set a window for each sub-region. Color samples are those pixels in the area of this window. Multiple Gaussian clusters are built on those foreground and background color samples respectively. The alpha value of a pixel is calculated by finding such a pair of foreground and background clusters, which can give this pixel the maximum probability in its distribution. This approach was improved by Bayesian matting in 2001. Chuang *et al.* [36] use the similar Gaussian distribution to form the distributions of foreground or background color samples, but select the color samples through a continuous sliding window. Moreover, the mattes are generated by solving the maximum a posteriori probability in a Bayesian framework. Wang *et al.* [132] train a Gaussian Mixture Model (GMM) from the scribbles. Each known pixel is assigned to a single Gaussian distribution in the GMM. The sample sets are pixels randomly selected from each Gaussian. They divide the pixels into U_c (certain) and U_n (uncertain) to represent whether or not the pixels have been processed. The problem is solved iteratively. In each iteration, first, update U_c by adding pixels in U_n that are close to those in U_c to U_c. Then, a Markov Random Field (MRF) is applied to the pixels in U_c and the matte is achieved by solving the energy function using belief propagation. Iterations repeat until no pixel is left in U_n or until convergence occurs.

Sampling-based methods heavily rely on how to gather color samples, locally or globally. Locally gathered samples only include the nearby colors. As a result, the matte will not be good if the true foreground or background colors are not nearby. Globally gathered samples have the ability to include the true color samples, but require large data storage and intensive computation making them hard to implement in real-time.

Propagation-based methods assume that the colors of pixels in a neighborhood follow a relationship, such as constant or linear. Using this assumption, alpha values can be propagated from the known pixels to the unknown pixels. Poisson matting [120] is proposed based on the assumption that the foreground and background colors are constant in a small neighborhood and thus $\nabla\mathbf{F}$ and $\nabla\mathbf{B}$ are approximated to be equal to zero. By performing the partial derivatives of Equation 4.1 on both sides:

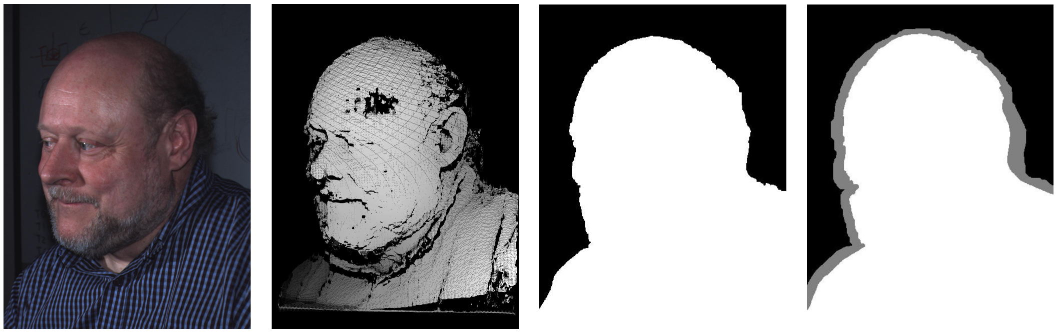
$$\nabla\mathbf{I} = (\mathbf{F} - \mathbf{B})\nabla\alpha + \alpha\nabla\mathbf{F} + (1 - \alpha)\nabla\mathbf{B}. \tag{4.5}$$

Equation 4.5 [120] could be approximated by the Poisson equation $\nabla\alpha \approx \frac{1}{|\mathbf{F}-\mathbf{B}|}|\nabla\mathbf{I}|$ and solved iteratively with a Dirichlet boundary condition.

Gastal *et al.* [47] assume that the pixels in a small neighborhood share the same foreground color, background color, and alpha value. Therefore, the computation on the sample selection is significantly decreased and thus this method is able to reach real-time performance. Wang [134] and Guan [53] assume that the alpha values vary smoothly in a small area and integrate the smoothness term in the energy function. In closed-form matting [82], Levin *et al.* assume the foreground and background colors could fit into a linear model in the local area, turning the matting problem into quadratic optimization. Zheng *et al.* [150] also solve the matting problem by assuming a linear relationship among neighboring pixels and train an alpha model for each pixel in the unknown region. The alpha values are calculated by determining the coefficients of the alpha model.

FuzzyMatte [151] is an online matting algorithm that defines an affinity $\mathcal{A}$ between every two neighbor pixels. The foreground and background colors of a pixel are the color samples which have the strongest path connection with this pixel.

Propagation-based methods frequently fail if the foreground or background has a complex texture as the assumption that colors follow a constant or linear assumption is not true anymore. The same problem also happens with sampling-based methods. Some approaches [120, 134, 151] both assume an affine relationship in the neighbor and estimate the foreground and background colors for each unknown pixel. These methods can be thought as a combination of the two categories [133] and produce the best results so far as they can overcome each other's disadvantages.



(a) Color image

(b) Corresponding initial disparity map

(c) Initial foreground boundary

(d) Trimap

Figure 4.2: Automatic generation of the trimap using disparity

4.3 Proposed Method

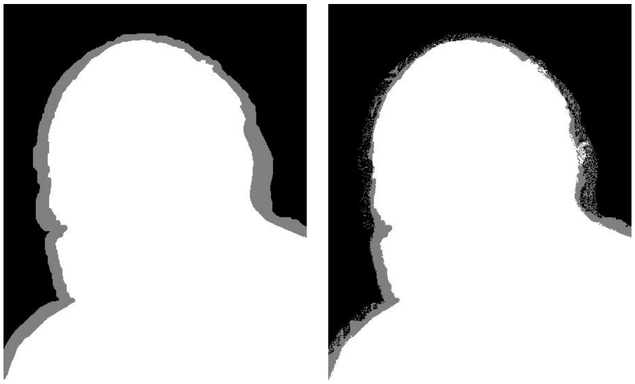
In this section, we first describe the proposed method for the automatic foreground and background separation using disparity information computed with the system described in Chapter 3. We then present how this initial segmentation can be further refined using an improved matting method. This new matting method is a combination of sampling-based and propagation-based algorithms. We search the foreground/background color samples using the trimap generated automatically by the disparity information, and then optimize this initial matte globally with a smoothness constraint in a small neighborhood.

4.3.1 Automatic Generation of the Trimap

After the stereo matching step, the generated initial disparity map of the IR cameras is re-projected to the position of the color camera. The foreground is composed by those pixels whose disparity values are within the threshold range. The threshold range is $[20, 250]$ in Figure 4.2. Small areas (less than 400 pixels) are thought as noises and deleted too. Erosion and dilation operations are applied to the initial boundary to create a gray belt area in Figure 4.2(d), which is the unknown region. The pixels in the unknown region are ready to be processed in the following matting steps.

4.3.2 Trimap Refinement

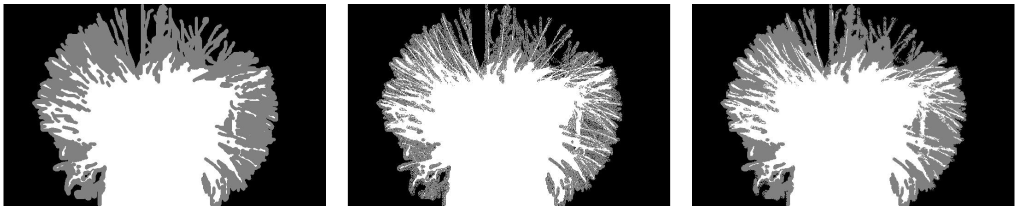
As we know, the trimap should be as fine as possible, since the pixels in the UR are closer to their true foreground and background colors. As shown in the previous chapter, it is hard to make a fine trimap using disparity information alone. In this step, the DF and DB are moved towards the UR to detect all possible pixels in the UR which should be in the DF or DB. We assume that the DF and DB have already included all the color samples, even in a coarse trimap. Following this idea, we first do a search in the DF and DB for each pixel $\mathbf{p}$ with the color $\mathbf{I}$ in the UR. If there is a match (equal in color), then the pixel $\mathbf{p}$ is marked as white or black. Then, we delete all the black or white



(a) Before

(b) After

Figure 4.3: Trimap refinement



(a) Original trimap

(b) Samples expansion into UR

(c) Trimap after deleting small regions

Figure 4.4: An example of fine trimap refinement

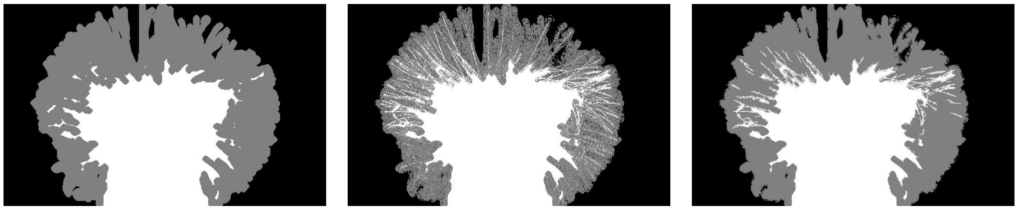
regions smaller than a threshold (for example 50 pixels) to get rid of the noise. Figure 4.3 shows the original trimap (Figure 4.2(d)) and the refined one.

The trimap refinement does not look useful if the original trimap is fine enough, but it is helpful to refine coarse trimaps. Figure 4.4 and 4.5 show such two cases.

4.3.3 Estimation of Foreground and Background Colors

After refining the trimap, we estimate the foreground and background colors using a simplified non-parametric color estimation in robust matting [134]. We introduce the original color estimation first. The color samples are those along the boundaries of the DF and DB. Every pixel **p** in the UR selects its nearest M color samples from the DF and nearest N samples from the DB. Each pair of foreground and background colors $(\mathbf{F}^i, \mathbf{B}^j)$ results in an alpha value of $\hat{\alpha}$. A confidence value is given to measure how well the pair $(\mathbf{F}^i, \mathbf{B}^j)$ fits the actual color of a pixel. The confidence value computed by Equation 4.6 [134] consists of three terms: $RD(\mathbf{F}^i, \mathbf{B}^j)$, $\omega(\mathbf{F}^i)$, and $\omega(\mathbf{B}^j)$.

$$conf(\mathbf{F}^i, \mathbf{B}^j) = exp\{-\frac{RD(\mathbf{F}^i, \mathbf{B}^j) \cdot \omega(\mathbf{F}^i) \cdot \omega(\mathbf{B}^j)}{\sigma^2}\}, \tag{4.6}$$



(a) Original trimap

(b) Samples expansion into UR

(c) Trimap after deleting small regions

Figure 4.5: An example of coarse trimap refinement

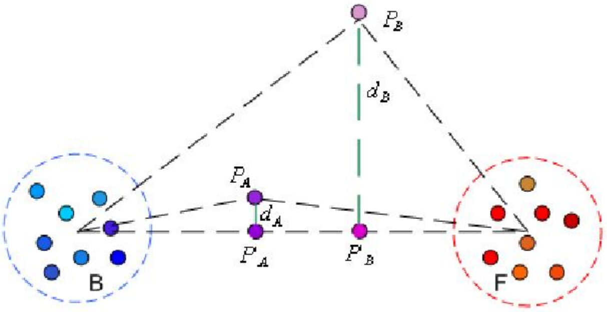


Figure 4.6: The distance of the real color and inferred color from the linear relationship [134]

where σ is assumed constant. The function $RD(\mathbf{F}^i, \mathbf{B}^j)$ is the distance ratio measuring the difference between the real color $\mathbf{I}$ and the estimated color inferred from Equation 4.1 given $\hat{\alpha}_{i,j}$. $RD(\mathbf{F}^i, \mathbf{B}^j)$ [134] is defined as:

$$RD(\mathbf{F}^i, \mathbf{B}^j) = \frac{\|\mathbf{I} - (\hat{\alpha}\mathbf{F}^i + (1 - \hat{\alpha})\mathbf{B}^j)\|}{\|\mathbf{F}^i - \mathbf{B}^j\|}. \tag{4.7}$$

The parameter $\hat{\alpha}$ in Equation 4.7 is the alpha value given to a pair of color samples $(\mathbf{F}^i, \mathbf{B}^j)$:

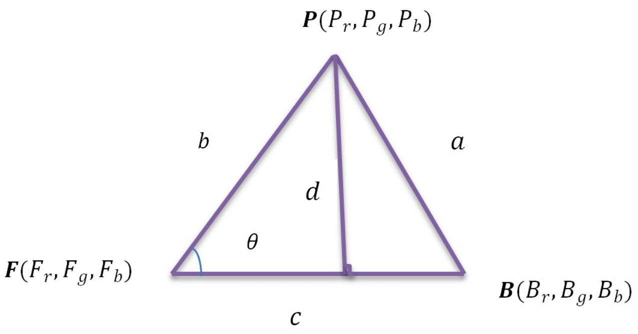
$$\hat{\alpha} = \frac{(\mathbf{I} - \mathbf{B}^j)(\mathbf{F}^i - \mathbf{B}^j)}{\|\mathbf{F}^i - \mathbf{B}^j\|^2}. \tag{4.8}$$

According to the matting function (Equation 4.1), the color $\mathbf{I}$ has a linear relationship with $\mathbf{F}^i$ and $\mathbf{B}^j$. This could explain why $\mathbf{P_A}$ is a better choice than $\mathbf{P_B}$ in Figure 4.6 [134].

In addition, there is a bias as we assume that most of the pixels in the UR should be either from foreground or background. Therefore, the terms $\omega(\mathbf{F}^i)$ and $\omega(\mathbf{B}^j)$ are weighting functions designed to assign a high weight if the color $\mathbf{I}$ is close to the DF or DB color samples and are defined by [134]:

$$\omega(\mathbf{F}^i) = exp\{\frac{-\|\mathbf{F}^i - \mathbf{I}\|^2}{(min_i(\|\mathbf{F}^i - \mathbf{I}\|))^2}\}, \tag{4.9}$$

$$\omega(\mathbf{B}^j) = exp\{\frac{-\|\mathbf{B}^j - \mathbf{I}\|^2}{(min_j(\|\mathbf{B}^i - \mathbf{I}\|))^2}\}. \tag{4.10}$$



$\boldsymbol{P}(P_r, P_g, P_b)$
b
a
d
θ
$\boldsymbol{F}(F_r, F_g, F_b)$
$\boldsymbol{B}(B_r, B_g, B_b)$
c

Figure 4.7: Illustration of distance from a point to a line in 3D space

The initial matte is the average alpha value of pixels with the top three confidences. In the next step, the initial matte is used to initialize the global optimization procedure based on a random walk.

We will see in the following section that the global optimization used in our proposed method is based on mutual information [115], which is not so sensitive to the initialization. Thus, one can skip the calculation of the initial alpha (Equation 4.8) and compute the RD directly.

Note that the numerator of RD is actually the distance d_A or d_B in Figure 4.6, which can be measured directly from the geometry - the distance between a point and a line in 3D space. Then, the problem is represented as (shown in Figure 4.7): given the coordinates of 3D point **P**, **F**, and **B**, calculate the distance d from another 3D point **P** to the line $(\mathbf{F}(i,j), \mathbf{B}(i,j))$. In our case, the 3D space is the RGB color space, so the coordinates are values in the R, G, and B channel. The coordinates of **P**, **F**, and **B** are (P_r, P_g, P_b), (F_r, F_g, F_b), and (B_r, B_g, B_b).

Let's define a, b, and c as the edges of the triangle shown in Figure 4.7. Their lengths are defined as:

$$\begin{cases} a = \|\mathbf{P} - \mathbf{B}\| \\ b = \|\mathbf{P} - \mathbf{F}\|, \\ c = \|\mathbf{F} - \mathbf{B}\| \end{cases} \tag{4.11}$$

where $\|\cdot\|$ means the Euclidian distance.
The edges and angles satisfy the triangulation identity:

$$a^2 = b^2 + c^2 - 2bc\cos\theta. \tag{4.12}$$

The distance d in Figure 4.7 is inferred from:

$$d = b\sin\theta = b\sqrt{1-(\cos\theta)^2} = b\sqrt{1-(\frac{b^2+c^2-a^2}{2bc})^2} = \frac{\sqrt{(2bc)^2-(b^2+c^2-a^2)^2}}{2c}. \tag{4.13}$$

Therefore, one can simplify $RD(i,j)$ as:

$$RD(\mathbf{F}^i, \mathbf{B}^j) = \frac{d}{c} = \frac{\sqrt{(2bc)^2-(b^2+c^2-a^2)^2}}{2c^2}. \tag{4.14}$$

The estimated foreground and background color pair is the average of the pairs with top three confidence values.

4.3.4 Energy Function and Global Optimization

With the pre-estimated foreground and background colors $\mathbf{F}^{'}$ and $\mathbf{B}^{'}$, one can create a temporary image $\mathbf{I}^{'}$ using Equation 4.1: $\mathbf{I}^{'} = \alpha\mathbf{F}^{'} + (1-\alpha)\mathbf{B}^{'}$, where α is unknown and need to be solved. The energy function built on Markov Random Field is defined as:

$$E(\alpha) = E_{data}(\alpha) + \lambda E_{smooth}(\alpha). \tag{4.15}$$

The data term is the sum of the dissimilarity between two pixels $\mathbf{I}(\mathbf{p})$ and $\mathbf{I}^{'}(\mathbf{p})$ and is defined by:

$$E_{data}(\alpha) = \sum_{\mathbf{p}} D_{\mathbf{p}}(\alpha_{\mathbf{p}}), \tag{4.16}$$

where $D_{\mathbf{p}}(\alpha_{\mathbf{p}})$ is the measurement of dissimilarity of two pixels. Some commonly used $D_{\mathbf{p}}$ are the Sum of Absolute Differences (SAD), Sum of Squared Differences (SSD), or truncated SSD. In this thesis, we propose to use mutual information [115] as a similarity function since mutual information has been proven to be a better measurement in similarity. The concept of mutual information was first introduced by Shannon in 1948 to measure the dependence of two random variables. The greater the mutual information is, the more similar the two variables are.

Let's assume that $\mathbf{I_1}$ and $\mathbf{I_2}$ are two images. Mutual information is expressed as [115]:

$$MI(\mathbf{I_1},\mathbf{I_2}) = H(\mathbf{I_1}) + H(\mathbf{I_2}) - H(\mathbf{I_1},\mathbf{I_2}), \tag{4.17}$$

where $H(\mathbf{I_1})$ and $H(\mathbf{I_2})$ are the marginal entropies of the images $\mathbf{I_1}$ and $\mathbf{I_2}$, while $H(\mathbf{I_1},\mathbf{I_2})$ is their joint entropy. Mutual information reaches its maximum when two images are identical. The entropy functions are expressed in discrete form as [115]:

$$H(\mathbf{I_1}) = \sum_{\mathbf{p}\in\mathbf{I_1}} p(\mathbf{p}) \log p(\mathbf{p}), \tag{4.18}$$

$$H(\mathbf{I_2}) = \sum_{\mathbf{q}\in\mathbf{I_2}} p(\mathbf{q}) \log p(\mathbf{q}), \tag{4.19}$$

$$H(\mathbf{I_1},\mathbf{I_2}) = \sum_{\mathbf{p}\in\mathbf{I_1}} \sum_{\mathbf{q}\in\mathbf{I_2}} p(\mathbf{p},\mathbf{q}) \log p(\mathbf{p},\mathbf{q}). \tag{4.20}$$

However, $MI(\mathbf{I},\mathbf{I}^{'})$ measures the total similarity of two images $\mathbf{I}$ and $\mathbf{I}^{'}$ while does not fit the form of the data term. Kim *et al.* [74] proposed a solution to this problem. First, since $H(\mathbf{I})$ does not depend on the alpha values and $H(\mathbf{I}^{'})$ can be regarded as constant, thus the data term of Equation 4.16 could be approximated by joint entropy:

$$E_{data}(\alpha) = \sum_{\mathbf{p}} D_{\mathbf{p}}(\alpha_{\mathbf{p}}) = -MI(\mathbf{I}, \mathbf{I}^{'}) \approx H(\mathbf{I}, \mathbf{I}^{'}). \qquad (4.21)$$

A negative sign is added in front of the term $MI(\mathbf{I}, \mathbf{I}^{'})$ to maximize the mutual information by minimizing the energy function [74]. They then brought the Taylor expansion in so that $H(\mathbf{I}, \mathbf{I'})$ can be written in the form of a pixel-wise sum by deleting some terms with high exponents and this solution has been successfully applied to the stereo matching problem. We use a similar framework to the image matting problem. **I** is the original image and **I'** is the temporary image. The probability distribution is defined as the Gaussian convolution of the joint histogram of the two images [74] and Equation 4.22 is the application to our case:

$$P_{\alpha_0}(\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha)) = (\frac{1}{|\mathbf{P}|} * JH_{\alpha_0}(\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha))) \otimes G(\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha)), \qquad (4.22)$$

where

$$JH_{\alpha_0}(\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha)) = \sum_{\mathbf{p}} T[(c_1, c_2) = (\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha_0))], \qquad (4.23)$$

is the joint histogram of the images **I** and $\mathbf{I}^{'}$. $\mathbf{I}(\mathbf{p})$ is the color of pixel **p** in image **I**. $\mathbf{I}^{'}(\mathbf{p}, \alpha_0)$ is the color of pixel **p** in image $\mathbf{I}^{'}$ and α_0 is the current value for the variable α. $\mathbf{c}_1$ and $\mathbf{c}_2$ are the color values in $[0, 255]$. $T[\,]$ is 1 if the argument is true; otherwise, $T[\,]$ is 0. Therefore, the bin (c_1, c_2) adds one if $\mathbf{I}(\mathbf{p}) = c_1$ and $\mathbf{I}^{'}(\mathbf{p}, \alpha_0) = c_2$. G is a 2D Gaussian function. $|P|$ is the number of pixels in an image. Finally, the data term is expressed as:

$$E_{data}(\alpha) = -\sum_{\mathbf{p}} (\frac{1}{|P|} * P_{\alpha_0}(\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha))) \otimes G(\mathbf{I}(\mathbf{p}), \mathbf{I}^{'}(\mathbf{p}, \alpha)). \qquad (4.24)$$

Please refer to Kim's paper [74] for more details. The minimization of the energy function starts from an initial matte α_0 and the optimization iterates until convergence. In our experiments, the initial alpha values α_0 for pixels in the UR are all set as 0.5.

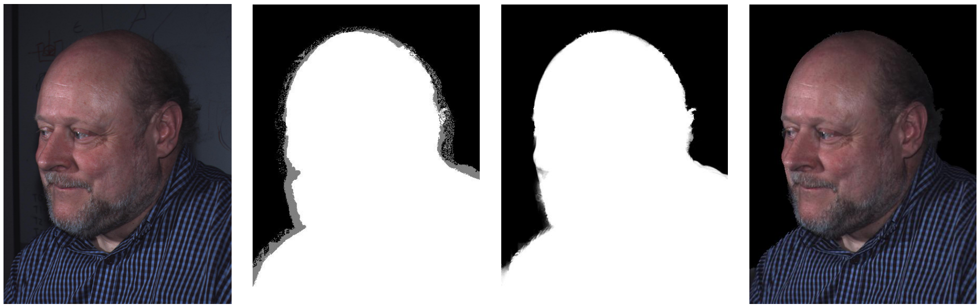
The smoothness term is defined in Equation 4.25 to keep the continuity of alpha values in a neighborhood,

$$E_{smooth}(\alpha) = \sum_{\mathbf{q} \in N(\mathbf{p})} |\alpha(\mathbf{p}) - \alpha(\mathbf{q})|. \qquad (4.25)$$

We use $alpha$- expansion Graph Cuts [24] (described in Algorithm 4.1) to solve the energy function. The $alpha$ is set as 100 levels, which corresponds to the matte value from 0.0 to 1.0 with an interval of 0.01.

Algorithm 4.1 Pseudo code of $alpha$-expansion

```
1:  start with an arbitrary initial labeling $f$
2:  $success = false$
3:  for each label $\alpha \in L$ do
4:      find the labeling $f^{'}$ with lowest $E(f^{'})$ with one $alpha$-expansion of $f$
5:      if $E(f^{'}) < E(f)$ then
6:          $f = f^{'}$ and $success = true$
7:      end if
8:  end for
9:  if $success == true$ then
10:     GOTO 2
11: end if
12: return $f$
```



(a) Color image

(b) Trimap

(c) Matte

(d) Extracted foreground

Figure 4.8: The matte and the extracted foreground of the tele-presence system

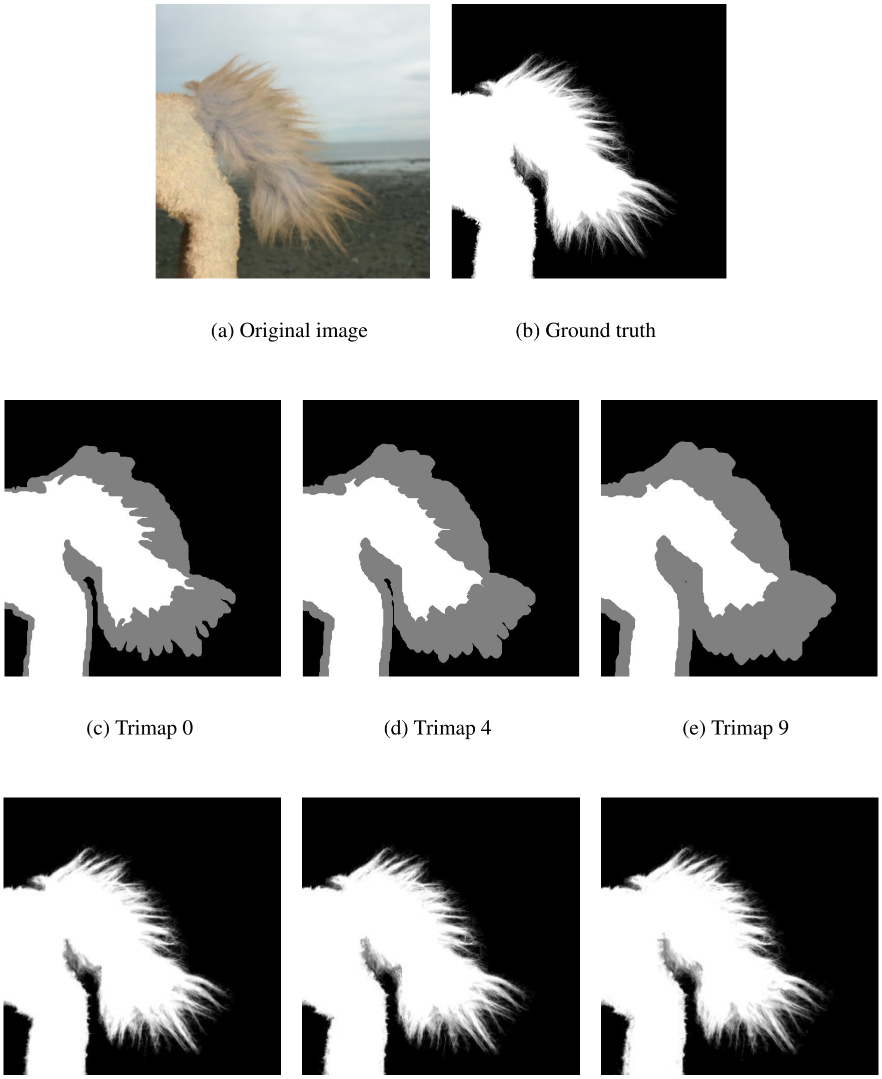
4.4 Experimental Results

Figure 4.8 shows the matting result for the tele-presence images. Compared to the segmentation result, in Figure 4.8, the eyelash and the hairs on the back of the head are well kept. To further test and compare the performance of our proposed matting method, we use the dataset [8] provided by Dr. Wang. This dataset gives eight test cases (T_1 to T_8) and each test case contains 10 levels of trimaps in different fineness. We use three of them: trimap 0, 4, and 9. This section shows both the visual and quantitative evaluations of our method, as well as and the comparisons with the state-of -the-art matting algorithms.

4.4.1 Visual Comparisons

First, we compare our results with ground truth visually. Figures 4.9 and 4.10 show results of two test cases. These datasets are specifically selected as they all contain large portion of hairs.

Figures 4.9 and 4.10 are only two test cases posted in this thesis, but the other test cases also prove the effectiveness of our proposed method. The way to choose the color samples is important. Since only the 20 closest pixels in the DF/DB are chosen in the color samples, it is very likely



(a) Original image

(b) Ground truth

(c) Trimap 0

(d) Trimap 4

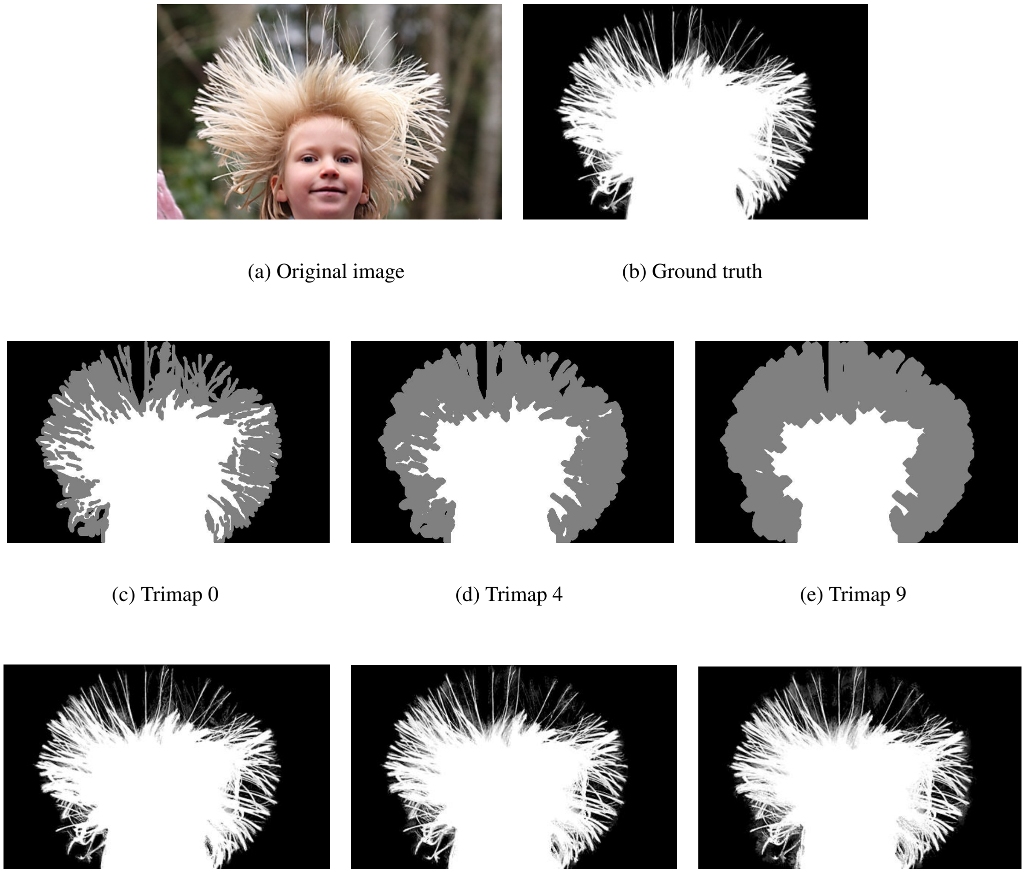
(e) Trimap 9

(f) Matte 0

(g) Matte 4

(h) Matte 9

Figure 4.9: Mattes of T_2 in trimap levels 0, 4, and 9



(a) Original image

(b) Ground truth

(c) Trimap 0

(d) Trimap 4

(e) Trimap 9

(f) Matte 0

(g) Matte 4

(h) Matte 9

Figure 4.10: Mattes of T_8 in trimap levels 0, 4, and 9

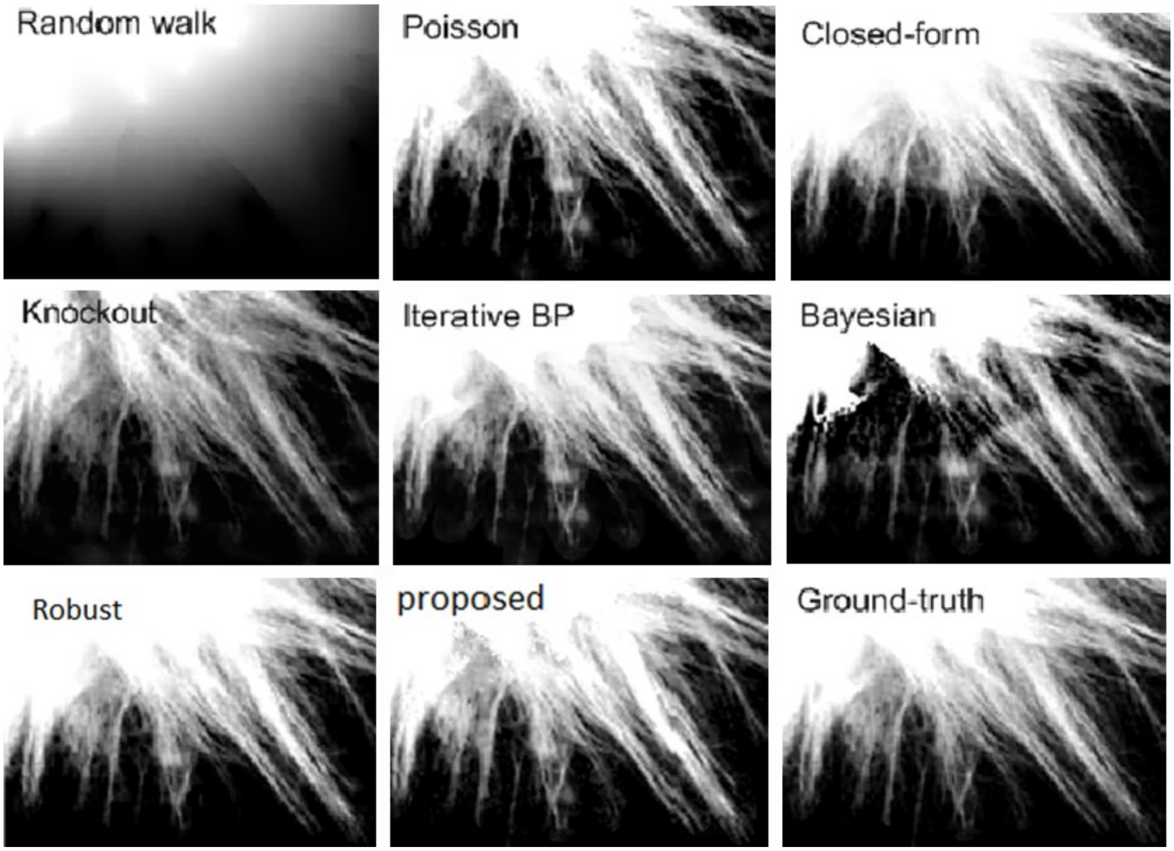


Figure 4.11: The comparison of our results with state-of-the-arts on partial matte T_2 (from trimap 0)

that the real colors are missed because they are spatially far away, especially in a coarse trimap. Moreover, it is hard to create a precise matte when the foreground and background colors are close, which causes ambiguity.

In Figures 4.11 and 4.12, we zoom in on parts of the mattes from Figures 4.9(f) and 4.10(f), and compare them with other popular approaches. They are robust matting [134], Closed-form matting [82], iterative BP matting [132], Bayesian matting [36], Poisson matting [120], Knockout matting [39], and Random-Walk matting [51]. The results of these approaches listed in Figures 4.11 and 4.12 are from [134].

Bayesian, Poisson, and Knockout matting methods are sampling-based methods so that they heavily depend on the trimaps' color estimation and fineness. Propagation-based methods produce much better and more stable mattes as their assumptions are more likely to be true, but these methods may fail if the texture patterns are complex or there are holes in the foreground [56]. Our method and robust matting are visually the best and most stable. However, since they only collect a few nearby colors as color samples, they fail if the true foreground and background colors are not in the samples.

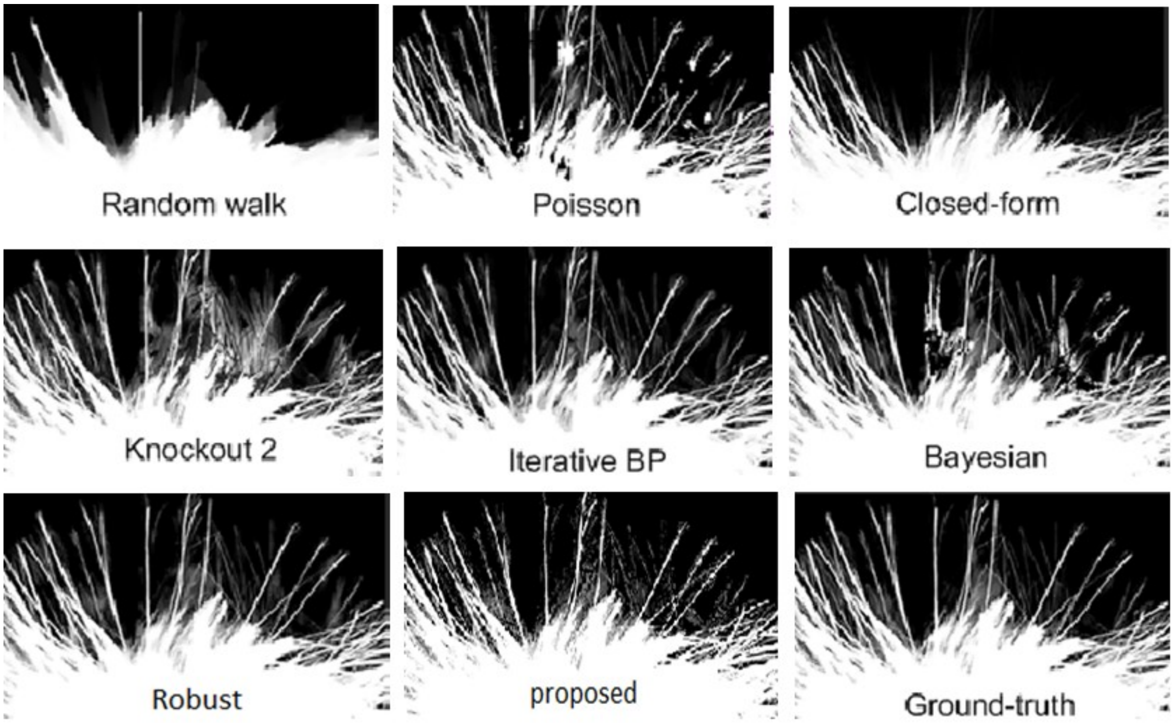


Figure 4.12: The comparison of our results with state-of-the-arts on partial matte T_8 (from trimap 0)

Table 4.2: Quantitative evaluation of the proposed method

		T1	T2	T3	T4	T5	T6	T7	T8
trimap 0	MSE	63.19	136.88	71.62	68.36	24.48	212.83	100.53	275.01
	ER	12.46	16.34	8.20	6.46	3.40	5.05	5.80	0.18
trimap 4	MSE	88.24	148.64	103.62	123.90	73.93	411.75	150.82	364.26
	ER	12.99	16.89	8.88	7.78	4.31	7.19	7.16	0.21
trimap 9	MSE	112.06	196.82	151.80	203.46	123.81	597.74	243.60	452.91
	ER	13.74	17.64	9.95	9.31	5.69	9.61	8.63	0.23

4.4.2 Quantitative Evaluations

Mean Squared Errors (MSE) and the Error Ratio (ER), two statistic standards commonly used to evaluate the matte's accuracy, are used as the quantitative measurements in our experiments. ER is the percentage of the alpha values unequal to the ground truth.

In all the test images in Table 4.2, the quality of the mattes does not degrade too much when the trimaps become coarser. The trimap refinement expands the colors in the DF and DB to the UR. It helps to make a smooth growth in MSE and ER. However, in the coarse trimaps, the DF and DB do not contain all the possible samples. This results in the situation that the true foreground and background colors are not in the DF and DB, and causes errors.

In addition, it is easy to extend our method to scribbles-based methods by using all the colors indicated by scribbles as color samples. The samples are updated when more scribbles are added in. The numbers of samples can be limited by a threshold T. If the numbers exceed T, the closest T

colors are selected.

4.5 Conclusion

In this chapter, we propose a novel method to separate the foreground from its background by using an image matting technique initialized by a rough disparity map. The color estimation is made by simplifying the procedure in robust matting since the global optimization used in our method is not sensitive to the initialization. Mutual information is re-defined to perform as the data term in the energy function built on Markov Random Field. Intensive experimentations demonstrate that the proposed matting method can separate both the main body and the fine details on the foreground from the background. The boundaries of the foreground from matting algorithms are more realistic than those from the segmentation. Fewer artifacts are essential for upgrading the immersive performance of the tele-presence systems. However, it is still too early to integrate the method into our system because it does not meet the real-time performance requirement. Mutual information is good at similarity measurement, but it is relatively slow. In the future, we will focus on simplifying the global optimization and the color estimation problem. A global sampling algorithm is needed to introduce the correct foreground and background colors to the pixels in the UR.

Chapter 5

Illumination Invariant Stereo Matching

In Chapter 3, we discuss how to use stereo matching and the projection of an infrared pattern onto the scene to help find the correspondence between images. Contrary to the standard views found in the literature, there is no guarantee that corresponding pixels in a pair of stereo images will have the same color intensity unless we have ideal conditions: Lambertian surfaces, uniform lighting, and uniform sensors sensitivities. In order to solve this problem, we have developed two new illumination invariant stereo matching algorithms. The first algorithm is based on an improved *Census* method which is robust to general illumination changes. The second algorithm is based on relative gradients and is designed for a non-Lambertian reflection model. The speed is also a concern since tele-presence systems must run in real time. To achieve the real-time performance, we limite our research on local stereo matching methods, as opposed to global methods, they are amiable to be speeded up with parallel computing architecture like the Graphics Processing Unit (GPU).

5.1 Introduction

The aim of stereo matching is to find the corresponding pixels from two images taken by cameras separated by some baseline distance. Most stereo matching algorithms (as discussed in Chapter 2) are based on color matching [111] which assumes that the corresponding pixels have similar color intensity. It's a long journey for a color light ray to go from a 3D point in the real word to a 2D pixel on the image plane. It is also a complicated process since the final color on the image plane is affected by many factors such as: the illumination energy, the surface reflection properties, pixel sensitivity, camera inner processing, etc. Figure 5.1 explains how a 3D point becomes a pixel.

There are two key elements in this process: transportation of light rays from an object to the camera sensor and the conversion of these light-rays into digital signals that are processed by the computer.

1) Transportation of Light Rays

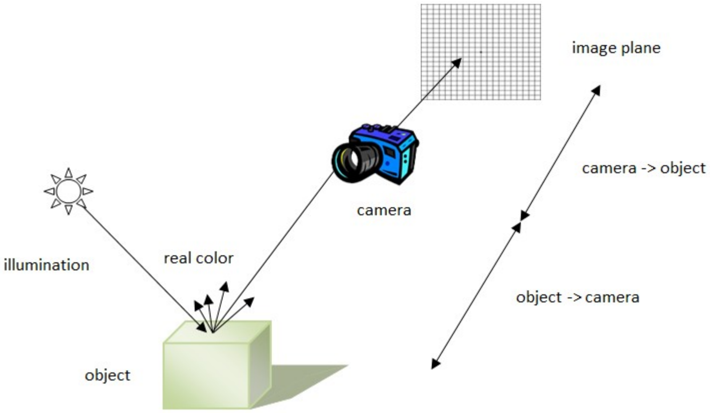


Figure 5.1: Real-world image formation

There are two main models analyzing how light reflects on the surface of an object: Lambertian and non- Lambertian models. If an object surface is assumed Lambertian, then the colors of each 3D point captured by different cameras should be independent where the cameras are located. This is due to the fact that the Lambertian surface reflects the incident light in all directions with the same power. The reflection model simulating the Lambertian surface reflection can be expressed by [114]:

$$\mathbf{C} = (\boldsymbol{n} \cdot \boldsymbol{i}) \int_{\lambda} f_c(\lambda)e(\lambda)b(\lambda)d\lambda + \int_{\lambda} f_c(\lambda)a(\lambda)k_a(\lambda)d\lambda. \tag{5.1}$$

This model relates to the illumination energy e, the diffuse reflection ratio b, the sensor sensitivity f_c, and the illumination energy from other objects a. The parameter k_a is the ratio of the energy a working on the surface and λ is the wavelength of the color. The parameter ***n*** is the normal of the surface patch, ***i*** is the normal of the direction of the illumination source. The first term in Equation 5.1 is the diffuse reflection (body reflection), which reflects illuminations in every direction with the same power. The second term is the ambient reflection which models the diffuse lights from other objects in the scene. If the surface of an object follows this model, the radiometric property of the object's surface is view-independent and is the assumption for many stereo algorithms in the literature. Color variations are only caused by the illumination energy e. Typically, color sensors are composed of filters in front of the Charge-Coupled Device (CCD) that absorb most wavelength except for the basic three color RGB with a peak response at 625 nm for red, 550 nm for green, and 450 nm for blue, respectively (see Figure 5.2).

By using this simple scheme, the color Equation 5.1 can be simplified as:

$$\begin{cases} C_R = e(R)b(R) + a(R)k_a(R) \\ C_G = e(G)b(G) + a(G)k_a(G), \\ C_B = e(B)b(B) + a(B)k_a(B) \end{cases} \tag{5.2}$$

where C_R, C_G, and C_B are view-independent and are considered to be the basic color components of a 3D point.

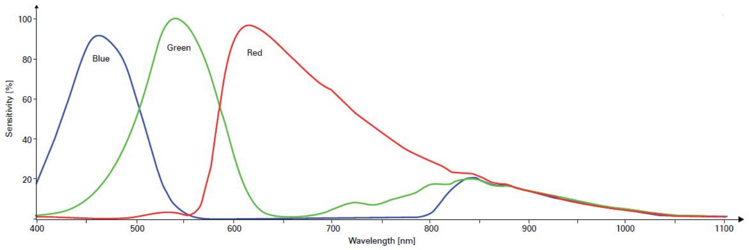


Figure 5.2: Spectral response of Sony ICX098BQ CCD used in many firewire cameras (image from [16])

However, most objects in the real world are not Lambertian making object reflectivity view dependent. View dependent colors vary among multiple cameras and make simple color matching algorithms unreliable. According to Shafer [114], non-Lambertian surface reflections can be modeled as:

$$\mathbf{C} = (\boldsymbol{n} \cdot \boldsymbol{i}) \int_{\lambda} f_c(\lambda) e(\lambda) b(\lambda) d\lambda + \int_{\lambda} f_c(\lambda) a(\lambda) k_a(\lambda) d\lambda + (\boldsymbol{r} \cdot \boldsymbol{v})^{\boldsymbol{\beta}} \int_{\lambda} f_c(\lambda) e(\lambda) s(\lambda) d\lambda. \quad (5.3)$$

The first two terms are the same as in Equation 5.1 and are view-independent. The third term is the specular reflection term (surface reflection) which is view-dependent. In this equation, $\boldsymbol{v}$ is the direction of the viewpoint and $\boldsymbol{r}$ is the direction of the reflection. The parameter $\boldsymbol{\beta}$ is a material shininess constant. The term s is the ratio of reflection of the specular term of incoming light. The parameters $\boldsymbol{\beta}$ and s are determined by the object materials property.

2) Conversion of Light Rays

After the light reaches the camera, the color signal goes through a non-linear gamma correction before it is stored as an integer on the imaging sensor. This non-linear inner camera processing is expressed as [57]:

$$I_i(\mathbf{p}) = \rho(\mathbf{p}) \alpha_i C_i(\mathbf{p})^{\gamma} \quad (i = R, G, B), \quad (5.4)$$

where $\rho(\mathbf{p})$ is the brightness factor of the pixel **p** and $\rho(\mathbf{p})$ is a factor affected by the surface normal and light directions. The parameters α_i are the scale factor for the Red, $Green$, and $Blue$ channels, respectively.

Therefore, in real-world scenes, due to the complex interactions among the illumination sources, objects' surfaces and color sensors, the colors of corresponding pixels are not always the same even for a uniform illumination energy. This explains why many stereo matching algorithms do not work well and there is a need for a true illumination invariant stereo matching algorithm.

5.2 Illumination Invariant Stereo Matching

Human eyes may not be able to perceive color variation of corresponding pixels when the illumination is uniform, as shown in Figure 1.17, but the differences become quite obvious if the images are from non-uniform illuminations. This implies that no corresponding pixels share the same intensity values. Similar to the regular stereo matching algorithms, illumination invariant stereo matching algorithms can be classified in local and global categories.

5.2.1 Local Radiometric Invariance Stereo Matching

Normalized Cross Correlation (NCC) [61] can handle the illumination variance but only works well if the change of illuminations follows a linear relation. Zero-mean Normalized Cross Correlation (ZNCC) [61] calculates the difference between a pixel and the central pixel, and uses this difference instead of the intensity in NCC. Therefore, ZNCC works better than NCC because ZNCC can get rid of the additive illumination variations as well as the multiplicative ones.

The *Rank* algorithm [148] uses Sum of Squared Differences (SSD) or Sum of Absolute Differences (SAD) to compare two windows. The elements in the windows are not intensities as they are replaced by their intensity ranks. The rank is actually the number of pixels in a local window whose intensities are less than this pixel and is expressed by:

$$R_{rank}(\mathbf{p}) = \sum_{\mathbf{q} \in W_{\mathbf{p}}} T[\mathbf{I}(\mathbf{q}) < \mathbf{I}(\mathbf{p})]. \tag{5.5}$$

The function $T[\,]$ is equal to 1 if the argument is true; otherwise it is equal to 0. The parameter $W_{\mathbf{p}}$ is the window size centered at pixel **p**. In this algorithm, the correspondence is established by the rank of intensity instead of the intensity itself. The assumption here is that if the lighting condition changes, the pixel intensities change as well, but the pixel order stays the same.

The *Census* algorithm [148] transforms a window into a bit string, where each bit corresponds to a pixel in the window. The bit is set to 1 if the intensity of this pixel is less than the central pixel; otherwise, the bit is set as 0. Then, the two bit strings are compared using the Hamming distance:

$$R_{\mathbf{p}}(\mathbf{i}) = \begin{cases} 1, T[\mathbf{I}(\mathbf{i}) < \mathbf{I}(\mathbf{p})] \\ 0, otherwise \end{cases} \tag{5.6}$$

Neither *Rank* nor *Census* algorithm relies on the pixel intensities directly or an explicit reflection model. As long as the illumination changes are monotonic, the rank of a pixel in a window stays the same. Thus, both *Rank* and *Census* algorithms are more reliable for matching images under different illumination conditions.

Mutual information (MI) was first applied to image processing applications in 1995, *i.e.* image registration [37], and to measure the similarity of medical images [129]. Point-based MI registration [105] was first published in 1999. Likar *et al.* [86] applied MI to non-rigid registration in 2001. MI has become one of the most popular methods for measuring the similarity between images or signals today.

Recall that the MI of two images $\mathbf{I_1}$ and $\mathbf{I_2}$ is introduced in Chapter 4, which is defined as [115]:

$$MI(\mathbf{I_1}, \mathbf{I_2}) = H(\mathbf{I_1}) + H(\mathbf{I_2}) - H(\mathbf{I_1}, \mathbf{I_2}), \tag{5.7}$$

where $H(\mathbf{I_1})$ and $H(\mathbf{I_2})$ are the marginal entropies of the images $\mathbf{I_1}$ and $\mathbf{I_2}$, while $H(\mathbf{I_1}, \mathbf{I_2})$ is their joint entropy. The MI function reaches the maximum if two images are exactly the same. Therefore, MI can be used as a matching cost function to measure the similarity between two windows [44]:

$$\mathbf{C}(i, j, d) = MI(W_L(i, j), W_R(i - d, j)), \tag{5.8}$$

where $W_L(i, j)$ is a local window in the left image centered at (i, j). Even though MI can tolerate illumination variations but does not solve problems with the image boundaries and texture-less regions which are common in local stereo matching. Zhang *et al.* [149] choose a supporting region on each window in the left and right images, and then apply a matching cost function NCC on the intersection of the two supporting regions. A voting-based scheme is then used to refine the initial disparity map.

5.2.2 Global Illumination Invariance Stereo Matching

Kim *et al.* [74] use the Taylor expansion to approximate the MI of two images as the pixel-wise sum of dissimilarities in a small neighborhood. In this scheme, MI is used as the data term in an energy optimization function based on Markov Random Field (MRF). Hirschmüller *et al.* [62] proposed a hierarchical calculation of MI and added a third term in the energy function as the penalty for the large disparity discontinuity.

Heo *et al.* [57] proposed an Adaptive Normalized Cross Correlation (ANCC) method to solve local variation and reach global optimization performance. ANCC method assumes that the reflection model is Lambertian and the color inconsistency is caused by different illunimation energies, different lighting geometry, and non-linear inner camera gamma correction. In 2009, Heo and his colleagues [58] used the same color transform [57] and utilized the space information by adding a Scale Invariant Feature Transform (SIFT) descriptor to construct the joint probability for the computation of MI as the data term. Based on this work [58], they proposed another Stereo Color Histogram Equalization (SCHE) image to update the SIFT descriptor and data cost in the energy function [59].

Besides of MI and MRF, Miled *et al.* [92] approximate the illumination changes as a linear model and solve the stereo matching in the framework of convex optimization.

Table 5.1 summarizes the typical illumination invariant stereo matching algorithms found in the literature. They usually include a color transform before the matching process begins or use a matching cost function with automatic elimination of the illumination variations. To date, MI-based

Table 5.1: Categories of illumination invariant stereo matching methods

Local	Global
NCC	Kim 03 [74]
ZNCC	Heo 08 [57]
Rank Zabih 94 [148]	Hirschmüller 05 [62]
Census Zabih 94 [148]	Heo 09 a [58]
Zhang 09 [149]	Heo 09 b[59]
	Miled 09 [92]

stereo matching algorithm [57, 58, 59] produce the best results. However, in Heo's scheme, the color transform uses the difference of each channel and the average of three channels, which limits the use of this method for color images. As with all global algorithms, they require very intensive processing and the convex optimization convergence time is unpredictable.

5.3 Hierarchical Stereo Matching Based on Improved Census Algorithm

Rank and *Census* algorithms are both local illumination invariant stereo matching methods. They work as long as the colors vary monotonically with the illuminations. In other words, *Rank* and *Census* algorithms perform well if the relative relations between the central pixel and the other pixels in the same local window are constant when the illuminations change. However, the same with other local methods, they fail at low textured regions and near the image boundaries. In this section, we focus our attention at improving the limitations of the *Census* algorithm.

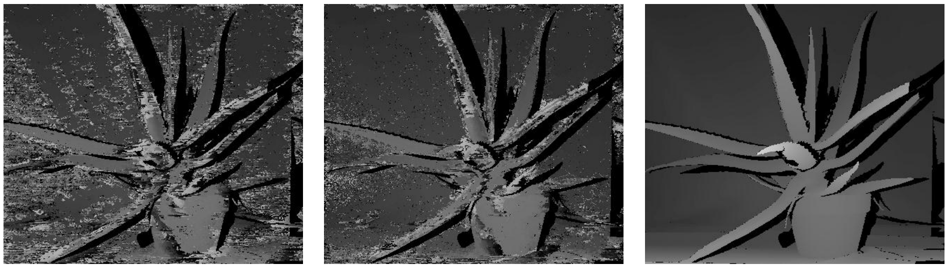
5.3.1 Improved Census Algorithm

The *Census* algorithm consists of two parts: binary conversion and string comparison using Hamming Distance. We perform the following tasks to on each part to improve the boundary problem.

1) Binary *vs.* 3-index color conversion

For local stereo matching, it is clear that the large window includes more details and thus leads to better results. However, the large window also includes pixels on different disparity planes (also called bad pixels). When the matching cost is accumulated, the assumption is that the contributions are all from the same disparity plane. If other pixels are involved, their ideal contribution should be zero.

The *Census* algorithm takes all the pixels in the local window into consideration, but those bad pixels prevent two windows from having a significant similarity even when they are centered at a pair of corresponding pixels. In order to get rid of the dependency on intensities and also reduce the impact of bad pixels, we suggest a 3-index color conversion scheme instead. First, to detect bad pixels, the color difference in the *RGB* color space is used to measure the distance of a pixel in a local window from the central pixel. The color difference $CD(\mathbf{p}, \mathbf{q})$ is calculated by:



(a) Hamming distance

(b) MI

(c) Ground truth

Figure 5.3: Similarity comparison: HD. *vs.* MI

$$CD(\mathbf{p},\mathbf{q}) = \sqrt{(R_{\mathbf{p}} - R_{\mathbf{q}})^2 + (G_{\mathbf{p}} - G_{\mathbf{q}})^2 + (B_{\mathbf{p}} - B_{\mathbf{q}})^2}, \tag{5.9}$$

where **p** is the central pixel in a window while **q** is any other pixel in the same window. The central pixel is selected as the seed. If $CD(\mathbf{p},\mathbf{q})$ is equal to or lower than a preset threshold, this pixel should be selected. The selected pixel is assumed to be on the same disparity plane as the central one. The set of selected pixels is marked by $\mathbf{S_p}$.

The index assignment is described by:

$$pixel(\mathbf{q}) = \begin{cases} 0, \text{if } \mathbf{I_{ref}}(\mathbf{q}) > \mathbf{I_{ref}}(\mathbf{p}) \text{ and } \mathbf{q} \in \mathbf{S_p} \\ 1, \text{if } \mathbf{I_{ref}}(\mathbf{q}) = \mathbf{I_{ref}}(\mathbf{p}) \text{ and } \mathbf{q} \in \mathbf{S_p} \\ 2, \text{if } \mathbf{I_{ref}}(\mathbf{q}) < \mathbf{I_{ref}}(\mathbf{p}) \text{ and } \mathbf{q} \in \mathbf{S_p}. \\ 1, \text{if } \mathbf{q} \notin S_{\mathbf{p}} \end{cases} \tag{5.10}$$

The function $\mathbf{I}(\mathbf{p})$ represents the color of **p**. The pixels in $\mathbf{S_p}$ are divided into 0, 1, or 2 categories according to whether their colors are greater, equal to, or less than the central pixel. If pixel **q** does not belong to $\mathbf{S_p}$, we assume that this pixel is from another disparity plane. Those pixels are marked as one as if they have the same color as the central pixel.

After the classification, the bad pixels are marked by the same index as the central pixel, as if they are on the same disparity plane. Hence, two corresponding local windows on boundaries will look similar because all bad pixels in both windows are indexed by the same number. Moreover, the 3-index color conversion algorithm assigns the same index to the corresponding pixels in differently illuminated images as long as the illumination change is monotonic.

2) Hamming Distance (HD) *vs.* Mutual Information (MI)

In addition, after the 3-index conversion, the *Census* algorithm transforms two windows into bit strings which are compared using the Hamming distance. However, the Hamming distance only does a simple count on how many pixels are different. We suggest using MI instead, which is a more accurate measurement of information distance, as it is based on entropy where corresponding pixels should have the maximal joint entropy. Figure 5.3 is an example, showing the results using HD and

MI to compare bit strings. Since we use the 3-index scheme to mark the pixels, a large window is preferred in order to include more samples to improve the histogram statistics in MI.

5.3.2 Holes filling: Interpolation vs. Image In-painting

After cross-checking [46], there are holes (pixels marked as errors in black) left on the disparity map. Some views of the 3D world can only be captured by one camera; therefore, those parts cannot find their corresponding pixels in other images. This is called occlusion and is one of the main reasons for holes. The other reason is low-textured areas there is not enough information to find the correct matches.

Global methods solve the problem by giving heavy penalties to disparity discontinuity. In local stereo matching, the disparities in the occlusions are usually replaced by the background disparities, or the disparities to the closest pixels, or interpolated from the neighbor pixels.

Equation 5.11 defines the process of interpolation. Let **k** be a pixel without a disparity value and let **p** be the pixel right before **k** and with a correct disparity value on the same scan line while **q** is the pixel right after **k** with a correct disparity value. The disparity value of **k** can be interpolation by:

$$D(\mathbf{k}) = D(\mathbf{p}) + \frac{||\mathbf{k} - \mathbf{p}||}{||\mathbf{q} - \mathbf{p}||} * (D(\mathbf{q}) - D(\mathbf{p})), \tag{5.11}$$

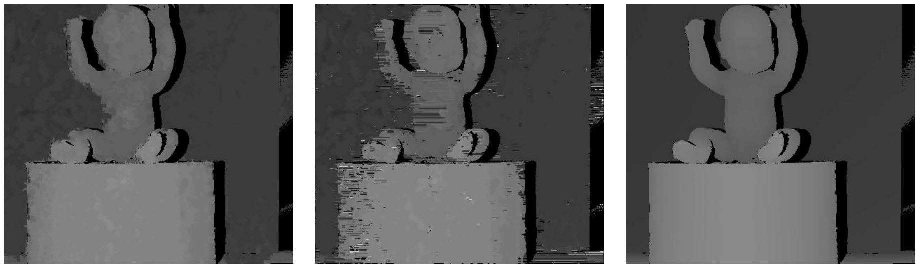
where $D(\mathbf{k})$ means the disparity value of pixel **k** and $||\mathbf{k} - \mathbf{p}||$ is the distance between pixel **k** and **p**. $||\mathbf{k} - \mathbf{p}||$ is actually $|\mathbf{k}.i - \mathbf{p}.i|$ since the three pixels have the same j value.

However, without precisely estimated 3D geometry, simple replacement or interpolation schemes create artifacts. In many ways, disparity map can be thought of as a kind of color texture where each disparity plane has its own texture properties. To avoid these problems, we use image in-painting techniques [41] to fill those holes (occlusions and mismatches). Let's define the areas with correct disparity values as $\mathbf{\Phi}$ and the areas to be filled as $\mathbf{\Omega}$. The border of $\mathbf{\Phi}$ and $\mathbf{\Omega}$ is represented by $\delta\mathbf{\Omega}$. The patch centered at pixel **p** ($\mathbf{p} \in \delta\mathbf{\Omega}$) is noted by $\mathbf{\Psi_p}$. For all patches centered at pixels in $\delta\mathbf{\Omega}$, we first compute the priority values. The patch $\mathbf{\Psi_k}$ with the maximum priority is chosen to find the patch $\mathbf{\Psi_g}$ in $\mathbf{\Phi}$ which has the minimum distance measured by SSD. The patch $\mathbf{\Psi_k}$ is then replaced by $\mathbf{\Psi_g}$ and the priority is updated.

Figure 5.4 is a comparison of holes filled by interpolating the neighbor pixels and the in-painting algorithm [41]. There are obvious streaks in Figure 5.4(b) caused by interpolation while the in-painted disparity map of Figure 5.4(a) is smoother and continuous.

5.3.3 Main Steps of the Proposed Method

Figure 5.5 shows the flowchart for the proposed method. The use of 3-index color conversion and MI does improve the quality of the disparity map, but it also degrades the requirement for real-time. By using a hierarchical structure, we will show that one can increase the processing speed and maintain



(a) In-painting

(b) Interpolation

(c) Ground truth

Figure 5.4: Holes filling by interpolation and in-painting

the real-time requirement. In this hierarchical scheme, the resolutions of the original images are down-sampled to half-size and quarter-size using a Gaussian pyramid. The first iteration starts from the lowest resolution. Then, the intermediate scale images are processed in the second iteration, and finally, in the third iteration, the full-size images are processed.

The procedures for each iteration are similar. We summarize the main steps here:

Step 1: Initialization of disparity maps. The disparity map from the previous iteration is used to initialize the disparity maps in the current iteration. Since the size of the image in the current iteration is four times as large as the one in the previous iteration, the disparity value of each pixel (m, n) is assigned to four pixels (i, j), if $(\lfloor i/2 \rfloor, \lfloor j/2 \rfloor) = (m, n)$. Notice that only those disparity values marked as correct by cross-checking [46] are used for initialization and the other pixels are left blank. The disparity map is initialized to zero in the first iteration.

Step 2: Stereo matching. The general window-based stereo matching is applied to calculate the disparities. The improved *Census* algorithm is used to find the best matches. In the first iteration, the disparities of all pixels are calculated. In the second and the third iterations, only those pixels marked as "error" after cross-checking are re-calculated.

Step 3: Filtering and cross-checking. Both disparity maps are smoothed with a median filter of size 5 in order to get rid of noises which are too bright or too dark. Pixels are marked as "correct" or "error" by cross-checking.

Step 4: Holes filling. Pixels without disparity values are filled using the image in-painting algorithm [41]. This step is only executed after the last iteration.

5.3.4 Experiments Results

We use the Middlebury database [4] to test our algorithm and compare its results to other local methods found in the literature. The results will be compared based on three criterions: visual, quantitative, and speed. We will demonstrate that our method produces some of the best results without a sharp reduction in processing speed.

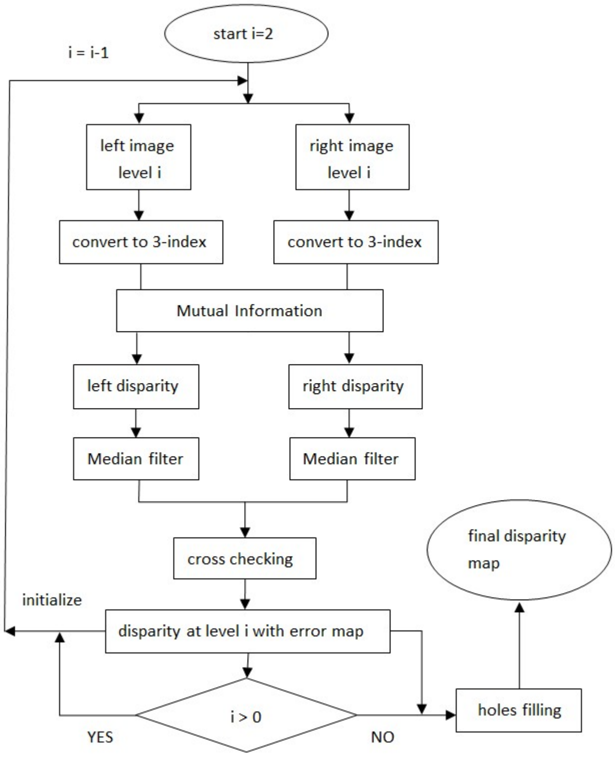


Figure 5.5: The flowchart for the proposed method

Comparison of Visual Quality

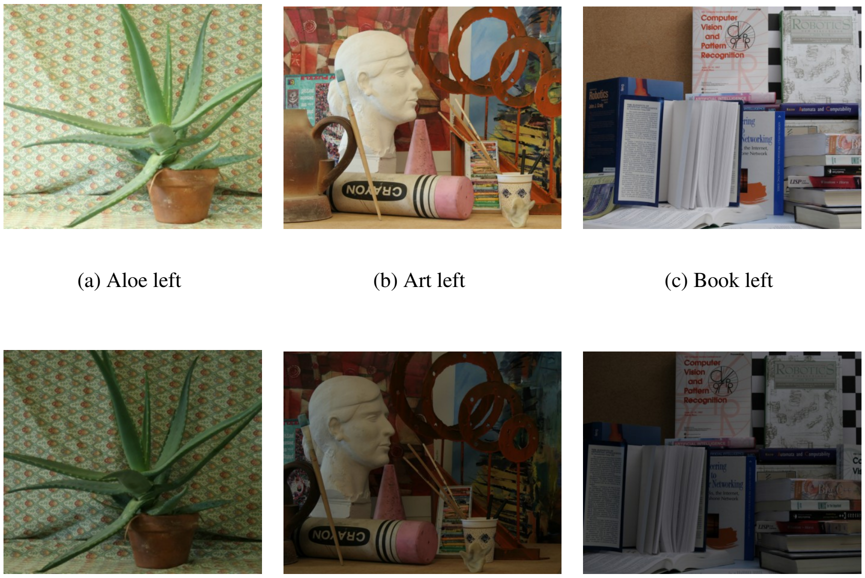
Seven local stereo matching algorithms are compared in this section: SAD, *Rank*, *Census*, NCC, ZNCC, MI [44], and the proposed algorithm. Figure 5.6 shows three test images in different illumination conditions. The right images are set as reference images. For all the algorithms used in the comparison, we set a window size of 25×25 pixels and use a Winner-Takes-All (WTA) strategy to choose the corresponding pixel. The results with the test images are illustrated in Figures 5.7, Figure 5.8, and Figure 5.9. The SAD algorithm totally relies on intensities and thus does not work at all. It is clear that our method is visually the best for all test images. NCC and ZNCC algorithms work better on the “Art” and “Book” images. The MI algorithm is the best for the “Aloe” images.

Quantitative Result Analysis

In order to prove the effectiveness of our method, we evaluate our results with more test cases quantitatively and compare with other algorithms by using the Mean Absolute Error (MAE) and error ratio. MAE [92] measures the mean absolute error between computed disparities and ground truth provided by the Middlebury database:

$$MAE = \frac{\sum_{i,j} |D_G(i,g) - D(i,j)|}{N}. \tag{5.12}$$

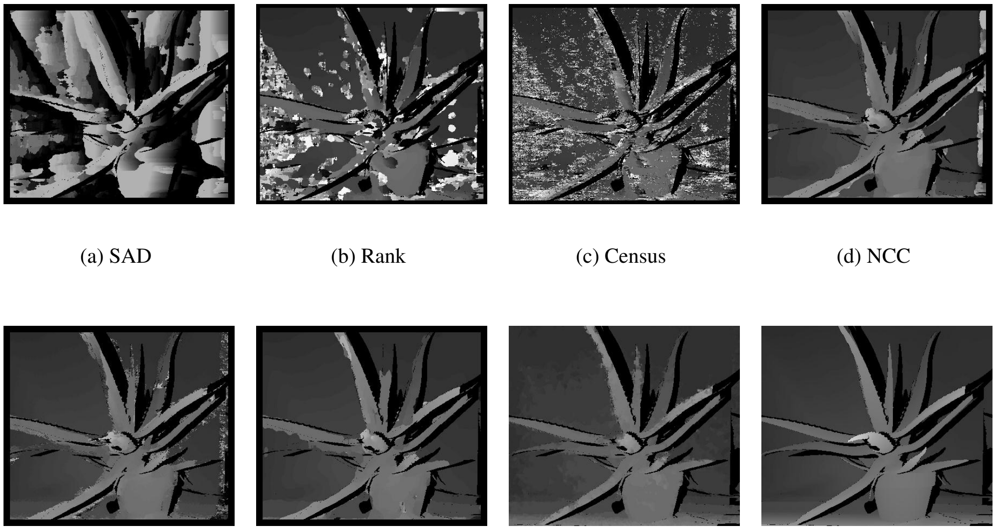
$D_G(i,j)$ is the ground truth disparity of (i,j), $D(i,j)$ is the computed disparity value, and N is



(a) Aloe left (b) Art left (c) Book left

(d) Aloe right (e) Art right (f) Book right

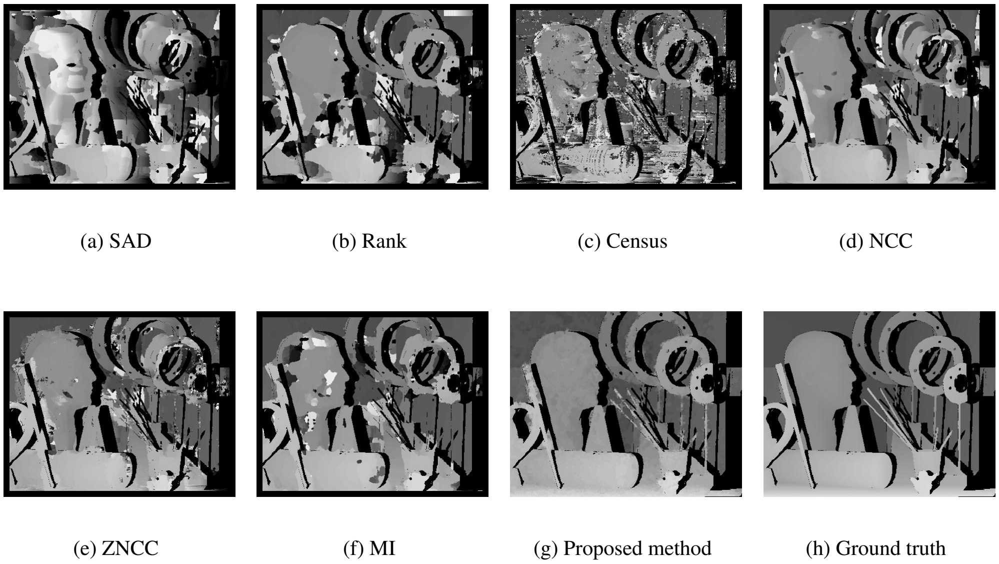
Figure 5.6: Test images with different illumination conditions (images from [4])



(a) SAD (b) Rank (c) Census (d) NCC

(e) ZNCC (f) MI (g) Proposed method (h) Ground truth

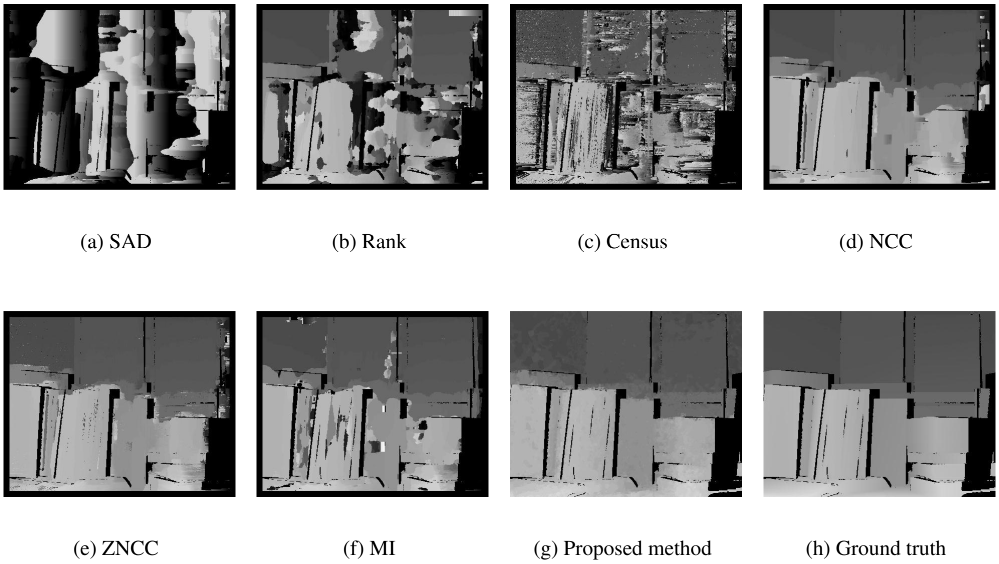
Figure 5.7: Result comparison of the “Aloe” image in different illuminations



(a) SAD (b) Rank (c) Census (d) NCC

(e) ZNCC (f) MI (g) Proposed method (h) Ground truth

Figure 5.8: Result comparison of the “Art” image in different illuminations



(a) SAD (b) Rank (c) Census (d) NCC

(e) ZNCC (f) MI (g) Proposed method (h) Ground truth

Figure 5.9: Result comparison of the “Book” image in different illuminations

Table 5.2: Quantitative analysis using MAE measure excluding the occlusion areas

	Aloe	Baby	Book	Wood	Art	Moebius	Cloth	Reindeer
SAD	18.98	22.97	25.95	24.54	11.53	11.48	17.87	14.65
Rank	9.67	9.79	13.13	3.25	9.97	15.53	30.07	7.05
Census	13.36	9.81	13.28	6.29	11.59	16.01	23.95	11.26
NCC	4.28	4.19	2.74	3.54	8.60	5.12	1.80	6.31
ZNCC	4.58	4.23	6.35	9.02	8.19	5.49	3.86	5.49
Mutual Information	2.16	3.67	3.70	11.31	7.29	3.75	0.44	9.23
proposed	**0.77**	**0.8**	**0.98**	**2.52**	**1.53**	**1.24**	**0.06**	**1.97**

Table 5.3: Error ratio on Middlebury

	Tsukuba			Venus			Teddy			Cones		
	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc
AdaptingBP	1.11	1.37	5.79	0.10	0.21	1.44	4.22	7.06	11.8	2.48	7.92	7.32
GeoSup	1.45	1.83	7.71	0.14	0.26	1.90	6.88	13.2	16.1	2.94	8.89	8.32
Proposed	**0.99**	**1.13**	**5.20**	**3.97**	**4.55**	**14.1**	**5.96**	**10.4**	**15.5**	**4.77**	**10.2**	**11.5**
AdaptWeight	1.38	1.85	6.90	0.71	1.19	6.13	7.88	13.3	18.6	3.97	9.79	8.26
GraphCut	1.94	4.12	9.39	1.79	3.44	8.75	16.5	25.0	24.9	7.70	18.2	15.3
DP	4.12	5.04	12.0	10.1	11.0	21.0	14.0	21.6	20.6	10.5	19.1	21.1

the number of pixels in the image. The quantitative results are listed in Table 5.2. Table 5.2 shows the proposed methods have the least error compared to ground truth. The MI algorithm is better than NCC and ZNCC except for the "Wood" and the "Reindeer" images. The NCC and ZNCC algorithms have similar error ratios and rank third. The *Rank* and *Census* algorithms rank fourth and the SAD algorithm does not work for all cases.

Table 5.3 shows the error ratio (the percentage of wrong disparities compared to the ground truth) of the proposed method compared to some of the classic methods on the Middlebury test bed [5]. The result of "Tsukuba" is the best of the four test images.

Speed Comparison

MI is computationally expensive and its preference for large windows makes it worse. Fortunately, our algorithm starts from quarter-size images; thus the search range is reduced significantly. For example, if the search range is set to 60 in the full-size image, it equals to 15 in the quarter-size image. Most correspondences can be found at low-resolution levels. Compared to the proposed method without hierarchical structure, the speed is increased on average by 20 times. All codes run on a laptop Thinkpad T400 with Intel (R) Core (TM) 2 Duo CPU, 2.4GHz and 2GB RAM. The processing times are listed in Table 5.4. Those numbers mean the processing time of a certain algorithm over the proposed method whose running time is supposed to be one unit.

Table 5.4: Processing time (times)

	Rank	Census	NCC	ZNCC	Mutual Information	proposed without hierarchical structure
proposed	**1.7**	**2.7**	**1.52**	**1.31**	**9.61**	**20.21**

5.4 Illumination Invariant Stereo Matching Based on Relative Gradients

As mentioned previously, the best three methods [59, 58, 57] for dealing with illumination variation only works for color images. However, in many applications, the images are grayscale. In addition, the three methods are all global and their complexity makes them difficult to be used for real-time applications.

Moreover, numerous papers published in the literature are based on the Lambertian assumption. In order to deal with non-Lambertian surfaces, we propose a new illumination invariant stereo matching algorithm based on the Relative Gradients (RG) [78, 65] which is an illumination invariant feature, and is able to deal with both color and gray scale images.

5.4.1 RG - The Illumination Invariant Feature

RG is defined as the intensity gradient normalized with the maximal local gradient. The RG of a pixel (i, j) can be expressed mathematically as:

$$RG(i,j) = \frac{gradient(i,j)}{max[gradient(i,j)]+1}, \tag{5.13}$$

where the $max[gradient(i, j)]$ is defined as the largest gradient in a 3-by-3 neighborhood centered at (i, j). The scalar “1” is added to the denominator in case the maximal gradient is close to zero.

The reflection model described by Equation 5.3 can be simplified as the combination of the view-independent color and view-dependent color as following:

$$\begin{aligned} I &= (\boldsymbol{n}\cdot\boldsymbol{i})\int_\lambda f_c(\lambda)e(\lambda)b(\lambda)d\lambda + \int_\lambda f_c(\lambda)a(\lambda)k_a(\lambda)d\lambda + (\boldsymbol{r}\cdot\boldsymbol{v})^{\boldsymbol{\beta}}\int_\lambda f_c(\lambda)e(\lambda)s(\lambda)d\lambda \\ &= \underbrace{(\boldsymbol{n}\cdot\boldsymbol{i})eb + ak}_{view\ independent} + \underbrace{(\boldsymbol{r}\cdot\boldsymbol{v})^{\boldsymbol{\beta}}es}_{view\ dependent} \quad . \end{aligned} \tag{5.14}$$

Assuming that the lighting geometries and the material properties are the same for pixels in a 3-by-3 neighborhood, through subtracting neighbor pixels by computing the local gradient, one can remove the variance to the term $(\boldsymbol{r}\cdot\boldsymbol{v})^{\boldsymbol{\beta}}es$ and ak. The influence of the illumination energy e term can be canceled using Equation 5.13. Equation 5.13 is the function of RG for one channel. For color images, the difference of corresponding pixels in color images is the sum of the difference of the RG for each channel. Since the transform of RG is independent for each pixel, stereo matching based on RG can be solved locally or globally. Speed is a concern in our applications, we implement the proposed method as a local stereo matching and the SAD function is used for matching cost.

5.4.2 Matching Cost Function of the Proposed Method

Similar to the other local stereo matching algorithms proposed, the matching cost is accumulated in two local windows. SAD or SSD function can be used to compute the RG difference between two pixels. Winner-Takes-All (WTA) is the strategy to decide the corresponding pixel. We use a Gaussian weighing function $G(\mathbf{q})$ to assign a weight to each pixel according to the color distance. This is based on the assumption that if a pixel is far away from the central pixel in color, it is probably from other disparity planes. In the reference (left) image, if the color of pixel **q** is close to the central pixel **p**, the Gaussian weighting function gives a high weight to the matching cost $C(\mathbf{q}, \mathbf{d})$; otherwise, pixel **q** contributes very little to the total matching cost $C(W_{\mathbf{p}})$. The weighting function is defined as:

$$C(W_{\mathbf{p}}) = \sum_{\mathbf{q} \in W_{\mathbf{p}}} G(\mathbf{q}) C(\mathbf{q}, \mathbf{d}) = \sum_{\mathbf{q} \in W_{\mathbf{p}}} \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(\mathbf{I_L}(\mathbf{q}) - \mathbf{I_L}(\mathbf{p}))^2}{2\sigma^2}) |RG_L(\mathbf{q}) - RG_R(\mathbf{q} - \mathbf{d})|. \tag{5.15}$$

5.4.3 Post Processing

In addition, we suggest a way to correct the inability to perform matching in low textured regions. Our method is based on the observation that disparities in the same disparity plane change smoothly and errors in these areas has a bias to be 255 or 0. Thus, the disparity of a pixel should be close to its neighbors and a more specific search range is useful to locate the correct correspondence quickly. Therefore, a second stereo matching is performed for each mismatched pixel detected by cross-checking [46] and a new search range is set to each of these pixels. The new search range is between the previous and next correct disparities on the same epipolar line:

$$searchrange(mismatch) \in [min(previous(dis), next(dis)), max(previous(dis), next(dis))]. \tag{5.16}$$

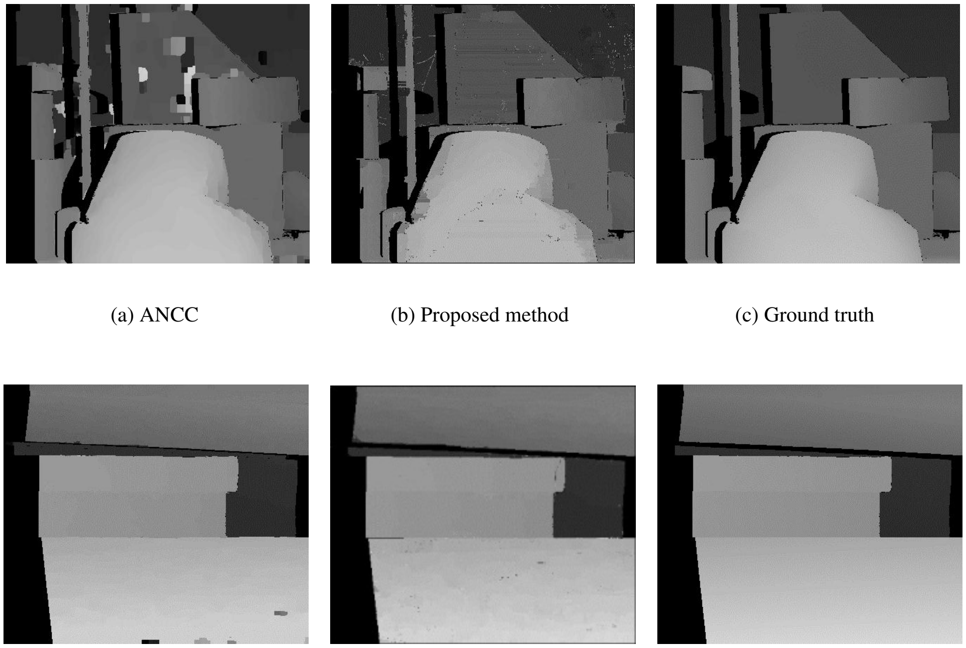
If the matching cost is still too large, the depth of this pixel could be filled by the image inpainting technique [41] or $min[(previous(dis), next(dis)]$.

5.4.4 Experimental Results

Visual comparison

First, we test the proposed method by comparing it with the ground truth visually. Disparity maps are grayscale images whose intensities represent depth information. The darker the pixel is, the further the object is from the viewer. The resulting disparities are compared with the ground truth pixel by pixel. Test images and ground truth images are all from the Middlebury dataset [4]. Figure 5.10 is a comparison between the proposed method and ANCC.

The two test images are “Lamp” and “Wood”. To solve the matching problem, ANCC uses global optimization, which is one of the best methods so far. Generally speaking, global methods should be better than local ones [60]. However, because of the improvements to the boundary and low texture areas, our local method can compete with ANCC.



(a) ANCC

(b) Proposed method

(c) Ground truth

(d) ANCC

(e) Proposed method

(f) Ground truth

Figure 5.10: Visual comparison of ANCC [57] with the proposed method

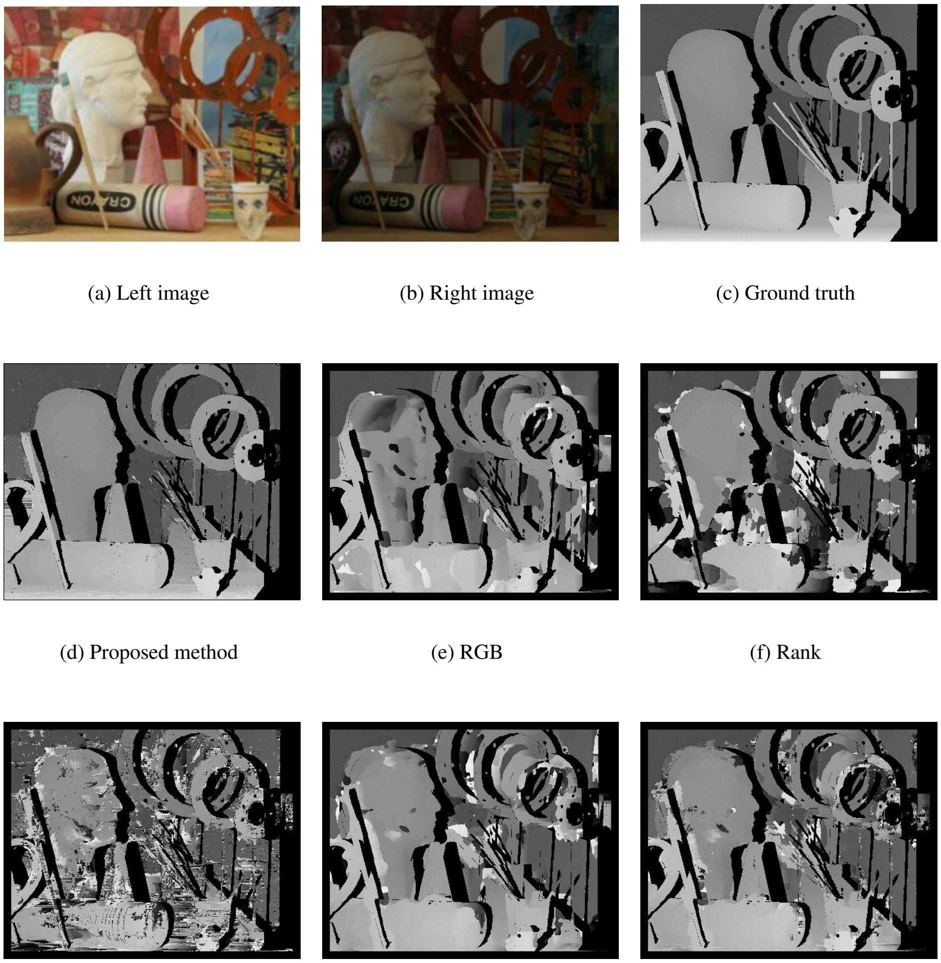
Figures 5.11 and 5.12 list the results of commonly used local methods for different illumination conditions. All the local methods use the WTA strategy to select the best match. The window size is 25×25 pixels. One can see that the boundaries in the proposed method are sharper than other methods and the disparities are good in the low texture areas.

Quantitative Comparison

To further demonstrate how well the algorithm works, we compare the results with other popular methods using various images. In Table 5.5. we summarize the quantitative analysis. The test images all have different illuminations. The error ratio is the percentage of the wrong disparities (excluding occlusions) over all pixels. One can see from Table 5.5 that the error ratios of our method

Table 5.5: Error ratio (%) comparison of local illumination invariant stereo matching methods(without occlusions)

	Art	Moebius	Wood	Rock	Book	Aloe	Baby
rgb	24.83	33.75	73.42	41.06	24.33	12.34	47.65
RANK	38.44	54.42	14.58	7.69	47.05	31.75	32.21
CENSUS	44.81	54.37	25.50	20.91	46.39	43.90	36.02
NCC	27.16	22.75	18.33	13.66	18.01	19.43	12.89
ZNCC	28.02	21.71	21.47	15.61	18.21	25.41	19.14
ANCC	7.15	7.95	3.94	6.46	11.09	3.12	4.75
proposed	**6.48**	**6.40**	**0.53**	**2.66**	**6.40**	**3.67**	**3.33**



(a) Left image

(b) Right image

(c) Ground truth

(d) Proposed method

(e) RGB

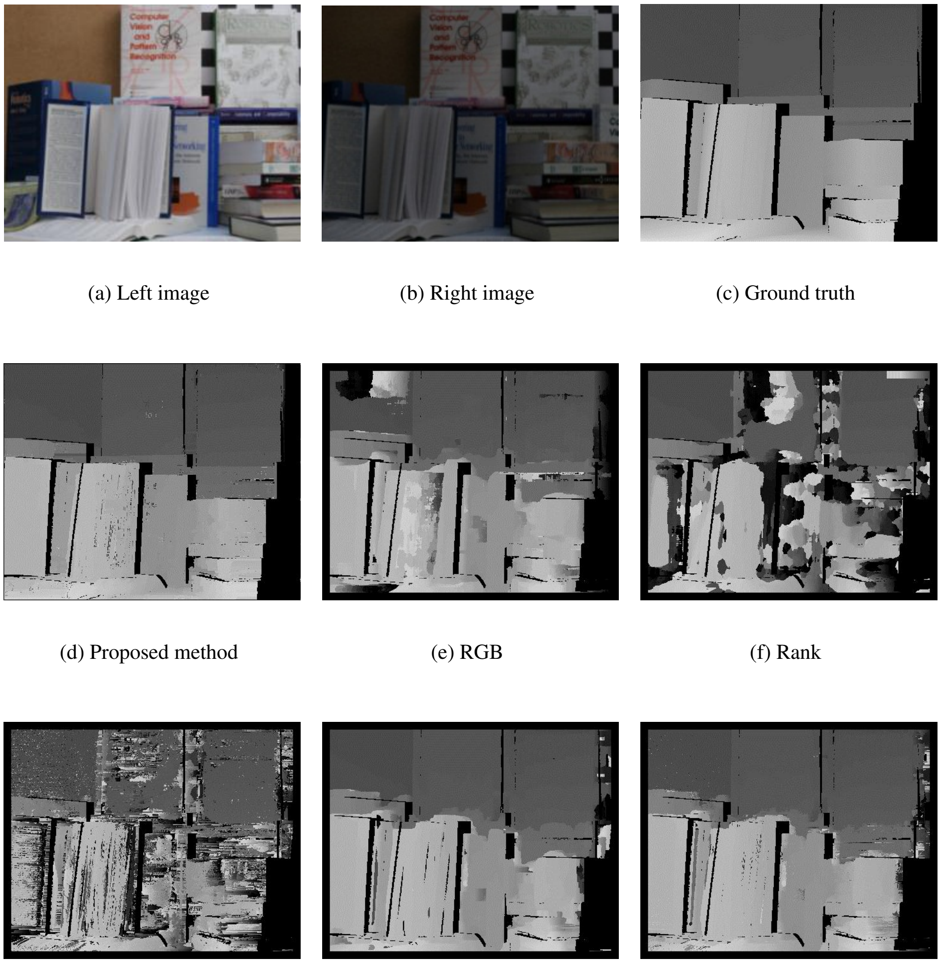
(f) Rank

(g) Census

(h) NCC

(i) ZNCC

Figure 5.11: The visual results of the “Art” image for different illuminations



(a) Left image

(b) Right image

(c) Ground truth

(d) Proposed method

(e) RGB

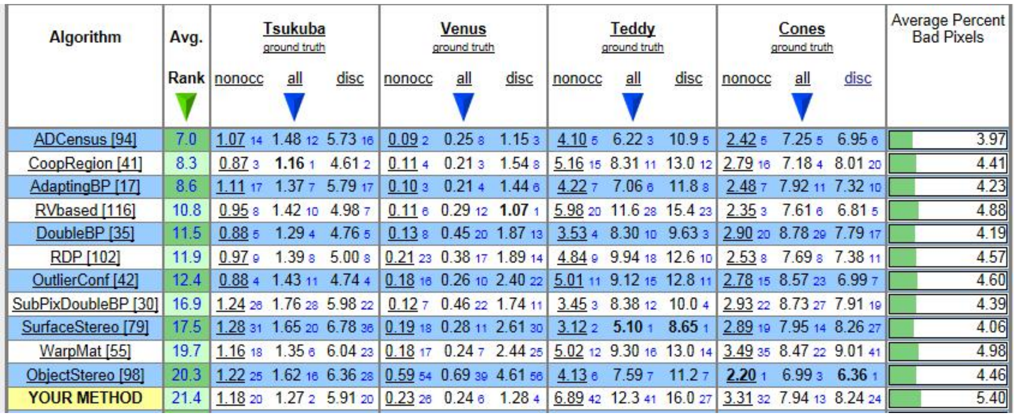
(f) Rank

(g) Census

(h) NCC

(i) ZNCC

Figure 5.12: The visual results for the “Book” image for different illuminations



Algorithm	Avg.	Tsukuba ground truth			Venus ground truth			Teddy ground truth			Cones ground truth			Average Percent Bad Pixels
	Rank	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	
ADCensus [94]	7.0	1.07 14	1.48 12	5.73 16	0.09 2	0.25 8	1.15 3	4.10 5	6.22 3	10.9 5	2.42 5	7.25 5	6.95 6	3.97
CoopRegion [41]	8.3	0.87 3	**1.16** 1	4.61 2	0.11 4	0.21 3	1.54 8	5.16 15	8.31 11	13.0 12	2.79 16	7.18 4	8.01 20	4.41
AdaptingBP [17]	8.6	1.11 17	1.37 7	5.79 17	0.10 3	0.21 4	1.44 6	4.22 7	7.06 6	11.8 8	2.48 7	7.92 11	7.32 10	4.23
RVbased [116]	10.8	0.95 8	1.42 10	4.98 7	0.11 6	0.29 12	**1.07** 1	5.98 20	11.6 28	15.4 23	2.35 3	7.61 6	6.81 5	4.88
DoubleBP [35]	11.5	0.88 5	1.29 4	4.76 5	0.13 8	0.45 20	1.87 13	3.53 4	8.30 10	9.63 3	2.90 20	8.78 29	7.79 17	4.19
RDP [102]	11.9	0.97 9	1.39 8	5.00 8	0.21 23	0.38 17	1.89 14	4.84 9	9.94 18	12.6 10	2.53 8	7.69 8	7.38 11	4.57
OutlierConf [42]	12.4	0.88 4	1.43 11	4.74 4	0.18 16	0.26 10	2.40 22	5.01 11	9.12 15	12.8 11	2.78 15	8.57 23	6.99 7	4.60
SubPixDoubleBP [30]	16.9	1.24 26	1.76 28	5.98 22	0.12 7	0.46 22	1.74 11	3.45 3	8.38 12	10.0 4	2.93 22	8.73 27	7.91 19	4.39
SurfaceStereo [79]	17.5	1.28 31	1.65 20	6.78 36	0.19 18	0.28 11	2.61 30	3.12 2	**5.10** 1	**8.65** 1	2.89 19	7.95 14	8.26 27	4.06
WarpMat [55]	19.7	1.16 18	1.35 6	6.04 23	0.18 17	0.24 7	2.44 25	5.02 12	9.30 16	13.0 14	3.49 35	8.47 22	9.01 41	4.98
ObjectStereo [98]	20.3	1.22 25	1.62 16	6.36 28	0.59 54	0.69 39	4.61 56	4.13 6	7.59 7	11.2 7	**2.20** 1	6.99 3	**6.36** 1	4.46
YOUR METHOD	21.4	1.18 20	1.27 2	5.91 20	0.23 26	0.24 6	1.28 4	6.89 42	12.3 41	16.0 27	3.31 32	7.94 13	8.24 24	5.40

Figure 5.13: The Middlebury rank of the proposed method

are much smaller than others for all test cases. Figure 5.13 is the evaluation on the Middlebury test bed [5]. Our proposed method ranks 12th out of 111 methods. The four test images "Tsukuba", "Venus", "Teddy", and "Cones" are in the same illumination condition.

5.5 Real-time Implementation of the RG Algorithm Using GPU

Compute Unified Device Architecture (CUDA) [9], developed by the NVIDIA Company, is a parallel structure on a GPU for general purpose parallel computation. Users communicate with the GPU using a C-like language. A group of threads executes one kernel function simultaneously. Threads are divided into blocks. Each block has its own shared memory.

Figure 5.15 describes five kinds of memory CUDA provides with different access speeds. Each thread has a register and each block has a shared memory. These two on-chip memories are accessed with very low latency. Global memory, constant memory, and texture memory are off-chip memories and have a lower access speed. Constant and texture memories are read-only.

CUDA is suitable to speed up our method because all the pixels are processed using the same instruction. In this section, we will describe the CUDA implementation of our proposed methods. The method we propose in Section 5.3 is in the category of local stereo matching, but the computation of the joint histogram in mutual information algorithm is not in parallel structure, which makes it too slow to be accomplished by a single GPU with the current computation ability. A paralleled version of the joint histogram or a more powerful GPU cluster may solve the speed problem. The second proposal is much lighter in computation. The calculation of RG and new search range are both independent for each pixel. Figure 5.16 shows how the CPU and GPU work together to finish the work which was totally on CPU.

A straightforward idea is to copy the whole image into the global memory. Each thread then accesses the global memory to fetch the pixel information. Unfortunately, it cannot reach real-time performance because of the image size and the two-round stereo matching algorithms. However, we

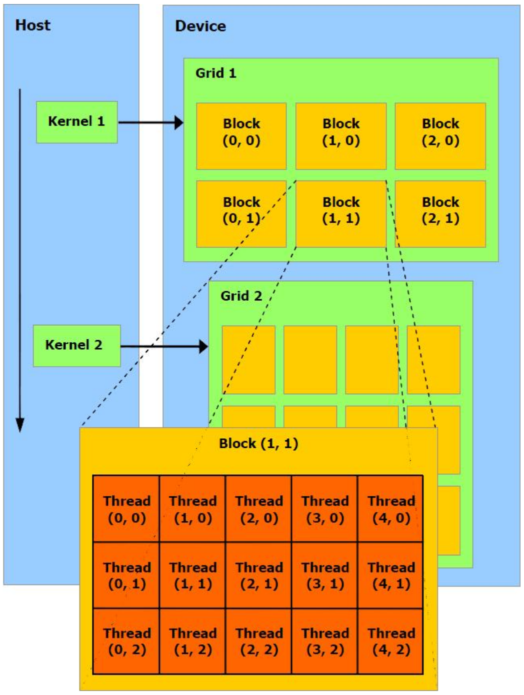


Figure 5.14: Thread batching [12]

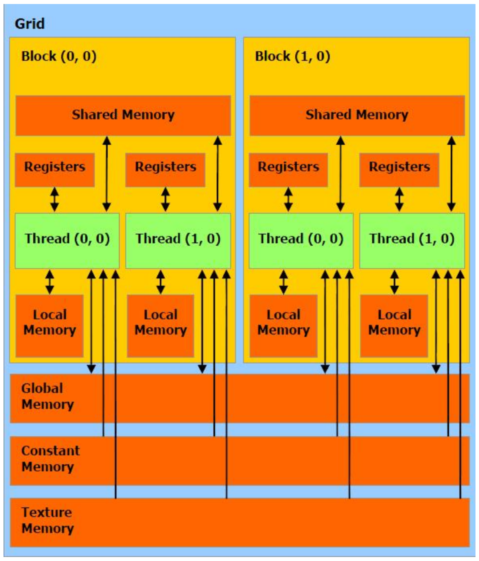


Figure 5.15: Memory model in CUDA [12]

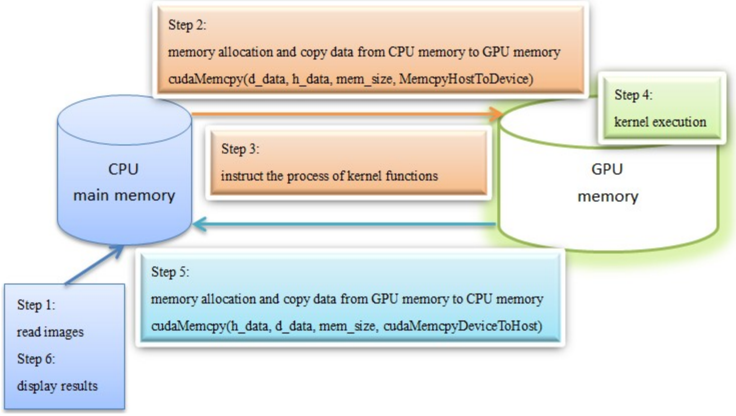

Step 2:
memory allocation and copy data from CPU memory to GPU memory
cudaMemcpy(d_data, h_data, mem_size, MemcpyHostToDevice)
Step 4:
kernel execution
CPU
main memory
Step 3:
instruct the process of kernel functions
GPU
memory
Step 5:
memory allocation and copy data from GPU memory to CPU memory
cudaMemcpy(h_data, d_data, mem_size, cudaMemcpyDeviceToHost)
Step 1:
read images
Step 6:
display results

Figure 5.16: Data processing between CPU and GPU

Table 5.6: Processing time of proposed method compared to GPU implementation (unit: second)

		Art	Moebius	Wood	Book	Rock	Aloe	Baby	Tsukuba	Venus	Teddy	Cones
disparity range		80	80	80	80	60	60	60	60	60	60	60
image seize	width	463	463	435	463	425	427	413	384	434	450	450
	height	370	370	370	370	370	370	370	288	383	375	375
GPU		0.046	0.049	0.037	0.045	0.036	0.040	0.038	0.035	0.036	0.042	0.038
fps		22	20	27	22	28	25	26	28	28	24	26

find that the threads in the same block only access a part of the image so that we can use the same way introduced in [100]. We just copy the part of image which could be shared by threads in the same block into the shared memory of that block, the data transmission time would be significantly decreased .

Table 5.6 shows the processing time for our GPU implementation compared to normal CPU implementation. The CPU version code runs on Intel (R) Core (TM) 2 Extreme CPU @ 3.20GHz with 4GB RAM. The GPU version code runs on the NVIDIA GeForce 9800 GX2. The window sizes are set as 25×25 pixels. All the test images can reach real-time performance.

5.6 Conclusion

This chapter introduces two new illumination invariant stereo matching algorithms. In the first contribution, a modified version of the *Census* algorithm is proposed. The method compares favorably to the Middlebury database and is truly invariant to illumination conditions. In addition, the hierarchical structure proposed in this section significantly accelerates the speed lowered by the estimation of the MI. The second algorithm is based on RG and is also capable of dealing with illumination variations and especially for the non-Lambertian surfaces. By using a Gaussian weighting function, an improved search range, and an image in-painting algorithm, we are able to improve disparity estimation at the boundaries and at low-textured areas as good as global methods. The experiments demonstrate its effectiveness and robustness in both visual and quantitative aspects. Because of its parallel structure, the real-time performance is demonstrated achievable using GPU implementation.

In many ways, the results show that the next generation of tele-presence systems may be based on passive stereo matching instead of the active stereo.

Chapter 6

Conclusion

6.1 Conclusion

Tele-presence systems are currently the high-end communication tools used in business enterprises today. As mentioned in Chapter 1, existing systems still suffer from poor eye contact capability as cameras are fixed and cannot be adjusted to an individual participant's gazing direction. The cost of these systems is also an issue. The requirements for creating the illusion of a common meeting room are expensive and not really practical on the long run. In order to solve some of those issues, we have focused our attention on three main problems:

- Real-time correction of eye contact;
- Segmenting participants from their backgrounds with sufficient quality to give the illusion that the participants are in a common virtual meeting room;
- Developing real-time stereo matching algorithms that can deal with different illumination conditions found in a tele-presence room.

Real-time automatic correction of eye contact: In many commercial systems, the necessary hardware used for correcting the eye contact is expensive and not easily accessible, which makes tele-presence an expensive technology to use. In addition, some hardware for tele-presence systems usually needs a room-sized space to set-up, which is not compatible with current trends in personal device market. This creates a strict set of requirements for the design of modern communication systems as the hardware must be affordable and portable, and the algorithms used must be real-time.

In this thesis, I have tried to design an eye contact correction solution for a tele-presence system that meets these requirements. Our solution includes a large monitor and at least two camera units. In each camera unit, there are two Infra-Red (IR) cameras, one IR projector, and one color camera. The two IR cameras are responsible for generating the initial disparity map of the scene. In order to enhance the accuracy and density of the initial disparity map, a random dot pattern is projected using IR light onto the scene to add artificial textures to low-textured areas, such as the skin and hair. The scene with random dots are detected using IR cameras and color camera is used to record the scene's real color.

The proposed solution is composed of eight main steps: After setting all cameras, the initial disparity map is created from two IR images and aligned with the color image. The 3D world coordinates of the pixels can be calculated by the initial disparity map and the calibration parameters. Then, the foreground is extracted from the video frames by disparity information and the disparity map of the foreground is interpolated. After projecting the 3D points onto two color images, each pixel in the color image can then find the position of its projection in the other color image. The shifts of all pairs of corresponding pixels in X-axis of two color images are recorded as the disparity map of the color images. Both the color image and its disparity map at each side are encoded and transmitted to remote participants. The receivers decode the color images and the disparity maps and then re-project them at the desired screen location. The final virtual view with correct eye contact is computed by blending corrected views from the left and right side. The foregrounds with correct eye contact are then placed in the same virtual conference room. If the monitor is large enough, the virtual conference room can be life-size.

The experimental results presented in this thesis have demonstrated that our solution to eye contact correction problem can produce a high quality virtual viewpoint. This system also gives end users the freedom to choose their own eye contact with whom they are talking instead of simply watching the corrected views imposed by the system designers. The virtual viewpoints can be adjusted at any time during the meeting using a mouse or keyboard. Furthermore, the proposed algorithm can address the large baseline problem caused by having cameras installed on the large displays commonly used in commercial tele-presence systems. The real-time performance is also one of the top priorities for tele-presence and is also one of the challenging parts of the software-based solutions. In the proposed solution, since only two camera units are used, the real-time performance can be supported by a single Graphics Processing Unit (GPU). Even the GPU implementation can significantly decrease the running time, there is still a limitation on the speed-up. More efficient algorithms and implementations on GPU or even GPU clusters need to be explored when the number of camera unit increases.

Foreground and background separation: Besides a solution to eye contact correction problem, we also work on some techniques to improve the immersive effect by separating participants from their backgrounds and posting them into a common virtual meeting room. Initially, disparity (depth) -based segmentation is first utilized in our system to extract the foreground. However, we notice that some parts of the hair and face are always cut off as the colors are much closer to the background. The same problem occurs on the boundary areas as well. To solve this problem, we propose a novel image matting method. Image matting, also called soft segmentation, analyzes the color of each pixel and results in a percentage describing how the color is mixed from the foreground and background colors. Existing image matting algorithms can be classified into two main categories:

sampled-based and propagation-based, and our proposed method takes advantage of the two categories. We first estimate the foreground and background colors for pixels by a simplified color estimation method in the robust matting [134]. Then, a mutual-information-based global optimization is applied to the energy function to generate the final matte under the constraint of a smoothness term. Mutual information is used in the data term and estimates the sum of local errors of each pixel in the image [74]. We test the proposed method by using eight test cases, all of which contain scenes with hairs. Each test case is performed with three levels of trimaps, from fine to coarse. The results show that the proposed method can produce good quality matte and hence improving the quality of the immersive experience. The error ratio increases very slowly when the trimaps become coarse. However, matting algorithms are computationally expensive and real-time image matting is still a challenging research topic. Most computation tasks lay on the steps of color estimation and global optimization. More efficient global/parallel color sampling or index algorithms have to be explored as well as the use of GPU.

Illumination invariant stereo matching: Since the key to depth-based view interpolation is the disparity map, we also try to improve robust passive stereo matching methods and find a solution to the illumination invariance problem. Because of the uneven lighting environment inside a typical tele-presence room, the images taken by the cameras at different locations will measure differently in color intensities. Even in the same illumination environment, a non-Lambertian reflection can cause differences in color. There are also many other reasons for color invariance, *i.e.*, lighting geometry, non-linear gamma correction, camera setting, etc.

In order to overcome these problems, we propose two illumination invariant stereo matching methods. Our first attempt is based on an improved *Census* algorithm. Three indexes are assigned to decrease the negative effect caused by pixels from other disparity planes. In our new implementation, mutual information replaces the Hamming distance to compare the window similarity and a hierarchical structure is used to decrease the computation load. This method is designed for general illumination changes as long as the changes are monotonous. Mutual information is very accurate at measuring the global similarity, but its unparallel structure makes it hard to be implemented in real-time. Our second method aims at dealing with non-Lambertian surface reflection. This model takes into consideration the diffuse, ambient, and specular reflection. Relative gradients are employed to eliminate the variance between the corresponding pixels while keeping a parallel computational structure allowing for real-time implementation. Taking advantage of the parallel structure of local stereo matching, a real-time GPU implementation is developed using CUDA. Post-processing of the disparity map is also important because of a lack of disparities in occluded regions. Mismatches caused by the boundaries and low-textured areas also need to be updated. In this thesis, we propose using limited search range and image in-painting. Limited search range shrinks the search range to avoid the white and black noises. Image in-painting [41] treats the disparities as textures and fills holes by searching the most similar patches in the disparity map.

Although the two proposed illumination invariant stereo matching methods are both classified as local stereo matching methods, their results are comparable to global matching methods. The proposed algorithms could also be applied to images taken in the same illumination. In particular, the relative-gradients-based method ranks 12th on the Middlebury test bed. The low-texture-area and the boundary problems are the bottleneck of the local stereo matching methods; besides, the speed is another concern.

6.2 Future Work

We are upgrading the cameras to High- Definition (HD) color cameras from the HP Company. We are also thinking about using Lytro light field cameras [3] in the future. Light field cameras record a scene's light field. Users are allowed to select a desired focus afterwards. To improve tele-presence systems, numerous advancements are necessary, such as: camera relocation, real-time image matting and stereo matching, and automatic camera selection.

Camera relocation: In the proposed solution, each camera unit has two IR cameras and one color camera. Each of the three camera captures the scene from a different viewpoint. Since the disparity map is generated by two IR images, new occlusions occur when the disparity map is aligned with the color images. In the future, we would like to relocate the color camera such that the optical axes of the IR cameras are co-aligned with the color camera axis, using an optical prism as in Wu's systems [138]. This solution would solve many problems and improve the image quality. A US patent has been filed for this concept by Dr. Boulanger.

Real-time image matting and stereo matching: Although the matting method we propose can produce good results, the speed cannot meet the system requirement. Moreover, the advent of HD cameras makes real-time computation for image matting and stereo matching even more challenging. Two things can be done to speed up our algorithms: the color samples can be reduced and a fast location algorithm used for the true colors. In addition, global stereo optimization methods can be modified to take the advantage of the GPU. We are now in the process of building an eight-GPU cluster in the AMMI laboratory. Using the computational power of this new cluster, it is our hope that all the processing steps involved in this systems can be performed in real time.

Automatic camera selection: In our experiments, only two camera units are in use. In the future, we will add more units to the system to reduce occlusions and improve image quality. Not all the units are used to create the disparity map, we only choose those units close to the eye gaze direction. Previous work relies on an eye tracker or head tracker to locate this direction, but we think this could be solved by software methods. Some features can be detected to find the closest cameras around, for example the distance of pupils, or the largest distance of the left and right sides of the face. The distance is the largest if the face is just in front of the camera and should be shorter in the camera capturing the side face.

Once, tele-presence systems were only considered in science-fiction movies. Nowadays, many companies are making tele-presence systems their top choice for communication devices. It not only saves considerably on traveling cost and time, it also increases the frequency of communications between business partners. Tele-presence is still on the way to becoming a powerful tool. Here are some trends one can expect in the near future:

Price and scale: The tele-presence system, because of its high price today, only large companies can afford to use it. However, tele-presence systems have the potential to be integrated with a wide range of applications, from large-scale multi-party group meetings to personal chats on mobility devices, i.e., laptops, tablets, and cell phones. The price has to be lowered to an affordable level to make tele-presence systems accessible for personal use. Hence, some functions performed by large, expensive hardware will be realized by low-cost software-based solutions. However, software-based solutions put a lot of pressure on the computational capacity of personal devices which one hopes should improve with time.

Functions: Currently, immersive tele-presence systems capture eye communication and body gestures, but the technology is still at an early stage. More research is required. The seamless meeting environment is one of the goals we are pursuing in the development of the immersive tele-presence. This virtual environment can be the reconstructed meeting room where the remote participants meet. Local participants can share the same floors, walls and even furniture with the remote participants so that they cannot feel the boundary between remote participants and themselves. It looks like they are in the same space. In addition, users have the possibility of interacting with the remote participants, exchanging handshakes and hugs. With the reconstruction of the 3D room, users can even navigate in the remote room and interact with the objects in that room. They can grab a chair, pick up a book, or open a drawer. Necessary sensors will be installed on these objects to receive signals and respond to the interactions.

Not long ago, this technology seemed to be something out of the space age, far off in the future. However, it is fast becoming a reality, and it is our hope that this thesis is bringing it one step closer to that day.

Bibliography

[1] http://en.wikipedia.org/wiki/principles_of_grouping.

[2] http://speckdesign.com/projects/view/telepresence_system_business_unit/showcase.

[3] https://www.lytro.com/.

[4] http://vision.middlebury.edu/stereo/datata/.

[5] http://vision.middlebury.edu/stereo/eval/.

[6] http://www.3dmd.com/.

[7] http://www.bradreese.com/cisco-it-telepresence-case-study.pdf.

[8] http://www.juew.org/data/data.htm.

[9] http://www.nvidia.ca/object/cuda_home_new.html.

[10] http://www.scholarpedia.org/article/radial_basis_function#eq-4.

[11] http://www.telepresence.co.uk/brands/cisco-telepresence/cisco-immersive-solutions/cisco-telepresence-system-3000/).

[12] Nvidia cuda compute unified device architecture programming guide. Technical report, NVIDIA Corporation, 2007.

[13] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994.

[14] E. H. Adelson and J. R. Bergen. The plenoptic function and the elements of early vision. *Computational Models of Visual Processing*, pages 3–20, 1991.

[15] F. Tombari amd S. Mattoccia, L. Di Stefano, and E. Addimanda. Near real-time stereo based on effective cost aggregation. In *Proceedings of International Conference on Pattern Recognition*, pages 1–4, 2008.

[16] Y. Andrèbe, R. Behn, B. P. Duval, P. Etienne, and A. Pitzschke. Use of webcams as tools for alignment and supervision of a thomson scattering system in the near infrared. *Fusion Engineering and Design*, 86(6-8):1273–1276, 2011.

[17] T. Aoki, K. Widoyo, N. Sakamoto, K. Suzuki, T. Saburi, and H. Yasuda. Monjunochie system: Videoconference system with eye contact for decision making. In *Proceedings of the International workshop on Advanced Image Technology*, 1999.

[18] A. D. Araujo, A. D. D. Neto, and A. M. Martins. Stereo map surface calculus optimization using radial basis functions neural network interpolation. In *Proceedings of the International Conference on Neural Information Processing*, pages 229–236, 2009.

[19] S. Avidan and A. Shashua. Novel view synthesis by cascading trilinear tensors. *IEEE Transactions on Visualization and Computer Graphics*, 4(4):293–306, 1998.

[20] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *Proceedings of the International International Conference on Computer Vision*, pages 1–8, 2007.

[21] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In *Proceesings of the International Joint Conference on Artificial Intelligence*, volume 2, pages 631–636, 1981.

[22] H. H. Baker, D. Tanguay, I. Sobel, D. Gelb, M. E. Goss, W. B. Culbertson, and T. Malzbender. The coliseum immersive teleconferencing system. In *Proceedings of the International Workshop on Immersive Telepresence*, 2002.

[23] A. Barbu and S. C. Zhu. Generalizing swendsen-wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1239–1253, 2005.

[24] Y. Boykov. Fast approximate energy minimization via graph cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[25] R. Brockers. Cooperative stereo matching with color-based adaptive local support. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns*, pages 1019–1027, 2009.

[26] M. D. Buhmann. *Radial basis functions: theory and implementations*. Cambridge University Press, 2003.

[27] W. Buxton. Telepresence: integrating shared task and person spaces. In *Proceedings of the Canadian Conference on Graphics Interface*, pages 123–129, 1992.

[28] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 67–76, 2001.

[29] J. C. Carr and R. K. Beatson W. R. Fright. Surface interpolation with radial basis functions for medical imaging. *IEEE Transactions on Medical Imaging*, 16(1):96–107, 1997.

[30] T. J. Cham, S. Krishnamoorthy, and M. Jones. Analogous view transfer for gaze correction in video sequences. In *Proceeding of the International Conference on Automation, Robotics, Control and Vision*, pages 1415–1420, 2002.

[31] S. Chen and L. Williams. View interpolation for image synthesis. In *Proceedings of the International conference on Computer Graphics and Interactive Techniques*, pages 279–288, 1993.

[32] S. E. Chen. Quicktime VR - an image-based approach to virtual environment navigation. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 29–38, 1995.

[33] W. C. Chen, H. Towles, L. Nyland, G. Welch, and H. Fuchs. Towards a compelling sensation of telepresence: demonstrating a portal to a distant (static) office. In *Proceedings of IEEE Conference on Visualization*, pages 327–333, 2000.

[34] G. Cheung, T. Kanade, J. Bouguet, and M. Holler. A real time system for robust 3-d voxel reconstruction of human motions. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 714–720, 2000.

[35] B. Choi, T. Kim, K. J. Oh, Y. S. Ho, and J. S. Choi. Intermediate view synthesis algorithm using mesh clustering for rectangular multiview camera system. *Optical Engineering*, 49(2):027002(1–8), 2010.

[36] Y. Y. Chuang, B. Curless, D. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 264–271, 2001.

[37] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens, and G. marchal. Automated multi-modality image rregistration based on information theory. In *Proceedings of the International Conference Information in Medical Imaging*, pages 263–274, 1995.

[38] D. Comaniciu and P. Meer. Robust analysis of feature spaces: color image segmentation. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 750–755, 1997.

[39] Powerworld Corporation. Knockout user guide, 2002.

[40] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs. A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3):542–568, 1996.

[41] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 721–728, 2003.

[42] A. Criminisi, J. Shotton, A. Blake, and P. H. S. Torr. Gaze manipulation for one-to-one teleconferencing. In *Proceedings of the International Conference on Computer Vision*, pages 191–198, 2003.

[43] F. Crow. Summed-area tables for texture mapping. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 207–212, 1984.

[44] G. Egnal. Mutual information as a stereo corresponding measure. Technical report, University of Pennsylvania, 2000.

[45] A. Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. In *Proceedings of the European Conference on Computer Vision*, pages 751–767, 2000.

[46] P. Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision and Applications*, 6(1):35–49, 1993.

[47] E. S. L. Gastal and M. M. Oliveira. Shared sampling for real-time alpha matting. *Computer Graphics Forum*, 29(2):575–584, 2010.

[48] J. Gemmell. Gaze awareness for video-conferencing: a software approach. *IEEE MultiMedia*, 7(4):26–35, 2000.

[49] S. J. Gibbs, C. Arapis, and C. J. Breiteneder. Teleport - towards immersive co-presence. *Multimedia Systems*, 7(4):214–221, 1999.

[50] S.J. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The lumigraph. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 43–54, 1996.

[51] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann. Random walks for interactive alpha-matting. In *Proceedings of the International Conference on Visualization, Imaging and Image Processing*, pages 423–429, 2005.

[52] W. E. L. Grimson. *From images to surfaces: a computational study of the early human visual system*. MIT Press, Cambridge, 1981.

[53] Y. Guan, W. Chen, X. Liang, Z. Ding, and Q. Peng. Easy matting: a stroke based approach for continuous image matting. In *Proceedings of Eurographics*, pages 567–576, 2006.

[54] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of vronoi dagrams. *ACM Transactions on Graphics*, 4(2):74–123, 1985.

[55] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[56] K. He, C. Rhemann, C. Rother, X. Tang, and J. Sun. A global sampling method for alpha aatting. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 2049–2056, 2011.

[57] Y. Heo, K. Lee, and S. Lee. Illumination and camera invariant stereo matching. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[58] Y. Heo, K. Lee, and S. Lee. Mutual information-based stereo matching combined with SIFT descriptor in log-chromaticity color space. In *Proceedings of the International Conference on Pattern Recognition*, pages 445–452, 2009.

[59] Y. Heo, K. Lee, and S. Lee. Simultaneous color consistency and depth map estimation for radiometrically varying stereo images. In *Proceedings of the International Conference on Computer Vision*, pages 1771–1778, 2009.

[60] H. Hirschmüller and D. Scharstein. Evaluation of cost functions for stereo matching. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[61] H. Hirschmüller and D. Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1582–1599, 2008.

[62] Heiko Hirschmüller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 807–814, 2005.

[63] L. Hong and G. Chen. Segment-based stereo matching using graph cuts. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 74–81, 2004.

[64] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann. Local stereo matching using geodesic support weights. In *Proceedings of the International Conference on Image Processing*, pages 2093–2096, 2009.

[65] Z. Hou and W. Y. Yau. Relative gradients for image lighting correction. In *Proceedings of the International Conference on Acoustics, Speech and Signal Proceesing*, pages 1374–1377, 2010.

[66] H.Saito, M. Kimura, S. Yaguchi, and N. Inamoto. View interpolation of multiple cameras based on projective geometry. In *Proceedings of the International Workshop on Pattern Recognition and Understanding for Visual Information*, 2002.

[67] H. Huang, H. Huangy, S. Nain, Y. Hung, and T. Cheng. Disparity-based view morphing - a new technique for image-based rendering. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 9–16, 1998.

[68] A. Ishikawa, M. P. Tehrani, S. Naito, S. Sakazawa, and A. Koike. Free viewpoint video generation for walk-through experience using image-based rendering. In *Proceedings of the ACM international Conference on Multimedia*, pages 1007–1008, 2008.

[69] A. Jones, M. Lang, G. Fyffe, X. Yu, J. Busch, I. McDowall, M. Bolas, and P. Debevec. Achieving eye contact in a one-to-many 3d video teleconferencing system. *ACM Transactions on Graphics*, 28(3):1–8, 2009.

[70] N. Joshi, W. Matusik, and S. Avidan. Natural video matting using camera arrays. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 779–786, 2006.

[71] T. Kanade, P. J. Narayanan, and P. W. Rander. Virtualized reality: concepts and early results. In *Proceedings of the IEEE Workshop on Representation of Visual Scenes*, pages 69–76, 1995.

[72] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: theory and experiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):920–932, 1994.

[73] T. Kanade, P. W. Rander, and P. J. Narayanan. Virtualized reality: constructing virtual worlds real scenes. *IEEE MultiMedia Magazine*, 1(1):34–47, 1997.

[74] J. Kim, V. Kolmogorov, and R. Zabih. Visual correspondence using energy minimization and mutual information. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1003–1010, 2003.

[75] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *Proceedings of the International Conference on Pattern Recognition*, pages 15–18, 2006.

[76] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *Proceedings of the International Conference on Computer Vision*, volume II, pages 508–515, 2001.

[77] P. Labatut, J. P. Pons, and R. Keriven. Robust and efficient surface reconstruction from range data. *Computer Graphics Forum*, 28(8):2275–2290, 2009.

[78] S. H. Lai and S. D. Wei. Reliable image matching based on relative gradients. In *Proceedings of the International Conference on Pattern Recognition*, pages 802–805, 2002.

[79] S. Laveau and O. Faugeras. 3d scene representation as a collection of images. In *Proceedings of the International Conference on Pattern Recognition*, pages 689–691, 1994.

[80] S. H. Lee, J. Park, and C. W. Lee. A new stereo matching algorithm based on bayesian model. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 2769 – 2772, 1998.

[81] B. J. Lei and E. A. Hendriks. Real-time multi-step view reconstruction for a virtual teleconference system. *EURASIP Journal on Applied Signal Processing*, 2002(1):1067 – 1087, 2002.

[82] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 61–68, 2006.

[83] M. Levoy and P. Hanrahan. Light field rendering. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 31–42, 1996.

[84] M. Lhuillier and L. Quan. Image interpolation by joint view triangulation. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2139 – 2145, 2139 - 2145.

[85] S. Z. Li. *Markov random field modeling in image analysis (Third Edition)*. Springer-Verlag, New York, 2009.

[86] B. Likar and F. Pernus. A hierarchical approach to elastic registration based on mutual information. *Image Vision Computing*, 19(1-2):33–44, 2001.

[87] J. Liu, I. P. Beldie, and M. wöpking. A computational approach to establish eye contact in video communication. In *Proceedings of the Workshop on Stereoscopic and Three Dimensional Imaging*, pages 229– 234, 1995.

[88] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194(4262):283–287, 1976.

[89] M. McGuire, W. Matusik, H. Pfister, J. F. Hughes, and F. Durand. Defocus video matting. *ACM Transactions on Graphics*, 24(3):567–576, 2005.

[90] L. McMillan and G. Bishop. Plenoptic modeling: an image- based rendering system. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 39–46, 1995.

[91] J. McVeigh, M. W. Siegel, and A. G. Jordan. Intermediate view synthesis considering occluded and ambiguously referenced image regions. *Signal Processing: Image Communication*, 9(1):21–28, 1996.

[92] W. Miled, J. C. Pesquet, and M. Parent. A convex optimization approach for depth estimation under illumination variation. *IEEE Transactions on Image Processing*, 18(4):813–830, 2009.

[93] K. Min and J. Chun. Image-based 3d face modeling from stereo images. In *Proceedings of International Conference on Computational Science and Its Applications*, volume 3980, pages 410–419, 2006.

[94] M. Minsky. Telepresence. *Omni*, pages 45–51, 1980.

[95] Y. Mishima. Soft edge chroma-key generation based upon hexoctahedral color space. U.S. Patent 5355174, 1993.

[96] K. Nakazawa. Proposal of a new eye contact method for teleconferences. *IEICE Transactions on Conmmunication*, E76-B(6):618–625, 1993.

[97] K. I. Okada, F. Maeda, Y. Ickikawaa, and Y. Matsushita. Multi-party videoconferencing at virtual social distance: Majic design. In *Proceedings of ACM Conference on Computer Supported Cooperative Work*, pages 385–393, 1994.

[98] M. Ott, J. Lewis, and I. Cox. Teleconferencing eye contact using a virtual camera. In *Proceedings of the INTERACT and CHI Conference Companion on Human Factors in Computing Systems*, pages 119 – 110, 1993.

[99] J. H. Park and H. W. Park. Fast view interpolation of stereo images using image gradient and disparity triangulation. *Signal Process: Image Communication*, 18:401–416, 2003.

[100] V. Podlozhnyuk. Image convolution with cuda. Technical report, NVIDIA, Santa Clara, CA, 2007.

[101] S. Pollard, M. Pilu, S. Hayes, and A. Lorusso. View synthesis by trinocular edge matching and transfer. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 168–173, 1998.

[102] T. Porter and T. Duff. Compositing digital images. *Computer Graphics*, 18(3):253–259, 1984.

[103] M. J. D. Powell. Radial basis functions for multivariable interpolation: a review. In *IMA Conference on Algorithms for the Approximation of Functions and Data*, pages 143–167, 1985.

[104] P. K. Rana and M. Flierl. Depth consistency testing for improved view interpolation. In *Proceedings of the IEEE International Workshop on Multimedia Signal Processing*, pages 384–389, 2010.

[105] A. Rangarajan, H. Chui, and J. S. Duncan. Rigid point feature registration using mutual information. *Medical Image Analysis*, 3(4):425–440, 1999.

[106] C. Rhemann, C. Rother, and M. Gelautz. Improving color modeling for alpha matting. In *Proceedings of the British Machine Vision Conference*, pages 1155–1164, 2008.

[107] J. Rittscher, J. Kato, S. Joga, and A. Blake. A probabilistic background model for tracking. In *Proceedings of the European Conference on Computer Vision*, pages 336–350, 2000.

[108] S. Roy and I. J. Cox. A maximum-flow formulation of n-camera stereo correspondence problem. *International Journal of Computer Vision*, 34(2):147–161, 1998.

[109] M. Ruzon and C. Tomasi. Alpha estimation in natural images. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 18–25, 2000.

[110] D. Scharstein. Stereo vision for view synthesis. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 852–858, 1996.

[111] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, 2002.

[112] O. Schreer, K. Chang, E. Hendriks, J. M. Schraagen, J. Stone, E. Trucco, and M. Jewell. Virtual team user environment - a key application in telecommunication. In *Proceeding of eBusiness and eWork*, 2002.

[113] S. Seitz and C. Dyer. View morphing. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 21–30, 1996.

[114] S. A. Shafer. Using color to separate reflection components. *COLOR Research and Application*, 10(4):210–218, 1985.

[115] C. E. Shannon. A mathematical theory of communication. *The Bell Systmes technical Journal*, 27(3):379–423, 1948.

[116] H .Y. Shum and S. B. Kang. A review of image-based rendering techniques. In *Proceedings of the International Conference on Visual Communications and Image Processing*, pages 2–13, 2000.

[117] A. R. Smith and J. F. Blinn. Blue screen matting. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 259–268, 1996.

[118] J. Song, Y. Hwang, and H. Hong. View morphing based on auto-calibration for generation of in-between views. In *Proceedings of the International Conference on Computational Science and Its Applications*, pages 799–808, 2004.

[119] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 246–252, 1999.

[120] J. Sun, J. Y. Jia, C. K. Tang, and H. Y. Shum. Poisson matting. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 315–321, 2004.

[121] J. Sun, Y. Li, S. B. Kang, and H. Y. Shum. Flash matting. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 772–778, 2006.

[122] J. Sun, H. Y. Shum, and N. N. Zheng. Stereo matching using belief propagation. In *Proceedings of the European Conference on Computer Vision*, pages 450–452, 2002.

[123] M. Tanimoto. Free viewpoint television. *Journal of Three Dimensional Images*, 15(3):17–22, 2001.

[124] H. Tao, H. S. Sawhney, and R. Kumar. A global matching famework for stereo computation. In *Proceedings of the International Conference on Computer Vision*, volume 1, pages 532–539, 2001.

[125] F. Tombari, S. Mattoccia, and L. D. Stefano. Segmentation-based adaptive support for accurate stereo correspondence. In *Proceedings of Pacific-rim Symposium on Image and Video Technology*, pages 427–438, 2007.

[126] H. Towels, W. C. Chen, R. Yang, S. U. Kum, and H. Fuchs. 3d tele-collaboration over internet 2. In *Proceedings of the International Workshop on Immersive Telepresence*, pages 28–31, 2002.

[127] O. Veksler. Fast variable window for stereo correspondence using integral images. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 556–561, 2003.

[128] R. Vertegaal, I. Weevers, C. Sohn, and C. Cheung. Gaze-2: conveying eye contact in group video conferencing using eye-controlled camera direction. In *Proceedings of the International Conference on Human-Computer Interaction*, pages 521–528, 2003.

[129] P. Viola and W. M. Wells III. Alignment by maximization of mutual information. In *Proceedings of the International Conference on Computer Vision*, pages 16–23, 1995.

[130] P. Viola and M. Jones. Robust real-time face detection. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 1254–1259, 2001.

[131] J. Wang, M. Agrawala, and M. Cohen. Soft scissors: an interactive tool for realtime high quality matting. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques*, pages 853–861, 2007.

[132] J. Wang and M. Cohen. An iterative optimization approach for unified image segmentation and matting. In *Proceedings of the International Conference on Computer Vision*, pages 936–943, 2005.

[133] J. Wang and M. Cohen. Image and video matting: a survey. *Foundations and Trends in Computer Graphics and Vision*, 3(2):97–175, 2007.

[134] J. Wang and M. Cohen. Optimized color sampling for robust matting. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[135] L. Wang, C. Zhang, R. Yang, and C. Zhang. Tofcut: towards robust real-time foreground extraction using time-of-flight camera. In *Proceedings of the International Symposium on 3D Data Processing, Visualization and Transmission*, 2010.

[136] Z. Wang and Z. Zheng. A region based stereo matching algorithm using cooperative optimization. In *Proceedings of the International Conference on Pattern recognition*, pages 1–8, 2008.

[137] Y. Wei and L. Quan. Region-based progressive stereo matching. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 106–113, 2004.

[138] Q. Wu, P. Boulanger, and W. F. Bischof. Automatic bi-layer video segmentation based on sensor fusion. In *Proceedings of the International Conference on Pattern Recognition*, pages 1–4, 2008.

[139] G. Wyszecki and W. S. Styles. *Color science: concepts and methods, quantitative data and formulae*. Wiley, New York, 1982.

[140] J. Xiao and M. Shah. From image to video: view morphing of three images. In *Proceedings of the International Workshop on Vision, Modeling, and Visualization Conference*, pages 495–502, 2003.

[141] L. Xu and J. Jia. Stereo matching: an outliner confidence approach. In *Proceedings of the European Conference on Computer Vision*, pages 775–787, 2008.

[142] Q. Yáng, L. Wang, R. Yang, H. Stewénius, and D. Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling. In *Proceedings of the International Conference on Pattern recognition*, pages 347–354, 2006.

[143] R. Yang and Z. Zhang. Eye gaze correction with stereovision for video- teleconferencing. In *Proceedings of the European Conference on Computer Vision*, pages 479–494, 2002.

[144] B. Yip. Face and eye rectification in video conference using affine transform. In *Proceedings of the International Conference on Image Processing*, pages 513–516, 2005.

[145] K. Yoon and I. Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):650–656, 2006.

[146] K. J. Yoon and I. S. Kweon. Stereo matching with the distinctive similarity measure. In *Proceedings of the International Conference on Computer Vision*, pages 1–7, 2007.

[147] N. R. Yoon and B. U. Lee. Viewpoint interpolation using an ellipsoid head model for video teleconferencing. In *Proceedings of the International Symposium on Visual Computing*, pages 287–293, 2005.

[148] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of the European Conference of Computer Vision*, pages 151–158, 1994.

[149] K. Zhang, J. Lu, G. Lafruit, R. Lauwereins, and L. V. Gool. Robust stereo matching with fast normalized cross-correlation over shape-adaptive regions. In *Proceedings of the International Conference on Image Processing*, pages 2357–2360, 2009.

[150] Y. Zheng and C. Kambhamettu. Learning based digital matting. In *Proceedings of the International Conference on Computer Vision*, page 889 896, 2009.

[151] Y. Zheng, C. Kambhamettu, J. Yu, T. Bauer, and K. Steiner. Fuzzymatte: a computationally efficient scheme for interactive matting. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[152] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. *ACM Transactions on Graphics*, 23(3):600–608, 2004.