

# 國立交通大學

電子工程學系 電子研究所碩士班

碩士論文

使用 RGB-D 影像評估景深估測演算法

Evaluation of Disparity Estimation Schemes using  
Captured RGB-D Images

研究生：劉哲瑋

指導教授：杭學鳴 教授

中華民國一〇二年七月

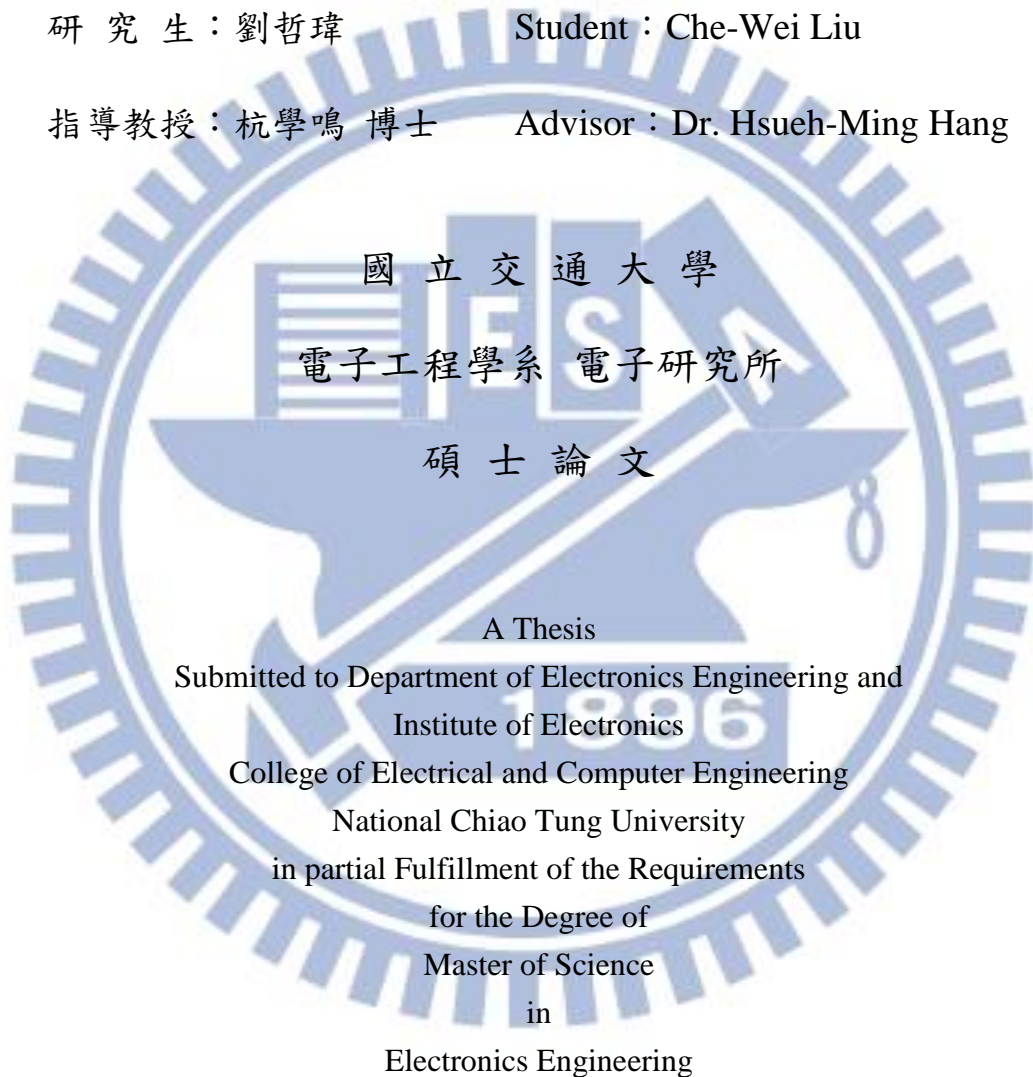
使用 RGB-D 影像評估景深估測演算法  
Evaluation of Disparity Estimation Schemes using  
Captured RGB-D Images

研究生：劉哲瑋

Student : Che-Wei Liu

指導教授：杭學鳴 博士

Advisor : Dr. Hsueh-Ming Hang



國立交通大學  
電子工程學系 電子研究所  
碩士論文

A Thesis  
Submitted to Department of Electronics Engineering and  
Institute of Electronics  
College of Electrical and Computer Engineering  
National Chiao Tung University  
in partial Fulfillment of the Requirements  
for the Degree of  
Master of Science  
in  
Electronics Engineering

July 2013

Hsinchu, Taiwan, Republic of China

中華民國一〇二年七月

# 使用 RGB-D 影像評估景深估測演算法

研究生：劉哲瑋

指導教授：杭學鳴 博士

國立交通大學

電子工程學系 電子研究所碩士班

## 摘要

在 3D 影像處理的領域中，基於左右視角的景深估測法，或者稱作立體匹配演算法，被廣泛運用於許多 3D 的應用程式中。其中一種使用景深的資訊作為輔助，抓取鏡頭中人的動作及姿勢，不同的應用程式對於深度圖準確率的要求皆不一樣，於是，為了給予不同的應用程式適合的景深估測法，如何評測這些方法成為研究人員傷腦筋的課題。傳統的方法，使用由少量電腦合成的測試影像組成的圖庫做評測，然而，這樣的方法對於處理真實場景的應用程式來說是不足夠的。

在此篇論文中，我們設計了多個場景，並由 RGB-D 攝影機抓取這些影像資料，組成一個圖庫，當中包含了多組的立體影像對及其對應的真實視差圖。我們依據可能會影響立體匹配演算法表現的多個因素，將這些立體影像對分成了兩大類：影像內容類及影像品質類。影像內容類包含了背景複雜度、物品數量、不同的手勢及不同圖樣的衣服；影像品質類有不同的 PSNR 及不同的影像校正誤差。

此外，每組立體影像對都有它對應的真實視差圖，所有的影像皆是擷取自一對 Kinect，為了產生適合圖庫的影像，我們需要對這些擷取到的左右彩色影像進行相機校正及影像校正，對於擷取到的景深圖，我們分別將它製成真實視差圖及而後評測用的 trimap。來自左右 Kinect 的彩色影像，我們用相機校正估測其相機參數，這些參數在影像校正時會用到；我們還需作色彩校正，解決兩張影像色調不同的問題。為了使真實視差圖更加完

美，我們提出了一個適應性補洞方法，其能夠自動分辨三個不同的黑洞種類，再依據種類的不同做補洞；最後，我們使用影像切割的概念，製作 trimap：一張由前景部分、後景部分及前景背景間區域所組成的影像。

評估方面，我們利用 trimap 分出前後景，並將錯誤量測的重點擺在前景區域，使用錯誤視差值比率及均方誤差作為錯誤量測的機制。實驗方面，我們用所提出的評估方法評測了三個立體匹配演算法，並從所收集的數據做比較分析。



# Evaluation of Disparity Estimation Schemes using Captured RGB-D Images

Student: Che-Wei Liu

Advisor: Dr. Hsueh-Ming Hang

Department of Electrical Engineering &

Institute of Electronics

National Chiao Tung University

## Abstract

In 3D image processing, the depth estimation based on the given left and right images (the so-called stereo matching algorithms) has been widely used in many 3D applications. One type of applications tracks the body motion and/or poses with the aid of depth information. How to evaluate depth estimation algorithms for different applications becomes an issue. The conventional method of evaluating these depth estimation algorithms is often using a small number of test computer-generated images, which is insufficient to reflect the problems in the real world applications.

In this study, we design a number of scenes and capture them using the RGB-D cameras; that is, our dataset consists of stereo pair images and their corresponding ground truth disparity map. Our dataset contains two categories of factors that may affect the performance of the stereo matching algorithms. They are image content factors and image quality factors. The image content factor group includes simple and complex backgrounds, different number of objects, different hand poses and clothing with various color patterns. In the group of

image quality factor, we create images with different PSNR and rectification errors.

In addition, each stereo pair has their ground truth disparity map. All images and the depth maps are captured by a pair of Kinect devices. To generate appropriate images for the test dataset, we need to calibrate and rectify the captured RGB image pairs and we also need to process the captured depth maps and create the so-called trimaps for evaluation purpose. For the left and right color images, because they come from different sensors, we must perform camera calibration to obtain the camera parameters, and color calibration to match colors in two images. Also, we align the left and right images using the existing camera rectification technique. To generate the ground truth disparity map, we first capture the raw depth map from Kinect, and we warp it from the view of the IR camera to the RGB camera. These depth maps have many black holes due to its sensing mechanism. To make the ground truth disparity map more reliable, we propose an adaptive hole-filling algorithm. Last, we adopt the matting segmentation concept to create a tri-value map (trimap) that classifies image pixels into foreground, background, and in-between regions. Our error metrics are bad-matching pixel rate and the mean square error between the ground truth disparity map and the estimated disparity map. We focus on the performance in the foreground region. In our experiments, three stereo matching algorithms are used to test our dataset and evaluation methodology. We analyze these algorithms based on the collected data.

## 誌謝

能夠完成碩士論文，最要感謝的就是杭學鳴老師。老師平時雖外務繁忙，仍定時與我們見面做討論。在專業領域上，老師給予了我許多研究方向的指導；同時，老師不時地提點我許多做研究的準則與方法，這些知識我相信在往後的職涯也是十分受用。除了課業方面，老師也會關心我們生活遇到的難題。這兩年很幸運能在老師的指導下學習，由衷地感謝！

感謝長廷與鈞凱，你們比我優秀太多了，看著你們強大的背影總能鞭策我自己，這兩年感謝你們在生活上及研究上的幫忙。感謝志堯，在每個方面都很突出，總是能夠給予我們這些後輩很多精準的建議及寶貴的經驗傳承。

同在 ED529 一起奮鬥的男鑫、夏銘、葆崧、信宏、家駿、子傑、暉翔、明孝、中威及琬瑜，有你們在的每一刻，整個實驗室就像一個溫馨的大家庭，充滿著熱鬧歡樂的氣氛，我自己也從個個身懷絕技的大家，學習認識到很多有趣的事物，謝謝你們！

感謝學長義文在研究上的幫忙，雖然已在業界打拼，但仍會抽空答覆我研究上的問題。感謝學弟妹，杭 group 的建誠、育綸、欣哲、靖倫、敬昆、司頻、治戎及哲瑋，同在 5 樓的資偉、偉生及昀楷，有些分擔了實驗室的雜務，有些幫了我研究上的忙，無論如何，感謝你們。

此外，感謝同是中央通訊人的晉瑋、昱銘、智皓、仁傑、昱銘、珮欣及巧翎，在不多的聚會裡分享生活趣事，在有困難的時候互相扶持。

最後，感謝無時無刻關心我的家人，給予我一個最溫暖的家及最堅強的支持，我才能夠在這裡完成我的碩士學位，繼續我人生的下段旅程。

# Content

摘要.....	I
Abstract .....	III
誌謝.....	V
Content.....	VI
List of Figures .....	VIII
List of Tables.....	XI
chapter 1 Introduction.....	1
1.1 Background .....	1
1.2 Motivations and Contributions.....	1
1.3 Organizations of Thesis.....	2
chapter 2 Depth Estimation.....	3
2.1 Introduction to Depth Estimation.....	3
2.2 Active Depth Sensing .....	4
2.3 Passive Depth Sensing .....	6
2.4 Comparison between active depth sensing and passive depth sensing .....	11
2.5 Evaluation of stereo matching algorithms.....	12
chapter 3 Calibration and rectification.....	15
3.1 Camera Calibration .....	15
3.2 Image Rectification .....	18
3.2.1. Epipolar Geometry .....	18
3.2.2. Rectification of stereo pairs .....	20
3.3 Color Calibration .....	21
chapter 4 Dataset Generation for Evaluation.....	22
4.1 Calibration of Kinect .....	23
4.1.1 Calibration of Kinect's RGB Camera .....	23



4.1.2	Calibration of Kinect's IR camera .....	25
4.2	RGB Image Rectification .....	27
4.3	Color Calibration of Kinect RGB Images.....	28
4.4	Generation of Disparity map's Ground Truth .....	30
4.4.1	Alignment of Depth Image and RGB Image.....	32
4.4.2	Hole-Filling Method .....	34
4.4.3	Conversion of Depth to Disparity .....	42
4.5	Introduction of Dataset and Error metrics .....	42
4.5.1	Image Content Factor .....	43
4.5.2	Image Quality Factor .....	48
4.5.3	Error Metrics.....	52
chapter 5	Experimental Results and Discussion .....	55
5.1	Experimental Environment .....	55
5.2	Three Stereo Matching Algorithms.....	56
5.3	Experimental Results on Image Content Factors.....	58
5.3.1	Background Complexity.....	58
5.3.2	Different numbers of Objects.....	65
5.3.3	Hand Poses .....	72
5.3.4	People in Different Clothes .....	74
5.4	Experimental Results on Image Quality Factors .....	77
5.4.1	PSNR .....	77
5.4.2	Rectification Errors .....	81
chapter 6	Conclusions and Future Work .....	85
6.1	Conclusions .....	85
6.2	Future Work.....	86
	Bibliography .....	87

# List of Figures

<i>Figure 1: The principle of TOF depth camera [4] .....</i>	4
<i>Figure 2: Simple concept of structured light depth camera [5] .....</i>	5
<i>Figure 3: The principle of structured-light depth camera [6].....</i>	5
<i>Figure 4: Introduction of Kinect for Xbox .....</i>	6
<i>Figure 5: The relation between the disparity and the perception of depth .....</i>	7
<i>Figure 6: An illustration of the transformation between depth and disparity.....</i>	8
<i>Figure 7: Quality comparison between active and passive depth sensing. ....</i>	12
<i>Figure 8: The pinhole model .....</i>	16
<i>Figure 9: The transformation between the world and camera coordinate frames .....</i>	17
<i>Figure 10: Epipolar geometry .....</i>	19
<i>Figure 11: Two rectified images.....</i>	20
<i>Figure 12: Flow chart of building the dataset.....</i>	23
<i>Figure 13: 18 pictures of checkerboard with different orientations .....</i>	24
<i>Figure 14: We use camera calibration toolbox to detect the feature points .....</i>	25
<i>Figure 15: How we get the IR images for calibrating the IR camera .....</i>	26
<i>Figure 16: The stereo pair before rectification .....</i>	27
<i>Figure 17: The stereo pair after rectification .....</i>	28
<i>Figure 18: Original images from left and right Kinects and their RGB histograms .....</i>	29
<i>Figure 19: The image from the left Kinect after color calibration .....</i>	30
<i>Figure 20: Flow chart of alignment of depth and RGB image.....</i>	32
<i>Figure 21: The depth image before and after 3D warping .....</i>	33
<i>Figure 22: Depth image after 3D warping and maximum filter .....</i>	33
<i>Figure 23: Image that we overlap the depth image and RGB image together.....</i>	34

Figure 24: One of the reasons why the black holes appear [22].....	35
Figure 25: IR light projects on objects made of reflective and refractive materials [22].....	36
Figure 26: (a) RGB image (b) depth map (c) binary map of holes .....	37
Figure 27: Group of holes .....	38
Figure 28: neighboring region of black holes .....	39
Figure 29: Holes are classified based on the number of the amount of dominant peaks. Red means one dominant peak. Green means two dominant peaks. Blue means three dominant peaks. Yellow means four or more dominant peaks. ....	40
Figure 30: The performance of our proposed hole-filling algorithm (a) the original depth map (b) the performance of [16] (c) the performance of our method .....	41
Figure 31: Conversion of depth to disparity. (a) depth map (b) disparity map .....	42
Figure 32: Stereoe pairs of textureless background and complex background .....	44
Figure 33: Stereo pairs of different cut .....	44
Figure 34: Pictures of different number of objects in the scene .....	46
Figure 35: Pictures with different hand poses .....	47
Figure 36: A person wears clothes with different patterns.....	48
Figure 37: Images with different PSNR .....	49
Figure 38: Unrectified and rectified images .....	50
Figure 39: Image with rectification error of 1-5 lines .....	52
Figure 40: RGB image and its trimap.....	53
Figure 41: Experiment set up .....	55
Figure 42: Flow diagram of the automatic mode [26].....	57
Figure 43: Disparity maps on the simple background test images .....	59
Figure 44: Disparity on complex background test images .....	60
Figure 45: Disparity maps on CUT1 images.....	62

<i>Figure 46: Disparity maps on CUT2 images</i> .....	62
<i>Figure 47: The ground truth disparity maps</i> .....	66
<i>Figure 48: The trimaps in this evaluation</i> .....	67
<i>Figure 49: Computed disparity map using WTA</i> .....	68
<i>Figure 50: Computed disparity map using DERS</i> .....	69
<i>Figure 51: Computed disparity map using GC-NS</i> .....	70
<i>Figure 52: Disparity maps on hadns down</i> .....	72
<i>Figure 53: Disparity maps on horizontal arms</i> .....	73
<i>Figure 54: Dispairty maps of two different hand poses with bad matching pixels marked on them as red</i> .....	74
<i>Figure 55: Disparity maps on images of multicolor plaid pattern T-shirt</i> .....	75
<i>Figure 56: Disparity maps on images with unicolor plaid pattern T-shirt</i> .....	76
<i>Figure 57: The ground truth disparity map</i> .....	77
<i>Figure 58: Computed disparity maps using WTA on gaussian noise images of different PSNR</i> .....	78
<i>Figure 59: Computed disparity maps using DERS on Gaussian noise images with different PSNR</i> .....	79
<i>Figure 60: Computed disparity maps using GC-NS on Gaussian noise images with different PSNR</i> .....	79
<i>Figure 61: Computed disarity maps on the rectified and unrectified images</i> .....	82

## List of Tables

<i>Table 1: Kinect technical specifications.....</i>	<i>30</i>
<i>Table 2: BPR of disparity maps on simple and complex background test images.....</i>	<i>60</i>
<i>Table 3: MSE of disparity map on simple and complex background test images .....</i>	<i>60</i>
<i>Table 4: Compison of BPR of computed disparity maps on different size of background .....</i>	<i>63</i>
<i>Table 5: Compison of MSE of computed disparity maps on different size of background .....</i>	<i>63</i>
<i>Table 6: BPR focusing on the repeated-pattern area.....</i>	<i>64</i>
<i>Table 7: MSE focusing on the repeated-pattern area.....</i>	<i>64</i>
<i>Table 8: BPR focusing on the complex area .....</i>	<i>65</i>
<i>Table 9: MSE focusing on the complex area .....</i>	<i>65</i>
<i>Table 10: BPR of computed disparity maps using images consisting of different amount of objects in the complex background.....</i>	<i>71</i>
<i>Table 11: BPR of computed disparity maps using images consisting of 3 objects in the simple background .....</i>	<i>71</i>
<i>Table 12: BPR of computed diparity maps with vertical arms .....</i>	<i>73</i>
<i>Table 13: MSE of computed diparity maps with horizontal arms .....</i>	<i>73</i>
<i>Table 14: BPR of the estimated disparity maps in Figure 55 and Figure 56.....</i>	<i>76</i>
<i>Table 15: MSE of the estimated disparity maps in Figure 55 and Figure 56.....</i>	<i>76</i>
<i>Table 16: BPR of computed disparity map from Figure 58, Figure 59, and Figure 60.....</i>	<i>80</i>
<i>Table 17: MSE of computed disparity map from Figure 58, Figure 59, and Figure 60.....</i>	<i>80</i>
<i>Table 18: Comparison of estimated disparity for images of no noise and images of PSNR=40.....</i>	<i>80</i>
<i>Table 19: BPR of computed disparity map on the rectified and unrectified images .....</i>	<i>82</i>
<i>Table 20: MSE of computed disparity map using rectified and unrectified images .....</i>	<i>83</i>
<i>Table 21: BPR of computed disparity map on the iamges of different rectification errors .....</i>	<i>84</i>
<i>Table 22: MSE of computed disparity map on the iamges of different rectification errors .....</i>	<i>84</i>

# chapter 1 Introduction

## 1.1 Background

Kinect is an innovation in human-computer interaction since it was released by Microsoft. The active depth sensor in Kinect provides the depth information, which can be very helpful for the applications such as human skeleton tracking. Some other systems use stereo RGB cameras to produce the depth information based on the stereo matching algorithms. This approach has the advantage of lower hardware cost since only one more camera is added to the system without the need of an active depth sensor. Here comes the challenge that there are many factors in a real scene, which lower the performance of stereo matching. Knowing which stereo matching algorithm is more robust to certain factors can help us to choose and improve the right matching algorithms for different applications. The focus of this thesis is design appropriate test images for evaluating stereo matching algorithms.

## 1.2 Motivations and Contributions

There were several evaluation methods proposed in the past years, and the method from [1] is the most popular one. The database contains a few image pairs with ground truth depth maps. It evaluates the algorithms by focusing on the accuracy in three regions: non-occluded region, discontinuity region and the entire image. However, this evaluation site does not match our purpose because the dataset contains a small number of images, which cannot cover the issues we are interested in such as complex background versus simple background, one object versus several objects, etc. In [2], they provide a dataset with three different noises: Gaussian noise, brightness differences, and blurring. But the sequences in the dataset are computer-generated, which are quite different from the real scenes.

In this thesis, we design a test dataset and propose an evaluation procedure. The dataset consists of stereo pair images of real scene and their corresponding ground truth disparity maps. These images reflect six kinds of factors that appear in the real world and may affect depth estimation performance. To produce the ground truth depth maps, we propose a hole-filling method to polish the captured disparity maps. We focus on the disparity in the foreground, which is more important in the human-computer interaction applications. In our experiments, we show the evaluation results produced by three stereo matching algorithms and we discuss the causes of leading to their varying performances on different scenes.

### **1.3 Organizations of Thesis**

In this thesis, we first introduce the depth estimation principles and the existing evaluation methods in chapter 2. In chapter 3, we talk about the fundamental principles of camera calibration, rectification and color calibration, which are used to generate the test dataset. Next, we describe how the dataset is created, and our error metrics to evaluate the stereo matching algorithms in chapter 4. In chapter 5, we show the experimental results and discussion. At last, conclusions and future work are mentioned in chapter 6.

## chapter 2 Depth Estimation

### 2.1 Introduction to Depth Estimation

In the recent years, 3D display becomes more and more popular. People love this kind of entertainment that gives them the depth perception. Besides, the motion sensing devices such as Kinect and Wii have huge success because people can involve deeper in the game when they can move their body to control the figures in the game. These two technologies mentioned above have the same feature – they all need the depth information. For example, Kinect uses the depth information to detect motion more precisely, and 3D movie need the depth to create disparity.

In image processing, the depth information is kept in the depth map. A depth map is an image that presents the distances from a camera to scene objects by grayscale. Typically, the depth value in the map is quantized into 256 levels (8-bit representation) by the following equation:

$$v = 255 \cdot \frac{\frac{1}{z} - \frac{1}{z_{far}}}{\frac{1}{z_{near}} - \frac{1}{z_{far}}} \quad (1)$$

where  $v$  is the quantized depth value,  $z$  is the original depth value, and  $Z_{near}$  and  $Z_{far}$  are the nearest and the farthest clipping planes in the 3D space. This non-uniform depth quantization is designed by Chai et al in [3]. When the quantized depth value is larger, the corresponding pixel is closer to the camera, and vice versa.

How to produce an accurate depth map is always an important issue for people doing research in the field of computer vision and image processing, and there are already many depth estimation methods to help people find the depth. Generally depth estimations can be divided into two categories: active depth sensing and passive depth sensing. We will describe



more details about them in the following sections.

## 2.2 Active Depth Sensing

The depth sensing devices which can directly measure the depth value are called active depth sensing devices. Different depth sensors have their own special sensing mechanism to operate in real time. Among these devices, the Time-of-flight (ToF) and the structured-light depth sensors are the most popular ones, and they happen to use infrared (IR) light in their sensing mechanism. In [4], they did an evaluation comparison between Time-of-flight and structured light depth cameras. Time-of-flight camera, such as Mesa SR4000, can estimate depth by measuring phase changing from the emitted infrared light to the reflected infrared light. Figure 1 shows that the green wave is the infrared light that is directed to the objects from the IR emitter. Blue wave is the infrared light that is reflected back to the receiver. The phase-delay between these two can be converted to depth of the scene objects.

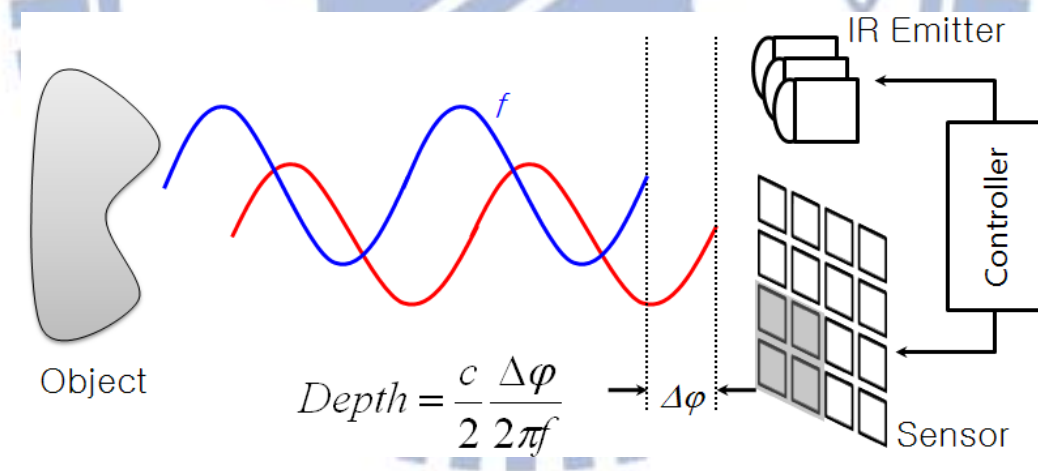


Figure 1: The principle of TOF depth camera [4]

Structured-light depth sensors use structured light scanning as its distance measurement method. As Figure 2 shows, the projector sends a pattern consisting of many stripes in different orientations. The pattern will distort geometrically so that we can get the 3D information of the scene objects. For example, the stripe will be straight when being projected

on the flat surface object, but distorted when being projected onto non-flat surface like hands.

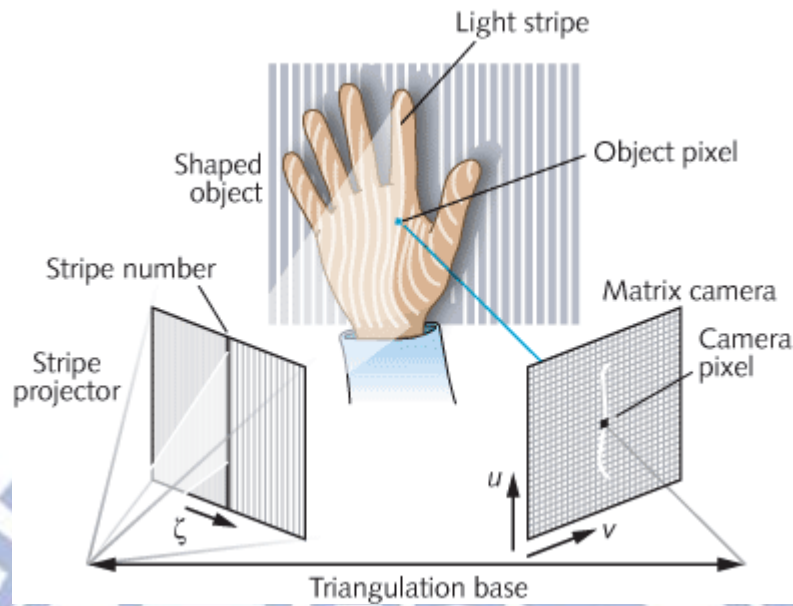


Figure 2: Simple concept of structured light depth camera [5]

Kinect is one of depth sensors that uses structured light sensing. However, Kinect's system is very different from the traditional method. The device consists of an infrared light projector, an infrared camera and a RGB camera. This is how it works: the IR light projector emits infrared light which goes through a Diffractive Optical Element and then becomes speckle patterns. These speckle patterns are projected on the objects in the scene and they have three different sizes when projected in the different depth range. Next, the IR camera detects these speckle patterns and the system on chip (SOC) calculates convert the data to the depth values. Figure 3 illustrates how Kinect estimates depth.

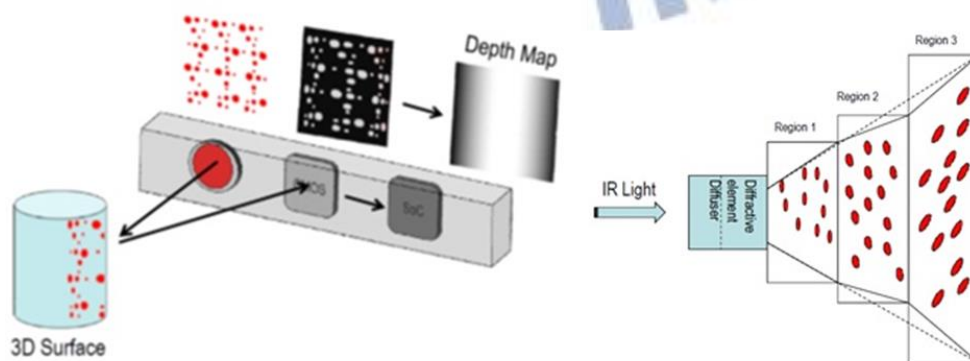


Figure 3: The principle of structured-light depth camera [6]

Kinect has been a great hit since it was released by Microsoft, because it is easy to access and not as expensive as Mesa SR4000. When playing motion sensing games, Kinect can track our body movement very well. When doing research, the official Kinect SDK or OPENNI provides many useful functions to make the best use of Kinect for researchers. Figure 4 shows the Kinect device.



Figure 4: Introduction of Kinect for Xbox

## 2.3 Passive Depth Sensing

Instead of using the depth sensing devices, we can also calculate the depth information from two or more images. Because we have to use the computer vision method to obtain the depth indirectly, we call this depth sensing method as passive depth sensing. Precisely speaking, what we calculate is not depth but disparity which can be converted to depth. Disparity is the shift when a point in the 3D world projects on different image planes. Because of disparity, we can perceive depth information, as the illustration in Figure 5. In Figure 5,  $O_L$  and  $O_R$  are the left and right camera centers, or we can regard them as our eyes.  $P$  is an arbitrarily point in the 3D world, and  $P_L$  and  $P_R$  are the points that  $P$  projects on the right and left image plane.  $X_L$  and  $X_R$  are the x coordinate of  $P_L$  and  $P_R$ . Because we assume that the

cameras are set in the 1D parallel configuration, the disparity between two corresponding points is  $X_R - X_L$ . As we can see P is closer to the camera in the left scene, which makes disparity value of the right corresponding pair is smaller than that of the left corresponding pair. This is how disparity information provides us the perception of depth.

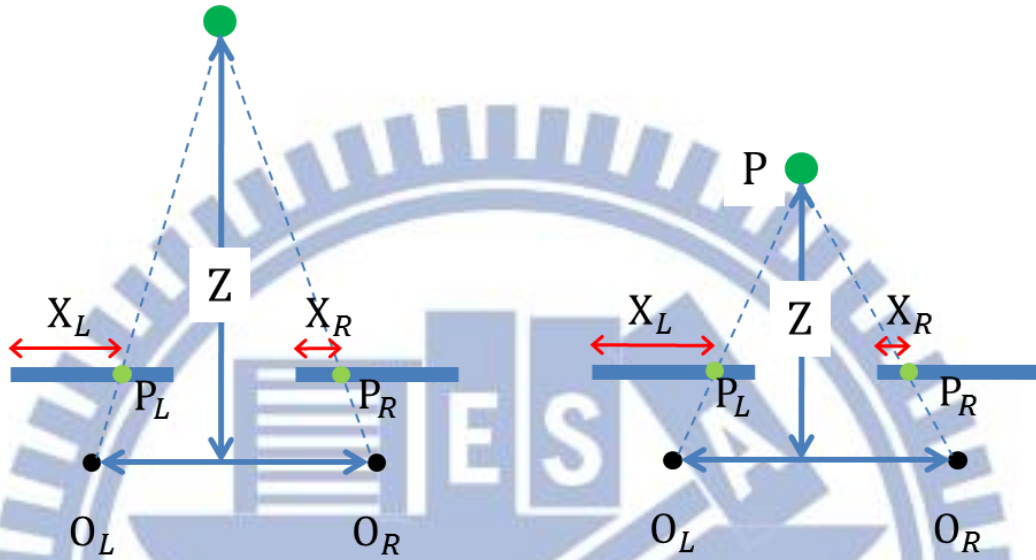


Figure 5: The relation between the disparity and the perception of depth

Back to the passive depth sensing method, to get the disparity value we have to find the pixel correspondences between two images. Once the disparity is correctly found, it can be converted to depth by triangulation. Their relationship is shown in Figure 6.

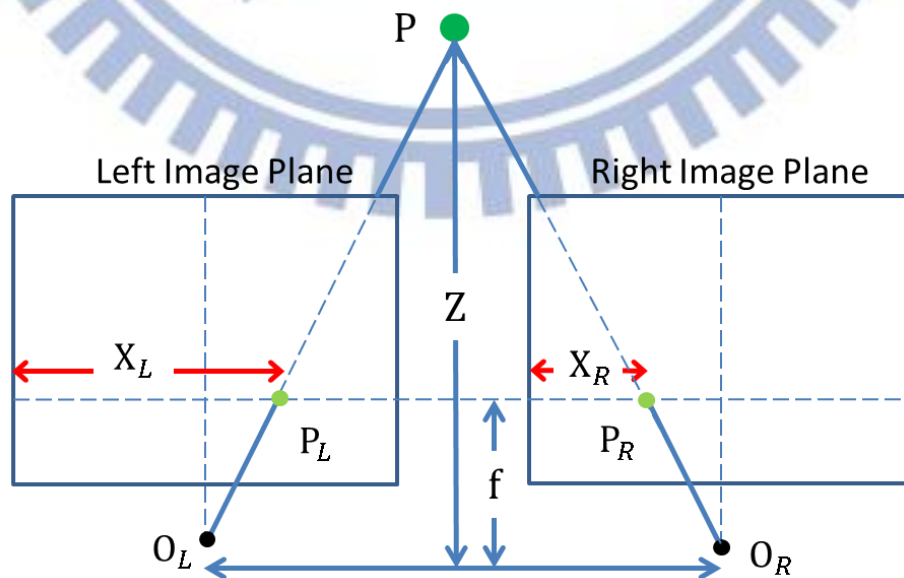


Figure 6: An illustration of the transformation between depth and disparity

In Fig. 1.6, most of the points have been defined in the previous paragraph.  $f$  is the focal length of two cameras.  $Z$  is an depth of the arbitrary 3D point  $p$ . Note that according to the 1D parallel configuration, the disparity of  $P_L$  and  $P_R$  is simply the horizontal shift:  $X_R - X_L$ . So the relation of depth and disparity, which can be derived from triangulation, is expressed as an equation:

$$Z = \frac{f \cdot b}{d}, \quad (2)$$

where  $d$  is the disparity and  $b$  is the baseline distance.

To sum up, the core concept of passive depth estimation is to find the matching pixels between two images. As a result, we call it passive depth estimation stereo matching algorithm. Nowadays, there are many state of the art stereo matching algorithms being proposed by the researchers. In general, there are four major steps performed in most stereo matching algorithms [7]:

1. Matching cost computation
2. Cost aggregation
3. Disparity computation/ optimization
4. Disparity refinement

In the first step, we calculate the matching cost which measures the similarity between pixels. That is to say, the smaller the cost, the higher probability that they may be a pair of corresponding pixels. From [8], we apply several dissimilarity measures as our matching cost. For example, the sum-of-squared-differences algorithm (SSD) or sum-of-absolute-differences algorithm (SAD) are commonly used. In addition, people use the metrics such the normalized-cross-correlation (NCC) algorithm or the zero-mean-sum-of-absolute-difference (ZSAD) algorithm to take the intensity differences caused by different shutter times, apertures of the cameras or light conditions into account.

Second, we aggregate the matching costs over a support window. This step can reduce the problem of errors due to noise sensitivity when using pixel-wise dissimilarity measure. But the drawback is that we will produce less detailed results, especially in the region of discontinuities. The bigger the window size is, the worse the quality of the discontinuous region. And if we set the window size small, the quality of the discontinuous region will perform better. However, it will become more sensitive to the noise. Not to fall into this trade-off, some people apply multiple windowing aggregation strategies instead of fixed window. Generally, these methods change its window size and shape adaptively depending on the texture of the neighbor pixels. In [9] they provide an adaptive window approach, which improves the results and can run in real time.

Third, we find the best match for each pixel. In this step, the algorithm can be categorized into local and global methods. Local methods simply choose the minimum matching cost at each pixel and this will determine the pixel's disparity. Winner-take-all (WTA) optimization is often applied in the local method. The advantage of local methods is that they are easy to implement and run in real time because of their simplicity. However, local methods have difficulty of handling the occlusion region, and they produce a smooth less result.

On the other hand, global methods determine the disparity by minimizing global energy function. The energy function usually contains a data term and smoothness term. It is shown below:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \quad (3)$$

$E_{data}(d)$  is the data term that measures the dissimilarity between pixels just like the matching cost mentioned in step 1. We define it as follow:

$$E_{data}(d) = \sum_{x,y} C(x, y, d(x, y)) \quad (4)$$

Where  $c$  is the cost function,  $x$  and  $y$  are the coordinates of the pixel to be found its

correspondence, and  $d(x, y)$  is the candidate of disparity  $d$ . The aggregation process can be ignored when we apply to global method.  $E_{smooth}(d)$  is the smooth term that computes the dissimilarity of disparities between the pixels to be matched and their neighbor pixels. Because we assume that the neighbor pixels may be more likely on the same objects, they should have equal or very similar disparity. To minimize the smooth term enable us to produce a more smooth disparity map. This explicit hypothesis is a feature in the global methods while the local methods make implicit smoothness assumptions. There are some methods to compute the smooth term. For example, we can easily use the difference of disparities between neighbor pixels as follow:

$$E_{data}(d) = \sum_{x,y} \rho(d(x, y) - d(x + 1, y)) + \rho(d(x, y) - d(x, y + 1)) \quad (5)$$

Where  $\rho$  is a monotonically increasing function of disparity difference.

Also, we can introduce the difference of intensity into the smooth term and it is shown as follows.

$$E_{data}(d) = \sum_{x,y} \rho_d(d(x, y) - d(x + 1, y)) \cdot \rho_i(\|I(x, y) - I(x + 1, y)\|) \quad (6)$$

Where  $\rho_i$  is a monotonically decreasing function of intensity difference that minimize the smooth term focusing on high intensity gradients.

Third, after introducing the cost function in local method and energy function in the global method, and we then talk about how to minimize them. There are several algorithms can be applied. For example, Graph Cuts and Belief Propagation which find the optimization by modeling the disparity map as a Markov Random Field (MRF) are commonly used to. A detailed comparison of the methods using MRF can be found in [10]. In addition, the dynamic programming technique which is an early termination method, is still used in the present stereo matching algorithm. Dynamic programming finds the global optimization along independent scan-line. In [11], they present a 2D scan-line structure instead of traditional 1D scan-line structure which causes streaking effect.

The advantage of global methods is that it can handle the uniform and occlusion texture well, and produces more smooth disparity results. As a result, the quality of the disparity map is better than the local method. Yet the computation time of the global methods is much longer than the local method because it seeks a global optimization.

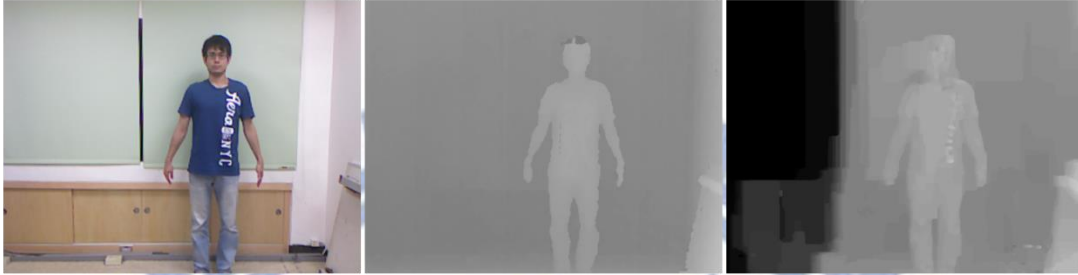
In the last step, we introduce some methods to make the disparity map more accurate. Although some applications such as human detection and skeleton tracking do not need very perfect disparity map, other applications like Depth Image Based Rendering (DIBR) cannot tolerate the disparity map with errors. Consequently, refinement of disparities is required. Many people use sub-pixel interpolation, which can be done by fitting a curve to the matching costs at discrete disparity values, or perform iterative gradient descent. Besides, people can simply use some of the filtering method as post processing of the disparity map. The morphological operators, median filtering and bilateral are commonly used.

## **2.4 Comparison between active depth sensing and passive depth sensing**

Active depth sensing, which uses a depth sensor to measure depth, can produce a depth map in real time. On the other hand, the passive depth sensing, which uses algorithm to calculate depth usually, has a longer computation time. Although researchers have proposed many real-time stereo matching algorithms, the quality still hardly competes with those having longer computation time. Active depth sensing can only be used in a limited range. For example, Kinect's sensing range is from 800mm to 4000mm, and SR4000 of wide field of view version's sensing range is from 100mm to 10000mm. However, passive depth sensing has no such limitation, and it is a better choice when we want to produce a depth map of outdoor scene. The overall quality of the depth map produced by active depth sensing is better



than passive depth sensing, though there are some noises caused by the scene objects with material that reflect or refract infrared light. If we focus on the quality of the object's edge, passive depth sensing is better than the active depth sensing. Figure 7 shows the quality difference between active and passive depth sensing:



(a) color image                      (b) depth map produced                      (c) depth map produced  
by active depth sensing                      by passive depth sensing

Figure 7: Quality comparison between active and passive depth sensing.

## 2.5 Evaluation of stereo matching algorithms

Stereo matching has been studied for a very long time. People work hard to solve all sorts of inaccurately computed disparity. Since there are many different kinds of stereo matching algorithms, people then try to build a system that can evaluate these algorithms fairly. Generally, we can divide the traditional evaluation method into two ways: qualitative and quantitative way. The qualitative evaluation method such as Structural Similarity (SSIM) illustrated in [12] used human related biasing factors to simulate a human-like evaluation result. Quantitative method thought to be more robust than the qualitative method is much more commonly used. Middlebury [1] is a platform devoted to quantitative evaluation methods and is well known by researchers in this area. There are more details in [7]. Middlebury provides several image pairs and their ground truths in their database. People use these image pairs to produce disparity maps by using their own algorithms. Then, they can upload these disparity maps to the website.

Middlebury evaluate the algorithms by the ratio of bad matching pixel in three sections: non-occluded areas, all areas and discontinuity areas. If the difference between the estimated disparity value and ground truth disparity value exceeds a given threshold, the pixel will be thought to be a bad matching pixel. The final evaluation result comes from comparing with all other algorithm uploaded to Middlebury: how the algorithm performs in these three parts will be ranked together with many other algorithms, and the rank is its score. The average of the three scores will be the final evaluation score. However, some researchers think Middlebury's evaluation methodology has some drawbacks. One of them is that it does not measure the magnitude of the estimated errors because no matter how big the errors are, they are all consider as bad matching pixel. Besides, errors below the threshold will be seen as a correctly estimated pixel, but this little error may affect some applications a lot such as DIBR. Moreover, since the matching cost is very important to the stereo matching algorithm, [13] presents an evaluation methodology that computes the insensitivity of different matching costs with respect to radiometric differences of the input images. Radiometric differences mean the difference between corresponding pixels. These variations can be caused by camera's settings or the position of illumination in the scene. To simulate some of the radiometric variations, the paper adds global intensity changes, local intensity changes and noise to the test images. After the disparity maps are calculated from the dataset by several stereo matching algorithms, they compute the bad matching pixel rate. The result they found is that all of the algorithms they tested cannot handle strong local radiometric changes caused by changing the location of the illumination.

Inspired by the evaluation methodology illustrated in the previous paragraph, we want to create an evaluation method that focuses on not only the textural accuracy but also the robustness against different kinds of factors influencing disparity's quality. The reason why we want to do this research is that there are many current applications make use of the depth

information. So we want to know which stereo algorithm produces results that satisfy certain application. This problem of choosing a proper algorithm based on how the algorithm performs when the input images may contain noise or lack texture, which may lead to poor depth estimation. In [2], they present an evaluation system that answers the question how the stereo algorithm performs under certain situations. It provides a dataset consists of several synthetic sequences, which are added with three noises: Gaussian noise, brightness difference and blurring. However, these sequences are not real and we know that we can get a totally different result using real sequences and synthetic sequences. Furthermore, the noises may not strong enough to handle the entire situation for different applications. In this thesis, we will present an evaluation methodology on the stereo matching algorithm using a dataset consisting of real images. Moreover, we add noises to the images. Roughly, there are two categories of the influencing factors: content factors and image quality factors.



## chapter 3 Calibration and rectification

In this chapter, we introduce three preprocessing algorithms applied to the stereo images before conducting the stereo matching. They are camera calibration, image rectification and color adjustment. In camera calibration, we find the transformation between the 3D world coordinates and 2D image coordinates, then we use this transformation to do the image rectification. Image rectification transforms the stereo images to a common image plane (epipolar geometry), which can simplify the search of correspondence to one dimension. In color adjustment, we ensure the same object feature points in two images have the same color intensity, because they usually are not the same due to different cameras or illumination reflection.

### 3.1 Camera Calibration

This is a process to find the camera parameters which enables us to fully understand the relation between the 3D world coordinates and 2D image coordinates. Camera parameters can be divided into intrinsic parameters and extrinsic parameters. Intrinsic parameters link 2D image coordinates to 3D camera coordinates, and the extrinsic parameters link 3D camera coordinates to 3D world coordinate. They are described in more detail in [14].

First, we introduce the intrinsic parameters. Intrinsic parameters can be described by a pinhole camera model, which is shown in the Figure 8:

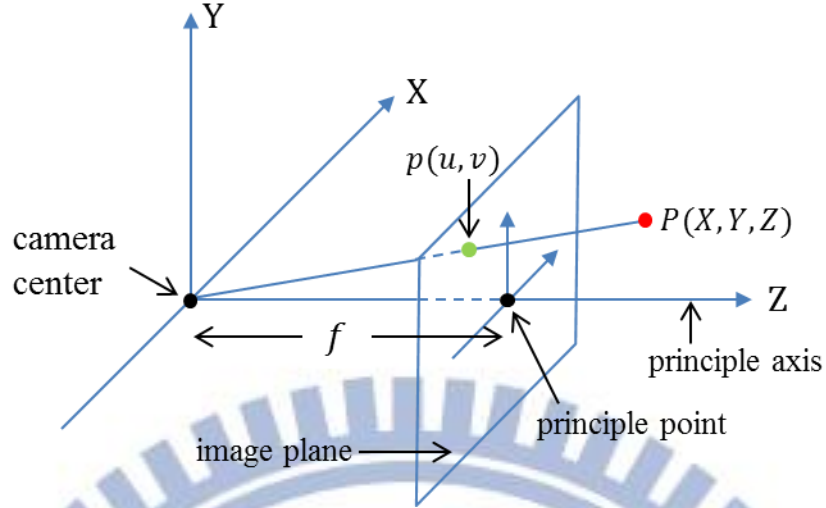


Figure 8: The pinhole model

This simplified model used for CCD like sensors describes the relationship between a point  $P$  in the 3D coordinate and its projection point  $p$  in the image plane. Camera center is the center of the projection, and is also called optical center. The line passing through the camera center and is perpendicular to the image plane is called the principal axis. The intersection of the principal axis and the image plane is the principal point. The coordinate of the principle point is  $(u_c, v_c)$ , and  $f$ , which stands for the distance between camera center and the principal point, is called the focal length of the camera. We can represent the projection point  $p$  by the following equation by triangulation:

$$(u, v) = \left( \frac{Xf}{Z} + u_c, \frac{Yf}{Z} + v_c \right), \quad (7)$$

where  $Z$  is the depth value of point  $P$ . If we rewrite  $p$  and  $P$  by using homogeneous coordinates, then the transformation from the image plane to the 3D camera coordinate can simply be a linear mapping, which is illustrated as follows:

$$Z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c & 0 \\ 0 & f_v & v_c & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (8)$$

In the above equation, we rewrite  $\begin{bmatrix} f_u & 0 & u_c & 0 \\ 0 & f_v & v_c & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$  into  $K[I | 0]$ , where  $K = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$ .

$f_u$  and  $f_v$  are the focal lengths of camera's x axis and y axis.  $\gamma$  is the skew parameter describing the angle between the x axis and y axis, and it is zero for most of the camera. Now, we let  $X_{cam}$  be a point in the 3D camera coordinates and  $x$  be the point in the image plane. Note that both of them are in the form of homogeneous vectors. So we can get a simple equation of relation between  $X_{cam}$  and  $x$ :

$$x = K[I \ 0]X_{cam} \quad (9)$$

We usually called the matrix  $K$  the intrinsic parameter matrix containing all the intrinsic parameters of camera.

Now we introduce the extrinsic parameters. As previously mentioned, the extrinsic parameters represent the link between 3D camera coordinates and 3D world coordinates. To illustrate this in details, first we take a look at Figure 9

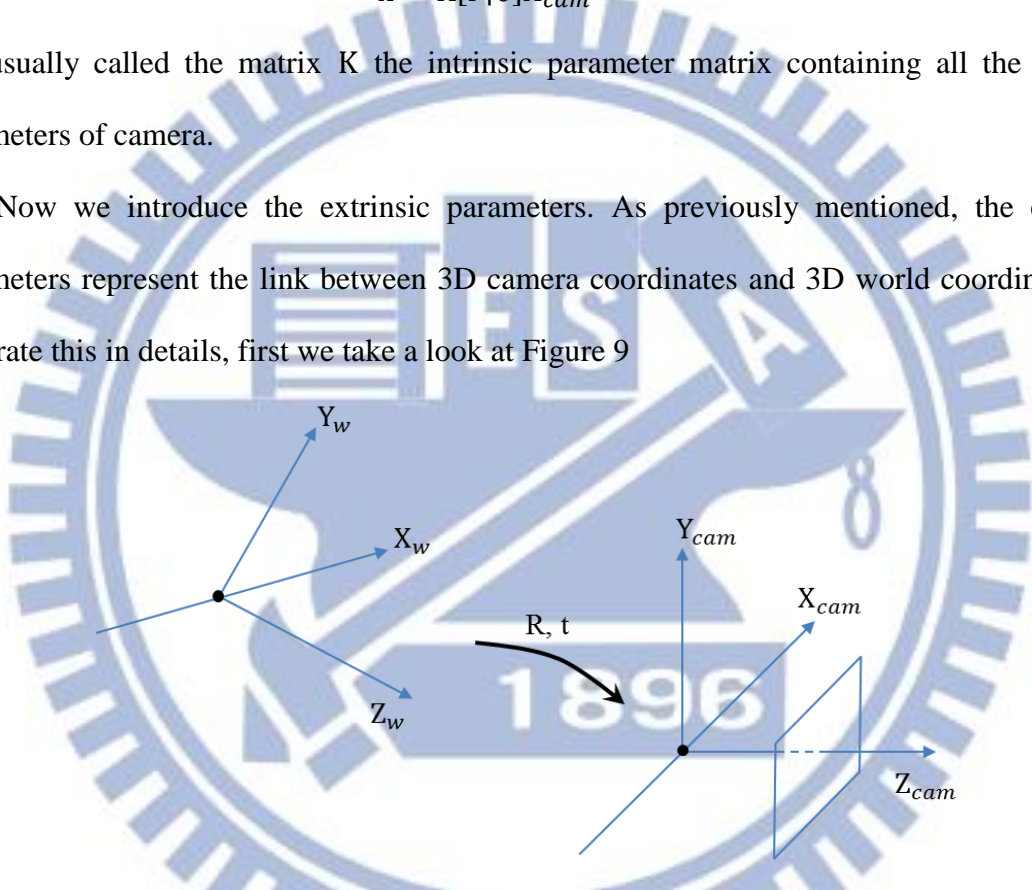


Figure 9: The transformation between the world and camera coordinate frames

We can see two different coordinate frames in the figure: one is world coordinate frame, and the other is camera coordinate frame. These two frames can be transformed by a rotation and a translation, and we call them the extrinsic parameters. Now we let  $\tilde{X}_{cam}$  be a point of inhomogeneous 3-vector in the camera coordinate frame, and  $\tilde{X}_w$  be the same point in the world coordinate frame. So we can write their relation in the following equation:

$$\tilde{X}_{cam} = R\tilde{X}_w + t \quad (10)$$

where  $R$  is a  $3 \times 3$  rotation matrix and  $t$  is a 3-vector translation:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, t = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (11)$$

Now we rewrite the relation between  $\tilde{X}_{cam}$  and  $\tilde{X}_w$  in homogeneous coordinate where they become  $X_{cam}$  and  $X_w$ :

$$X_{cam} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} X_w \quad (12)$$

Then we put the equation above into Eq. 9 and we get:

$$x = K \begin{bmatrix} I & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} X_w = K \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} X_w = M \cdot X_w \quad (13)$$

We call  $\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$  the extrinsic parameter matrix.  $M$  which is a  $3 \times 4$  matrix is the link of 3D world coordinate and 2D image plain, and we call it the projection matrix. Note that when we use the projection matrix, the two coordinates should be in homogeneous coordinates.

## 3.2 Image Rectification

Image rectification is the process to align the two images' corresponding epipolar lines. When doing the stereo matching, this preprocessing is very important because the search of the correspondence can be done simply along the horizontal lines after rectification. To rotate and translate the original images into the rectified images, we have to use the camera parameters obtaining from camera calibration. Before explaining how to do image rectification, we first introduce epipolar geometry, and there are more details in [錯誤! 找不到參照來源。\]](#).

### 3.2.1. Epipolar Geometry

To well describe epipolar geometry, we have to see the Figure 10 below:

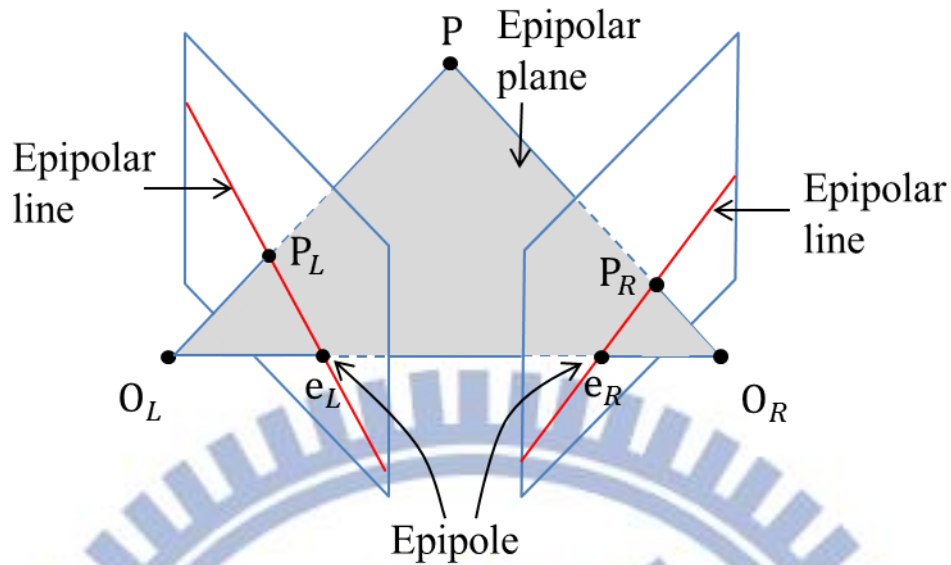


Figure 10: Epipolar geometry

$O_L$  and  $O_R$  are the left and right camera centers.  $P$  is a point in the 3D worlds, and  $P_L$  and  $P_R$  are the projections onto the left and right image planes. The line  $O_L O_R$  is called baseline. The right camera center  $O_R$  projects  $e_L$  into the left image plane, and  $e_R$  is similarly specified.  $e_L$  and  $e_R$  are called epipoles.  $P$ ,  $O_L$  and  $O_R$  form a plane which is call the epipolar plane. We can see that when the two cameras' positions are fixed, then the epipoles are fixed as well. So if we can find the corresponding pair of epipoles, we are able to know the relative positions of the cameras. If we see the line  $O_L P$  from the left camera, point  $P_L$  shows up on left image plane. But if we see from the right camera, it is a line on the left image plane. We call this line epipolar line. This epipolar line can help us find corresponding points. For examples: we want to find the corresponding point of  $P_L$  on the right image plane. Once we find the epipolar line on the right image plane, then the point  $P_R$  must lie on this line. This constraint, which we called it epipolar constraint, can simplify the work of stereo matching.



### 3.2.2. Rectification of stereo pairs

Now we know that epipolar constraint make stereo matching simpler. But when the epipolar lines are not horizontal, it causes problems when we turn the skew epipolar lines into pixel-based presentation. As a result we need to transform the stereo pairs into new ones that their epipolar lines are parallel and horizontal, and this transformation is called rectification. Fig 3.3 can be seen as two unrectified images, and Figure 11 shown below are the two rectified images:

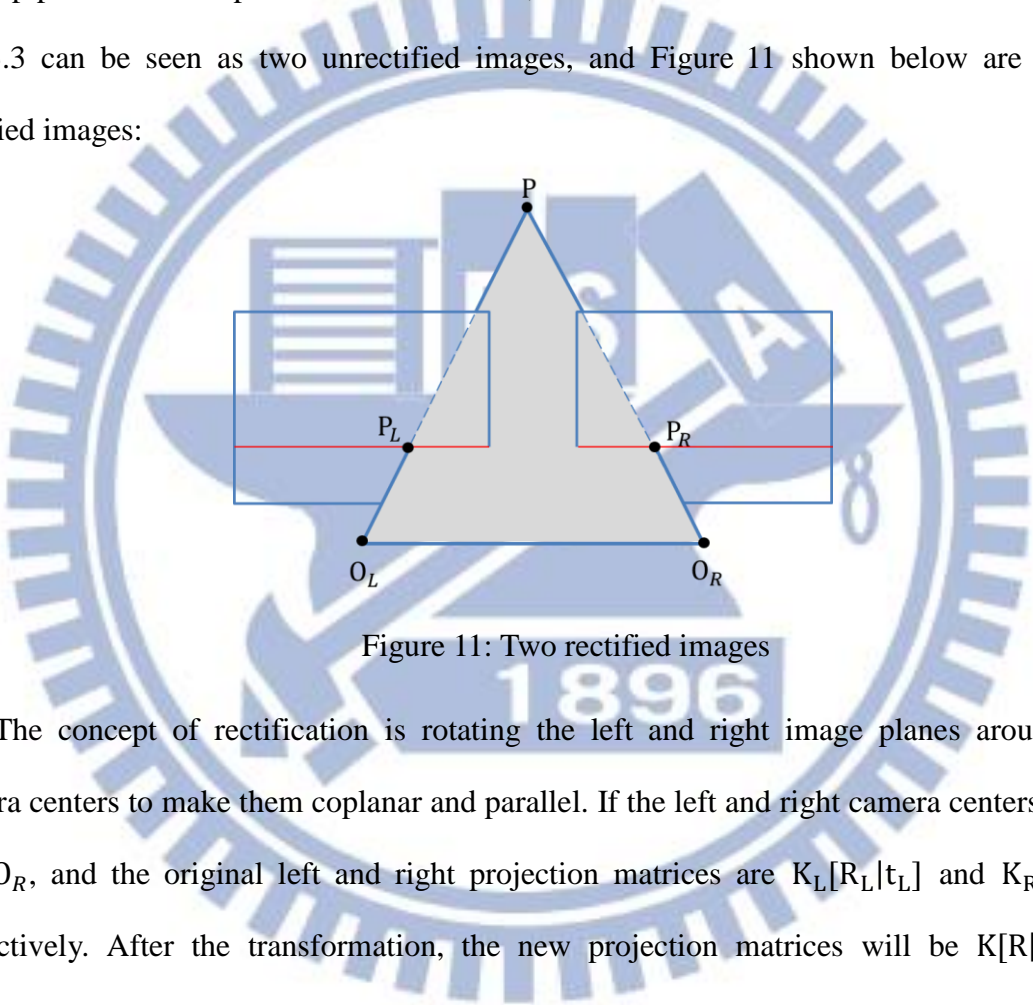


Figure 11: Two rectified images

The concept of rectification is rotating the left and right image planes around their camera centers to make them coplanar and parallel. If the left and right camera centers are  $O_L$  and  $O_R$ , and the original left and right projection matrices are  $K_L[R_L|t_L]$  and  $K_R[R_R|t_R]$  respectively. After the transformation, the new projection matrices will be  $K[R|t_L]$  and  $K[R|t_R]$ . The new intrinsic parameter matrix  $K$ , which can be set arbitrarily and we choose:

$$K = \frac{1}{2}(K_L + K_R) \quad (14)$$

The new rotation matrix is defined as follows:

$$R = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} \quad \text{where } r_1 = \frac{O_L - O_R}{\|O_L - O_R\|}, \quad r_2 = r_1 \times Z_{old}, \quad r_3 = r_1 \times r_2. \quad (15)$$

$r_1$  is the new X axis that must parallel to the baseline to make sure that the epipolar lines are

horizontal.  $r_2$  is the new Y axis which is the cross product of the new X axis and the Z unit vector of the original left matrix.  $r_3$  is the new Z axis which is the cross product of the new X axis and new Y axis. There are more details described in [15].

### 3.3 Color Calibration

Color calibration, which makes sure that a pair of correspondence has the same color intensity, is an important preprocessing step before conducting the stereo matching algorithm. Even the identical cameras may not have the same color responses, not to mention using different cameras. The concept is that we have the images from the left and right cameras, and then we match the color of one image to another by a certain algorithm. Here we introduce a simple algorithm we used in this thesis, histogram-based color calibration.

This is how the algorithm to be done. Given that we have a reference image  $I_1$  and the adjusted image  $I_2$ , calculating their histogram separately. These histograms represent the probability mass function (PMF), which describes the probability of each color intensity (0~255). Then we derive their cumulative mass functions (CDF) from PMFs:  $C_1$  and  $C_2$ . Because the CDF is monotonically increasing, we can design an inverse function for  $C_1$ . Finally we can write the relation between the old intensity and new intensity of  $I_2$ :

$$I_{2,new}(u, v) = C_1^{-1}[C_2[I_{2,old}(u, v)]], \quad (16)$$

where  $I_{2,old}(u, v)$  is the color intensity of  $I_2$  before color calibration, and  $I_{2,new}(u, v)$  is the color intensity after color calibration according to  $I_1$ .

## chapter 4 Dataset Generation for Evaluation

In this chapter, we introduce our evaluation methodology applied to the stereo matching algorithm. First, as mentioned in section 2.4, we build a dataset that consists of stereo pair images and their corresponding disparity map. The stereo pairs can be categorized into two groups according to the factors affecting disparity estimation: quality factor and content factor. The quality factor group contains images corrupted with White Gaussian Noise and rectification errors, and the content factor group contains images with different backgrounds, objects, hand poses and dresses. Three stereo matching algorithms are used to estimate disparity map of each stereo pair. Then, we evaluate the estimated disparity map using selected error metrics.

We capture the images and disparity maps from two Kinects. We need to rectify captured images and depth maps. First, we do camera calibration. After we obtain the camera parameters of Kinect, we rectify the images. Next, we perform color calibration on captured RGB images. After this step, most images are ready for evaluation. For testing purpose, we add additional processing: White Gaussian Noise and rectification errors. To generate the ground truth disparity map, we modify an existing procedure to refine the captured depth maps [16]. The flow chart in Figure 12 shows the processing steps. More details will be described later in the following sections.

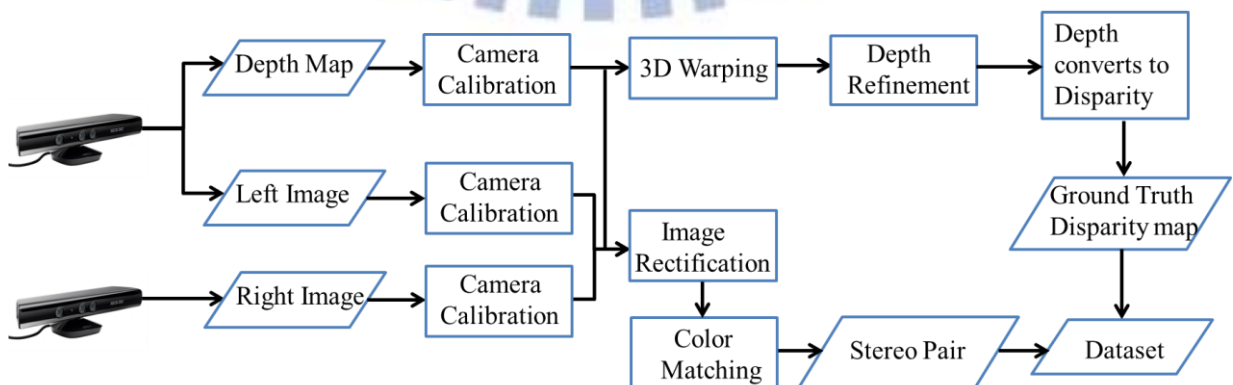


Figure 12: Flow chart of building the dataset

## 4.1 Calibration of Kinect

In section 3.1, we introduce why and how we need to do camera calibration. In section 4.1, we introduce the exact method we use to find the camera parameters. There are two different types of cameras on Kinect: RGB camera and an infrared (IR) camera. Generally, they use the same camera calibration method. However, the IR camera needs some preprocessing before calibration. As a result, we separate them into two parts. First we discuss the calibration of Kinect's RGB camera.

### 4.1.1 Calibration of Kinect's RGB Camera

In [17], they classify the camera calibration techniques into two groups: Photogrammetric Calibration and Self Calibration. In Photogrammetric Calibration, we take picture of a selected object which is usually composed of two or three planes and they are orthogonal. In addition, we need to know precisely the object's geometry in the 3-D space. In Self Calibration, we use one stationary object but we move a few times. Three or more images are taken by the same camera with fixed internal parameter at several locations. Then, we can reconstruct the 3-D scene based on the correspondence among these images. Reference [17] proposed a camera calibration procedure using Photogrammetric Calibration and Self calibration, and this procedure is widely used nowadays. The followings are brief description of their calibration procedure:

1. Print a pattern and attach it to a planar surface;
2. Take a few images of the object plane under different orientations by moving either the object or the camera;
3. Detect the feature points in the images;

4. Estimate the five intrinsic parameters and all the extrinsic parameters using the closed-form solution;
5. Estimate the coefficients of the radial distortion by solving the linear least-squares;
6. Refine all parameters by minimizing the maximum likelihood functions

In this thesis, we use a camera calibration toolbox provided by [18]. Their method is defined based on [17]. Users can easily access this toolbox because it is free online, and its GUI based interface makes users convenient to use. The patterns that we usually put on the planar surface are rectangular of alternating white and black colors. Because it looks like the checkerboard in playing English draughts, we call it checkerboard. The following pictures captured by the left Kinect are the checkerboard set at different orientations. Here, we take 18 pictures.

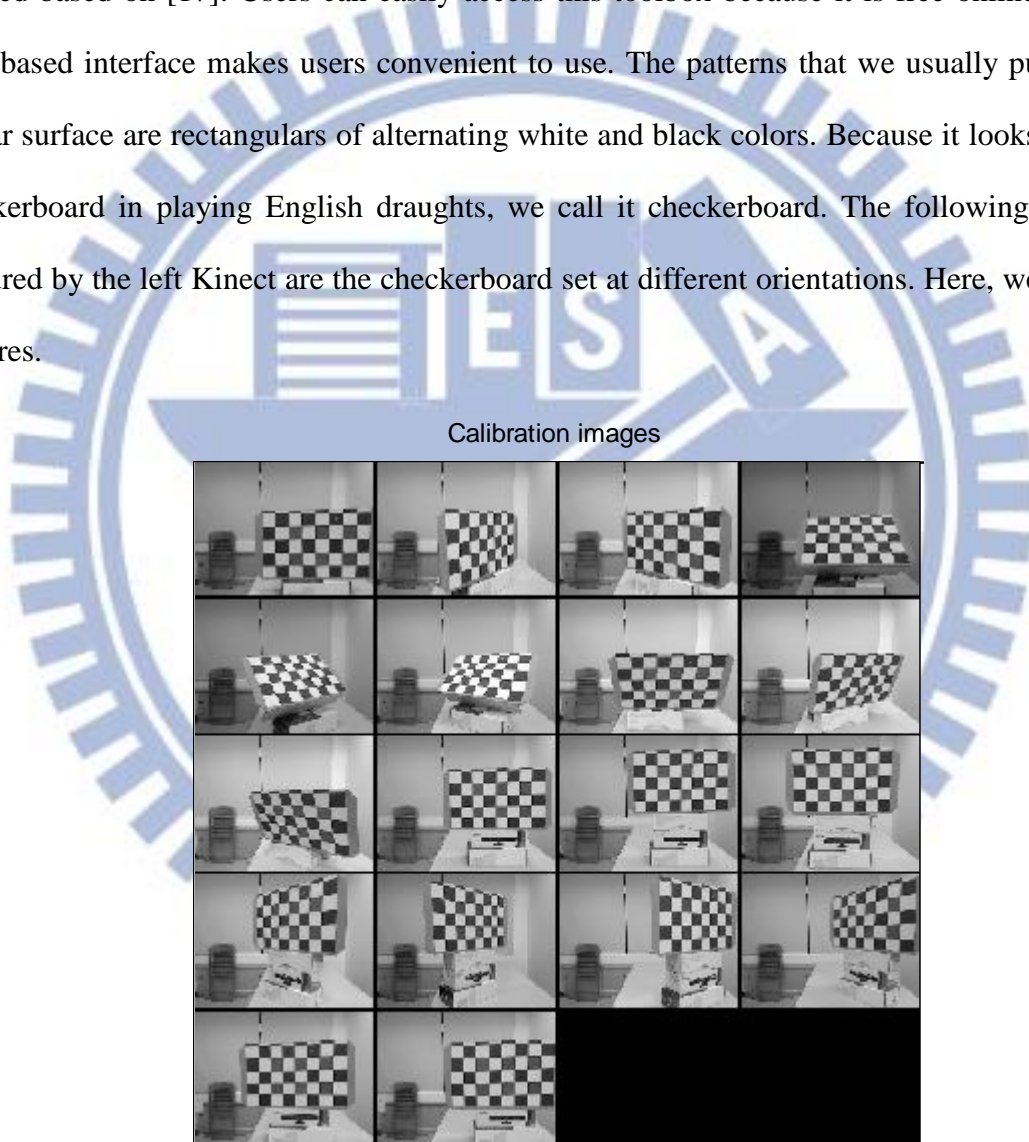


Figure 13: 18 pictures of checkerboard with different orientations

The feature points in this toolbox are the corners of each rectangular. The software can detect

the corners automatically after we draw an endorsed target region identically on each image and tell it how many rectangulars in the region. Figure 14 shows the software can precisely detect the corners:

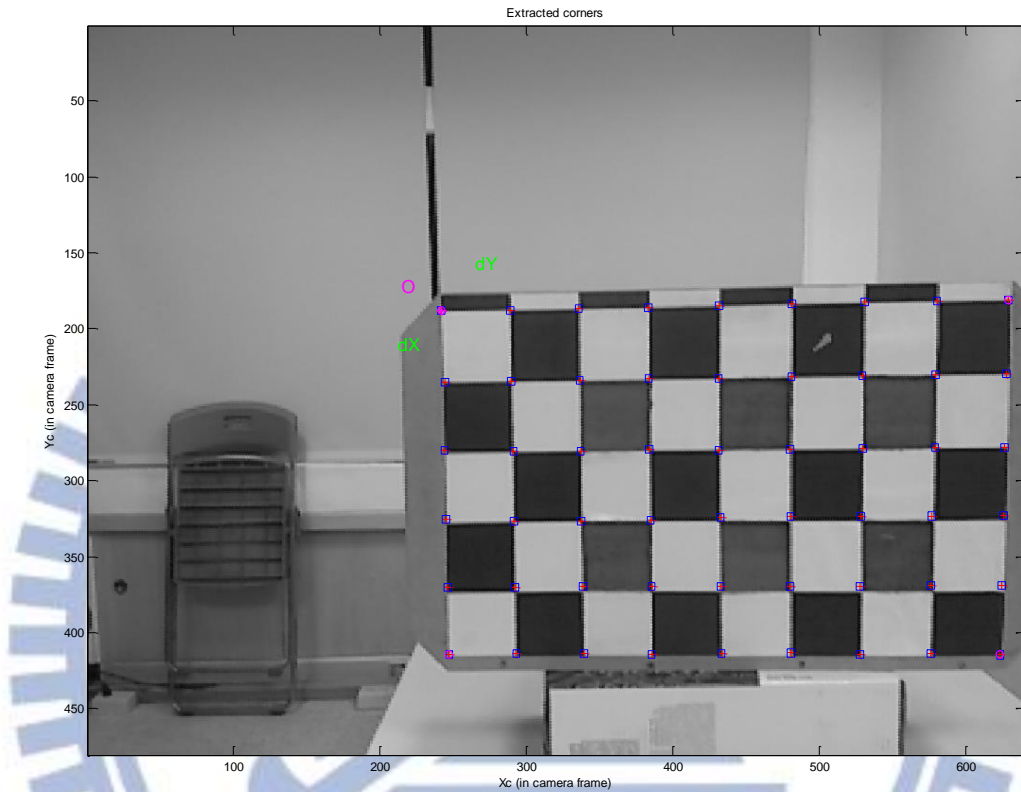


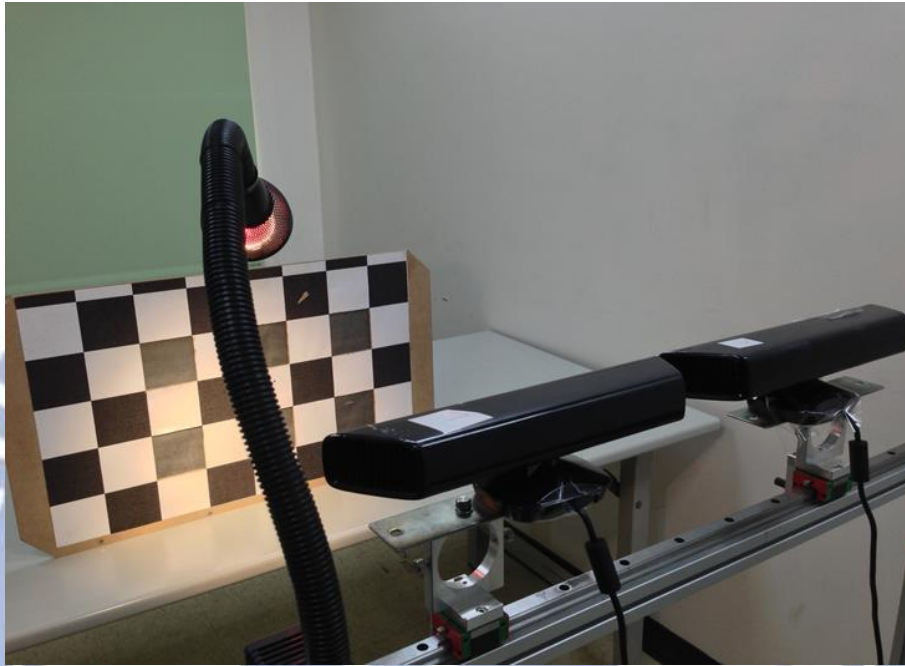
Figure 14: We use camera calibration toolbox to detect the feature points

After the feature points are detected, the toolbox makes an initial estimation of the planar homographies and then refined by the method presented in [17]. At the end, we can get the camera parameters of the Kinect's RGB camera.

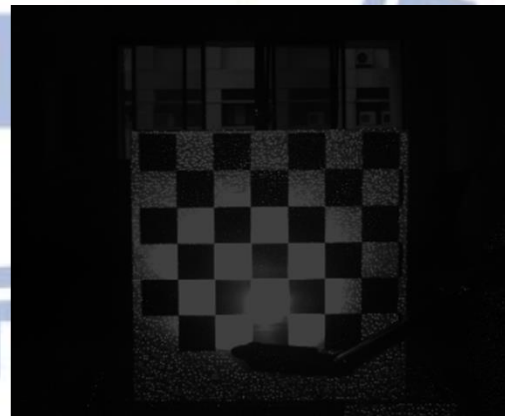
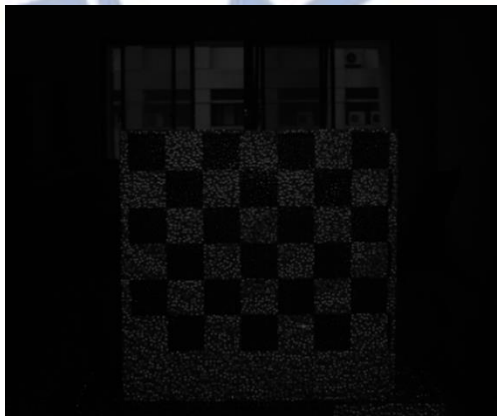
#### 4.1.2 Calibration of Kinect's IR camera

To calibrate the IR camera, we need to get the output images. As said in section 2.1: the IR camera captures the scene with speckle patterns which we call it IR image. Then, the SOC converts the IR image to depth image. Here we used the IR images to do camera calibration.

We use [18] for calibration, and the procedure is similar to that of the RGB camera. However, the raw IR image contains of black dots that affect the accuracy of the calibration results. In order to remove the dots, we use the method suggested by [19]: block the IR projector and illuminate the checkerboard by a halogen. Figure 15 shows how we set up the equipment.



(a) Environment setup for calibrating the IR camera



(b) Raw IR image of checkerboard

(c) IR image of checkerboard illuminated

by a halogen lamp instead of IR projector

Figure 15: How we get the IR images for calibrating the IR camera

## 4.2 RGB Image Rectification

In section 3.2.2, we talk about how to get the new camera parameters that make the two image's epipolar lines parallel and horizontal. Here we talk more about how to compute the transformation from the original image plane onto the rectified image plane. According to section 3.2.2, the original left projection matrix is  $P_{ol} = K_L[R_L|t_L]$ . The a new projection matrix will be  $P_{nl} = K[R|t_L]$ . To transform a 3D point  $\mathbf{w}$  to new image plane, we can write:

$$\begin{cases} \mathbf{m}_{ol} \cong P_{ol}\mathbf{w} \\ \mathbf{m}_{nl} \cong P_{nl}\mathbf{w} \end{cases} \quad (17)$$

$\mathbf{m}_{ol}$  and  $\mathbf{m}_{nl}$  are the points on the original and the new image plane. From the above equations, we can easily derive the  $3 \times 3$  transformation matrix  $T = P_{nl}P_{ol}^{-1}$ , and we can rewrite the  $\mathbf{m}_{nl}$  as follow:

$$\mathbf{m}_{nl} = T \cdot \mathbf{m}_{ol} \quad (18)$$

We apply the same method to the right image as well. Figure 16 and Figure 17 shows our result of rectifying the stereo pair image, and we also mark the feature points and its corresponding epipolar lines. After rectification, the epipolar lines in two images become horizontal.

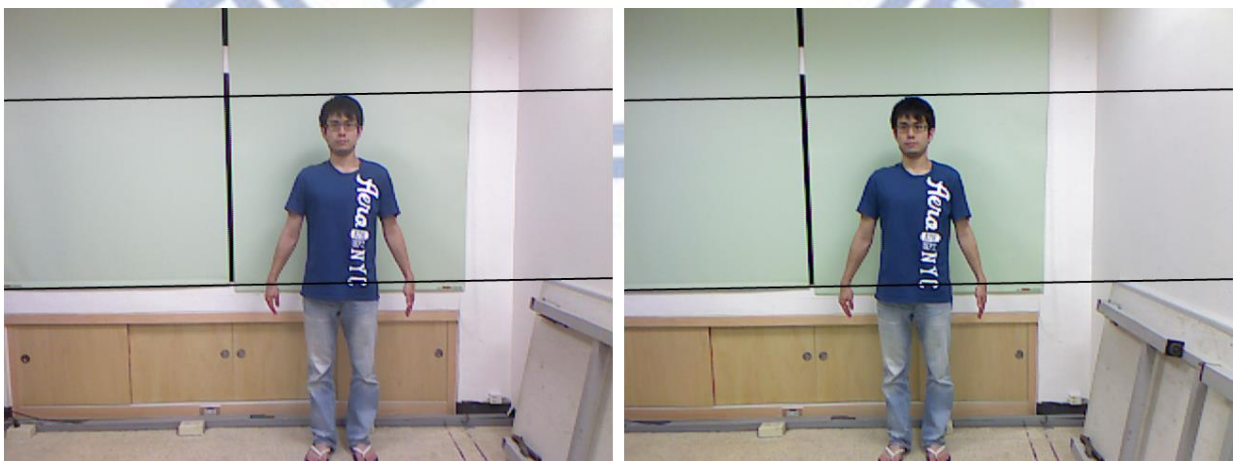


Figure 16: The stereo pair before rectification



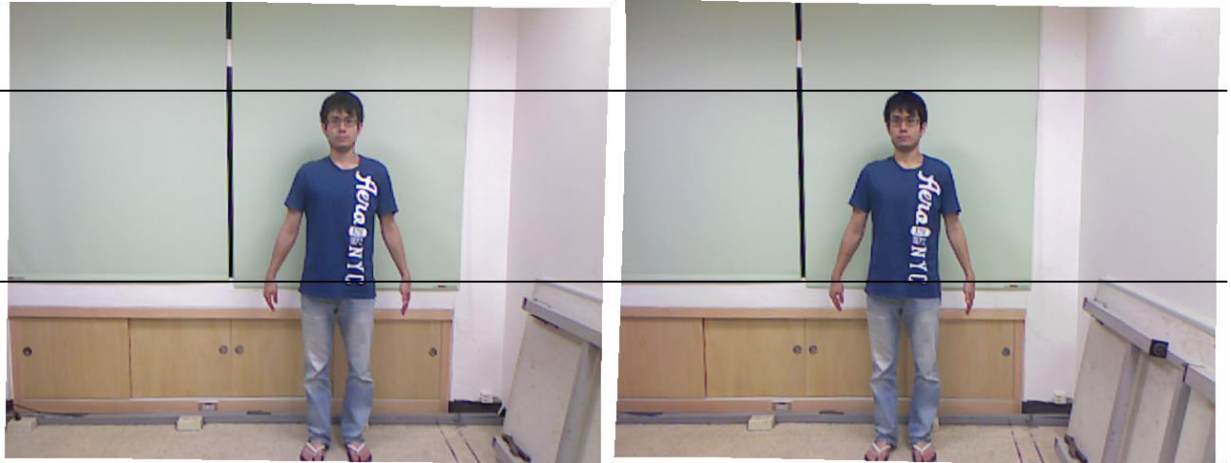
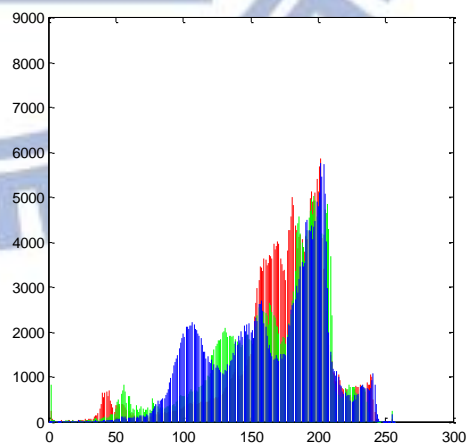


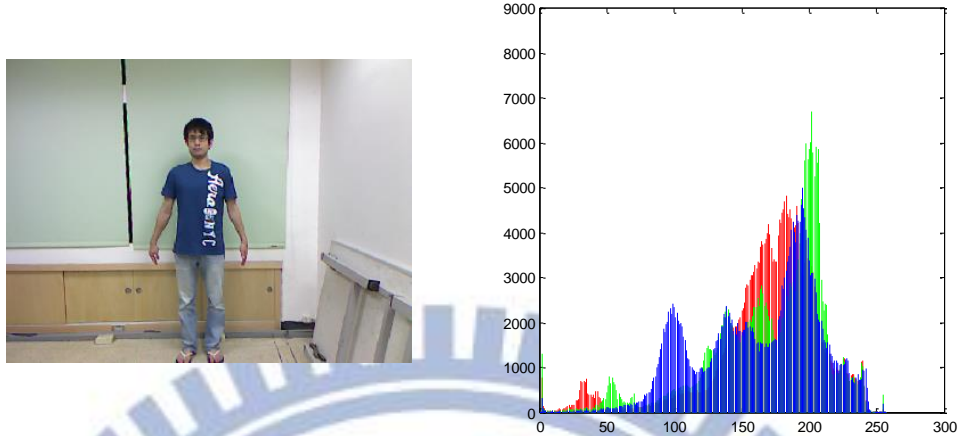
Figure 17: The stereo pair after rectification

### 4.3 Color Calibration of Kinect RGB Images

We apply color calibration to RGB images before stereo matching to reduce intensity differences between two corresponding pixels. Because the color tone of the RGB camera of Kinect is uncontrollable, the color difference of the same object can be quite significant. Hence, the color calibration process becomes even more important. Figure 18 shows a typical sample.



(a) Raw RGB image captured by the left Kinect



(b) Raw RGB image captured by the right Kinect

Figure 18: Original images from left and right Kinects and their RGB histograms

Comparing their color histograms, we can easily see that there are a lot different between the pictures captured by two Kinects. As a result, color calibration must be performed on the images captured by different Kinects. The algorithm we use is described in section 3.3. Here, we set the image from the left Kinect as the reference image, and we set the right Kinect image as the target image. The target image color histogram will be mapped to the reference image's color histogram. The following figures show the results: image from the right Kinect before and after color calibration. With color calibration, the colors of the target image matches those of the reference image much better both subjectively or objectively.

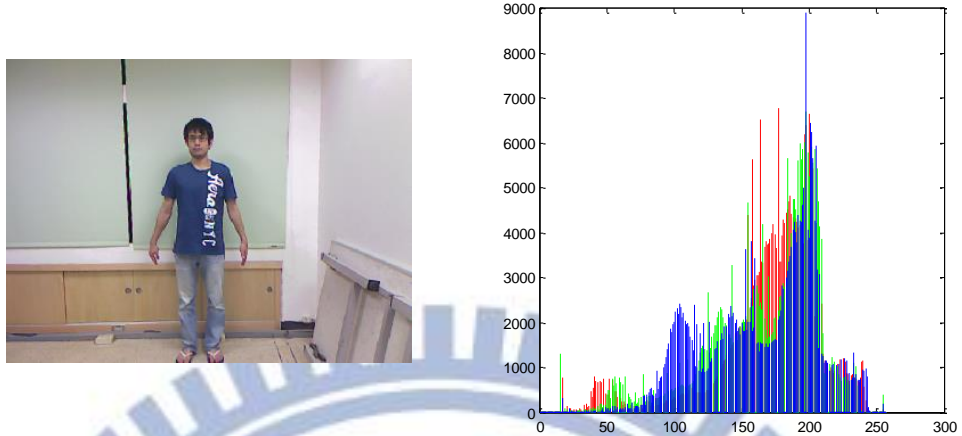


Figure 19: The image from the left Kinect after color calibration

#### 4.4 Generation of Disparity map's Ground Truth

In this study, the ground truth disparity map is necessary for our evaluation. After a few attempts, we finally decided to use the depth data captured by Microsoft Kinect. Generally, Kinect has three kinds of outputs: RGB image, IR image and depth image. The maximal resolution of the depth image is  $320 \times 240$ . Each pixel of the depth image has a 12-bits data which represents its actual distance in millimeters. Table 1 from [20] shows the Kinect's technical specification:

Table 1: Kinect technical specifications

Sensor	Color and depth-sensing sensors
	Voice microphone array
	Tilt motor for sensor adjustment

Field of View	Horizontal field of view: 57 degrees
	Vertical field of view: 43 degrees
	Physical tilt range: $\pm 27$ degrees
	Depth sensor range: 1.2m-4m
Data Streams	320×240 16-bit depth @ 30 frames/sec
	640×480 32-bit color @ 30 frames/sec
	16-bit audio @ 16kHz
Skeleton Tracking	Tracks up to 6 people, including 2 active players
System	Tracks 20 joints per active player
Audio System	XBOX LIVE party chat and in-game voice chat
	Echo cancellation system enhances voice input
	Speech recognition in multiple languages

According to Table 1, Kinect provides many ways for us to interact with XBOX 360 and computers. Microsoft releases Windows software development kit (SDK) for Kinect that contains drivers, tools, APIs, device interfaces and code samples that help people to develop applications. Microsoft updates the Kinect SDK frequently. Through SDK, we can access and control those sensors. More information about the Kinect SDK can be found at [21].

In this section, we introduce how to convert the depth image to the ground truth disparity map. In our system, the ground truth disparity map is chosen for the left image, so we capture the depth image from the left Kinect. This raw disparity map has to go through the following processing: (1) alignment of Depth image and RGB image, (2) hole filling and (3) conversion

of depth to disparity. First, we discuss the alignment between depth image and RGB image.

#### 4.4.1 Alignment of Depth Image and RGB Image

The first thing we have to do is to match the raw depth image and the RGB image from the left Kinect, because they come from two different sensors. Figure 20 illustrates how we do the alignment:

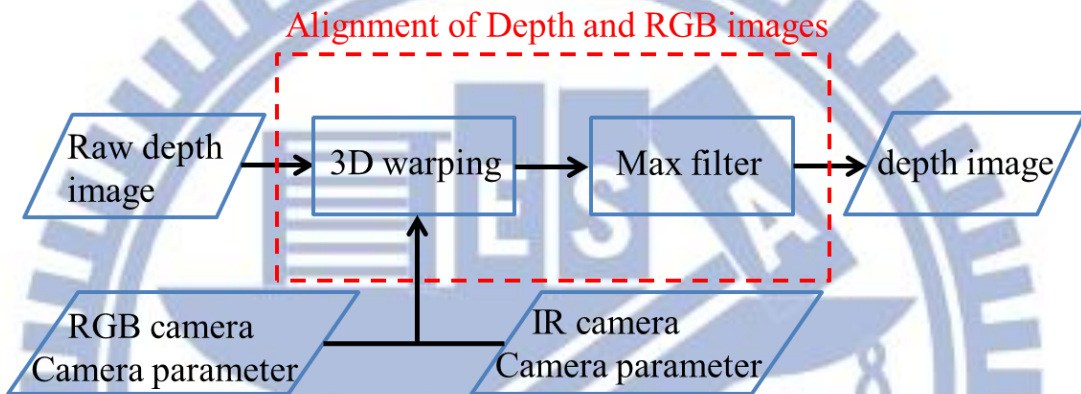


Figure 20: Flow chart of alignment of depth and RGB image

The concept is that we convert the depth image from the depth image coordinate to the RGB image coordinate. Since we can get the camera parameters applying camera calibration to the RGB camera and the IR camera, this job seems to a routine. What we need to do is to project each pixel in the depth image into the 3D world coordinates, and then reproject them into the RGB image coordinates. We call this algorithm 3D warping. Assume that  $X_{depth}$  is the position of a pixel in the raw depth image,  $P_{IR}$  is the projection matrix of the IR camera and  $P_{RGB}$  is the projection matrix of the RGB camera.  $X_{new\ depth}$  is the pixel's new position after 3D warping, and then the warping is represented by the following equation:

$$X_{new\ depth} = P_{RGB} \cdot P_{IR}^{-1} \cdot X_{depth}. \quad (19)$$

It is very similar to Eq. 18 in section 4.2. Rectification is to project a point to a virtual view, while here we project a point to an existing camera's view.

Fortunately, we can use Kinect SDK to do 3D warping. Figure 21 shows the depth image before and after 3D warping.

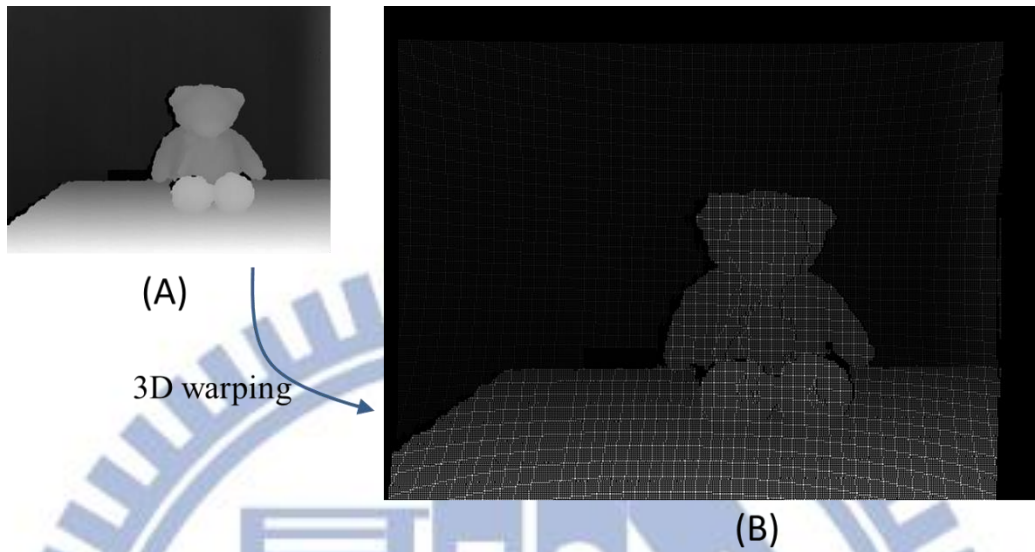


Figure 21: The depth image before and after 3D warping

Because 3D warping is a one to one warping, half of the pixels in Figure 21 (B) do not have information because the resolution of input image is  $320 \times 240$  and the output image is  $640 \times 480$ . Here, we use the maximum filter to fill the empty space. Figure 22 can shows the results after filling:

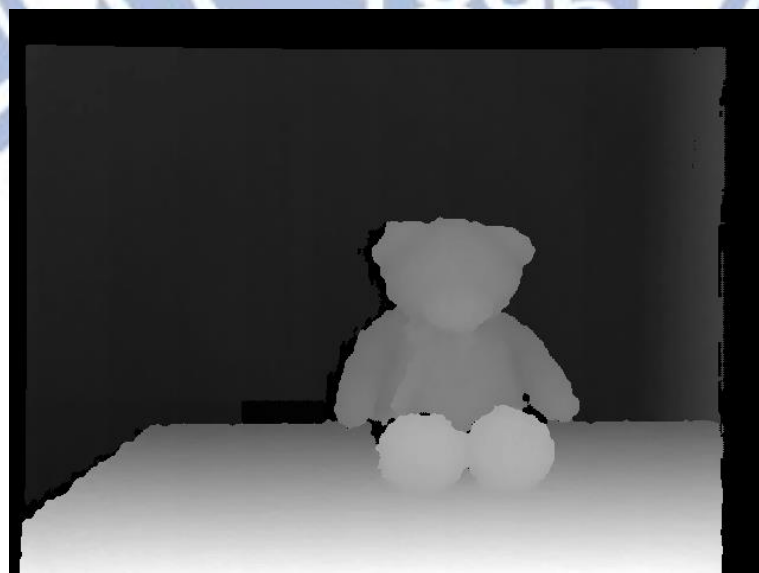


Figure 22: Depth image after 3D warping and maximum filter

To check whether the alignment works, we overlap the depth image and RGB image together

in Figure 23



Figure 23: Image that we overlap the depth image and RGB image together

Examine the edge carefully, we can find that some regions do not match exactly. There are two possible reasons. One is that the function from the Kinect SDK may not be very accurate, and another one is the inaccuracy due to the way Kinect estimates depth. The small misalignment will cause a lot of problems if we use the depth map to do the applications such DIBR. However, the depth map in our proposed evaluation system can tolerate a little misalignment. As a result, the SDK outputs are ok for our study. From Figure 23, there are still some black regions in the image. The black regions around pictures borders come from 3D warping, and we can cut off this region. The resolution of the depth map then becomes  $550 \times 420$ . The black holes inside the picture will be filled using our hole-filling algorithm, which will be introduced in the next section.

#### 4.4.2 Hole-Filling Method

The black holes appear mainly because the IR camera detects no speckle patterns in these areas. There are two reasons. The First reason is that the objects in the foreground obstruct the speckle patterns that are to be projected to in the background. So the occlusion region in the

background does not have depth information. It marked by dark area in the raw depth image.

Figure 24 from [22] can explain this phenomenon.

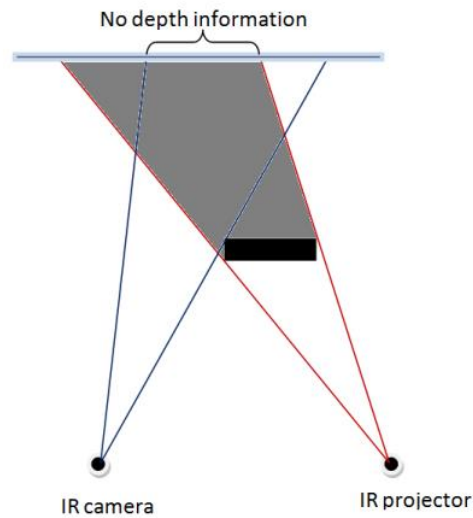


Figure 24: One of the reasons why the black holes appear [22]

The second reason is that the object may produce some optical phenomena such as refraction and reflection. The object material such as glass will refract the light, and thus the IR camera cannot receive the speckle pattern correctly. This phenomenon happens when the object is made of mirror-like material that will reflect the speckle pattern. Figure 25 ([22]) describes this phenomenon.



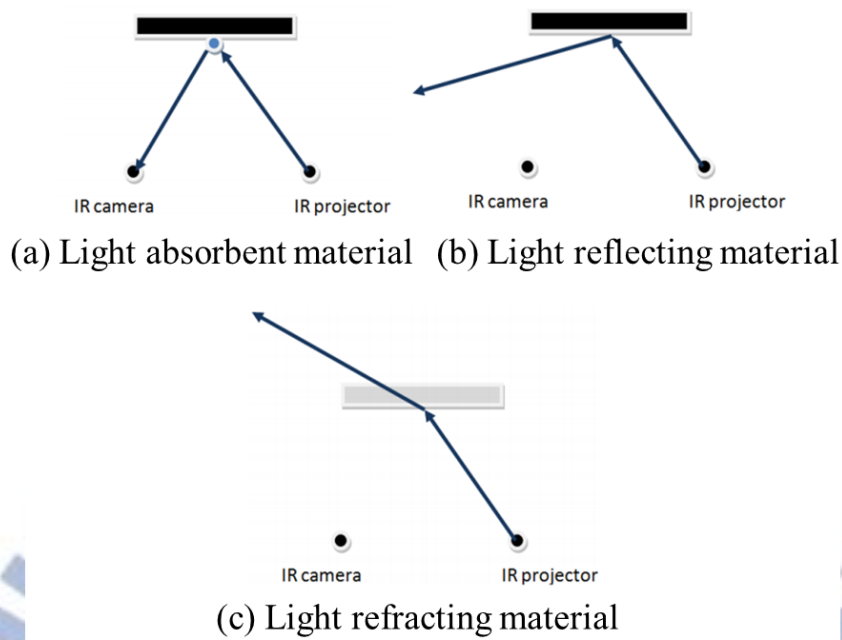


Figure 25: IR light projects on objects made of reflective and refractive materials [22]

Here, we propose a method to fill up these black regions according to the depth distribution of their neighboring regions. The idea of the depth distribution of neighboring regions comes from [16]. This paper finds the neighboring region of black holes in the depth map and computes the histogram to determine the depth value to be filled in the depth holes. In our scheme, we use the same method to find the neighboring region, but our method of filling the black holes is somewhat different from the method in [16].

Before finding the neighboring region, we need to detect those black holes. Usually the intensity value which equals to zero can be viewed as a black hole, so we can convert the depth map into the binary map that 255 stands for the hole. Figure 26 shows the RGB image, the depth map and the binary map of holes.



Figure 26: (a) RGB image (b) depth map (c) binary map of holes

Then we cluster these black holes. Here, we use the 8-connectivity rule to group them. Referring to [23], the definition of an 8-connected component is: A set of white pixels  $P$  in the binary map is an 8-connected component if for every pair of pixels  $p_i$  and  $p_j$  in  $P$ , there exists a sequence of pixel  $p_i, \dots, p_j$  such that

- (a) all pixels in the sequence are in the set  $P$ , i.e., they are all black, and
- (b) every two pixels that are adjacent in the sequence are 8-neighbors

Figure 27 shows the result after completing the 8-connectivity, and we use different colors to represent different groups.

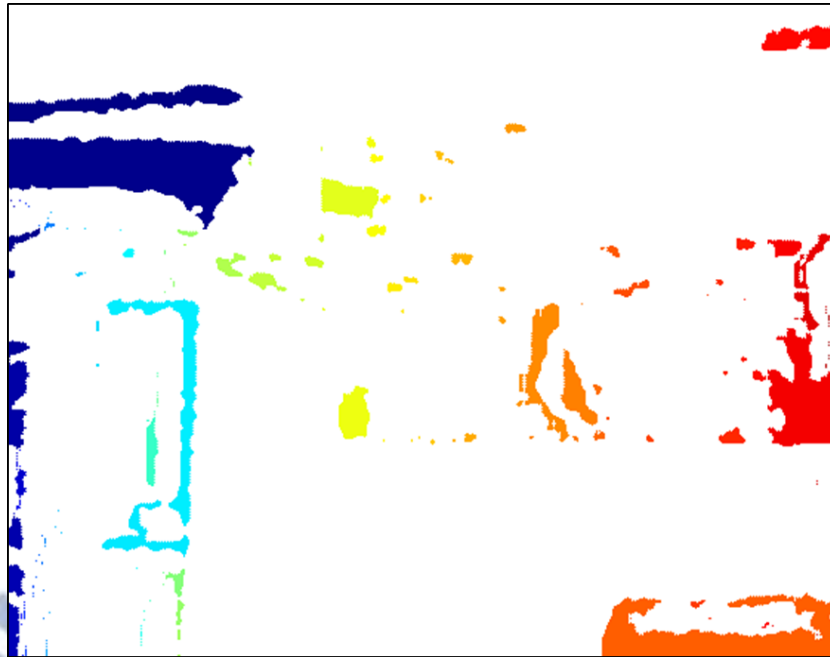


Figure 27: Group of holes

Next, we find the neighboring region of these labeled holes. The idea is to expand the holes to identify their neighbors. The subtraction of the new holes and the original holes is the hole's neighboring region. Dilation, is a basic morphology operation, is used for expansion purpose. The structural element in dilation is  $5 \times 5$  square. Figure 28 shows the neighboring region of black holes.



Figure 28: neighboring region of black holes

From the previous discussions, we assume that the depth values in the neighboring regions are highly correlated with the missing depth value in the holes. Then, we compute the distribution of each neighboring region, and identify the most probable depth value according to the distribution, which is to be used to fill the hole. Our procedure is as follows: we compute the histogram of each neighboring region, and find the dominant peaks in the histogram. The dominant peaks are those peaks having more pixels than a certain percentage of the total pixels in a neighboring region. In [16], they choose the percentage to be 10%, but we find that this value will eliminate too many possible candidates. As a result, we set the threshold as 5% after many experiments. After determining the dominant peaks, [16] eliminates the dominant peaks that are less than the average value of the dominant peaks, and then takes the median of these values as the depth to fill the holes. It means that they tend to fill in the value that is farther to solve the occlusion problem. We use a different method to determine the dominant depth value. Before we talk about our method, we first take a look at the Figure 29. In the picture, we categorize the holes according to how many dominant peaks they have.



Figure 29: Holes are classified based on the number of the amount of dominant peaks. Red means one dominant peak. Green means two dominant peaks. Blue means three dominant peaks. Yellow means four or more dominant peaks.

Observing Figure 29, we can draw some conclusion of the correlation between different kinds of holes and their amount of dominant peaks. The holes that have one or two dominant peaks usually appear inside the objects, and the holes that have three or more dominant peaks usually appear in the occlusion areas or the surfaces that reflect or refract the IR light. As a result, we fill the holes depending on the number of dominant peaks in their neighboring regions:

(a) For one or two dominant peaks: We fill in with the depth value of the highest peak.

This is because most of the holes are on the same object with their neighboring region, and filling in with the most dominant depth is the most appropriate.

(b) For three or more dominant peaks: We have to recognize whether it is an occlusion region or a light-reflecting/ refracting surface first, and then we fill in different value respectively. To

do the classifying, we perform a k-means clustering, and we choose k to be 2. If the difference of the two centroids is larger than a threshold, then we treat it as an occlusion region. On the other hand, if the difference is smaller than a threshold, we say it is a light-reflecting or light-refracting surface. For an occlusion region, we fill in with the smallest depth value from the candidates. For a light-reflecting or light-refracting surface, we fill in with the most dominant peak. Here, we experimentally set the threshold to be 20. The following figure shows our results after our proposed hole-filling algorithm. We also put on the original depth map and the depth map using the hole-filling algorithm in [16]. We can tell that the performance of our method is better than the method in [16], especially in the occlusion region.

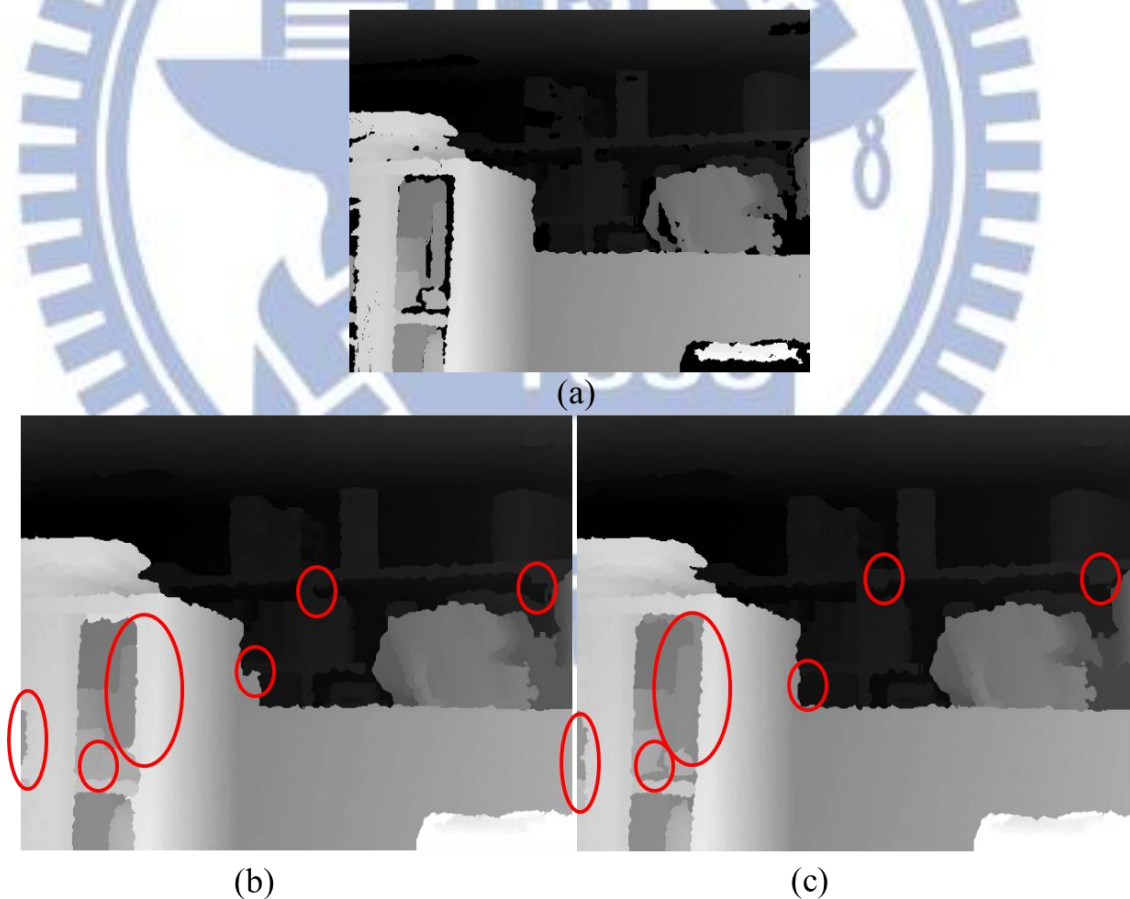


Figure 30: The performance of our proposed hole-filling algorithm (a) the original depth map

(b) the performance of [16] (c) the performance of our method

### 4.4.3 Conversion of Depth to Disparity

The last step of generating a ground truth disparity map is to convert the depth value to the disparity value. According to the Eq. 2 in section 2.3, once we know the depth, the baseline distance and the focal length, and then we can calculate the disparity. Depth value comes from the raw depth map, and the baseline distance is the distance between the right and the left Kinect. By doing the camera calibration, we can get the focal length of the IR camera. Note that the unit of the baseline distance and the depth value is millimeter, and the unit of the disparity and focal length is pixel. Figure 31 shows the results of the conversion of depth value to disparity. In our experiment, the baseline distance which is the distance between the left and right IR camera is 283 mm. The focal length is 525 pixels.

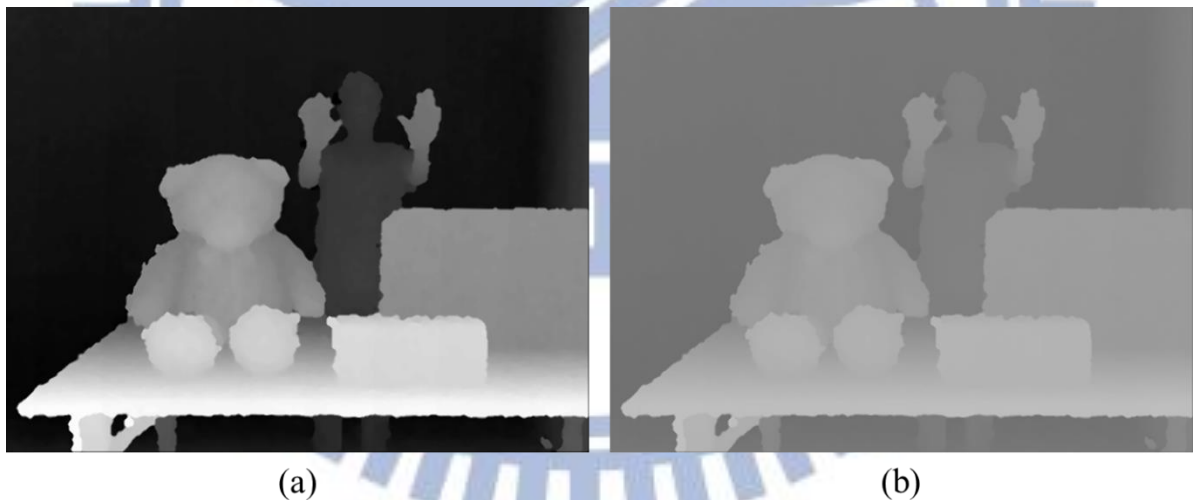


Figure 31: Conversion of depth to disparity. (a) depth map (b) disparity map

## 4.5 Introduction of Dataset and Error metrics



We introduce some evaluation methods on depth estimations in section 2.5. In our study, we construct a dataset consisting of stereo pair images. In all test images there is always a person

in the scene, because these images are designed for the applications of human-computer interaction. Many factors may affect the results of depth estimations. The factors can be categorized into two groups: image quality factor and image content factor. We design test images that include those factors. We will give detailed descriptions and display these images in the next two sub-sections. In 4.5.3, we discuss how to use this dataset to evaluate the depth evaluation.

#### 4.5.1 Image Content Factor

In the image content factor category, we want to check what kind of the content will influence the stereo matching algorithms. The factors include background complexity, numbers of objects, hand pose and clothing.

There are two parts in background complexity. Part one is the images of complex background and textureless background. Part two is the images of textureless background and the image with reduced textureless background. In part one, we first capture pictures whose background is nearly textureless, and then we add two complex textures posters into the background: repeated texture and non-repeated texture. They help us to see if the repeated texture will interfere the depth estimation process as reported before. Figure 32 shows the pictures in part one.

	Left Image	Right Image
Textureless Background		



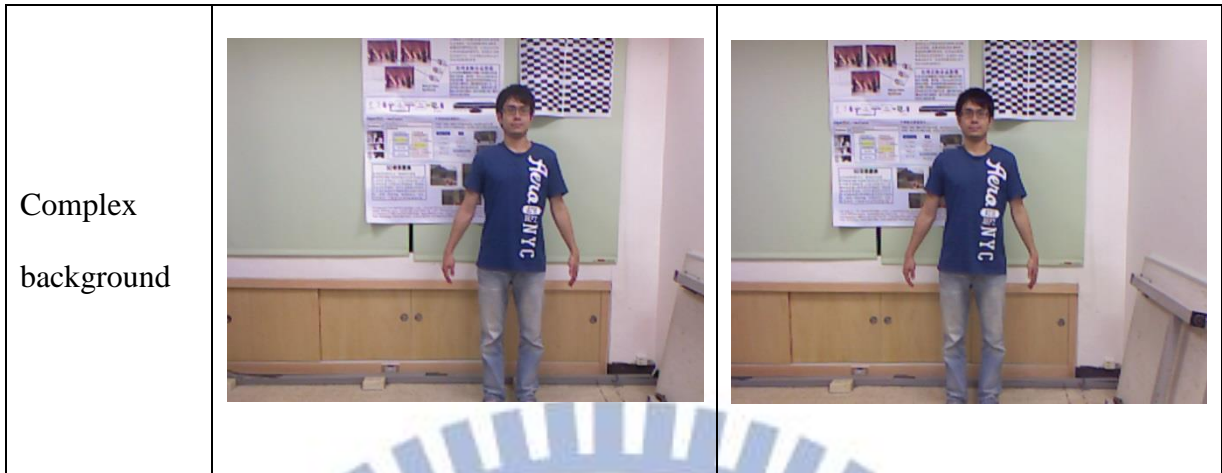


Figure 32: Stereopairs of textureless background and complex background

In part two, we want to check if cutting off the textureless region can help improve the accuracy of depth estimation algorithm. Figure 33 shows the pictures in part two. The first “cut” removes the left  $\frac{1}{3}$  background. The second “cut” further removes the upper strip of the background.

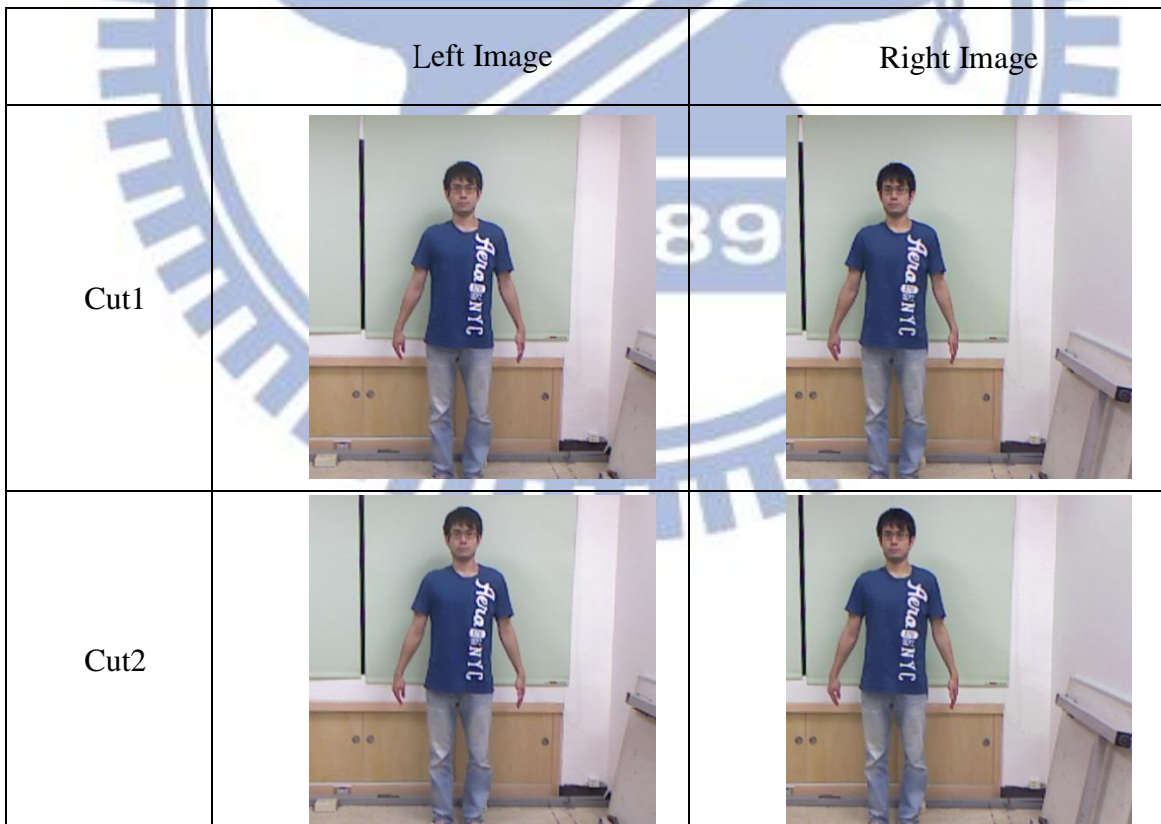
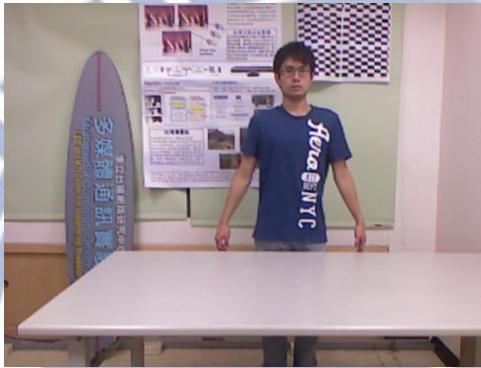
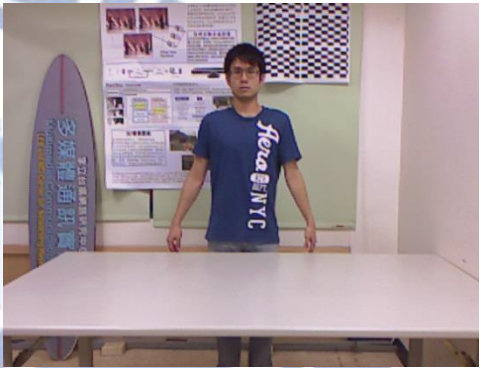




Figure 33: Stereopairs of different cut

Next, we want to know if the number of the objects will affect the performance of depth estimation. First set of images simply have a person and a table, and then we add in a bear doll, a checkerboard and a box in order. Note that all of these images are with complex background. We additionally take the pictures that contain all objects in the simple background. So, we have five sets of images, and Figure 34 shows all of them:

	Left Image	Right Image
Number of objects=0		
Number of objects=1		







<p>Number of objects=2</p>		
<p>Number of objects=3 (complex background)</p>		
<p>Number of objects=3 (simple background)</p>		

Figure 34: Pictures of different number of objects in the scene

The hand pose may affect the result of the depth estimation algorithm. If we raise our arms horizontally, finding the corresponding pixel in the hand area may be more difficult than putting our hands down. So we take a set of the pictures with arms up horizontally and with

hands down. Figure 35 shows these pictures.

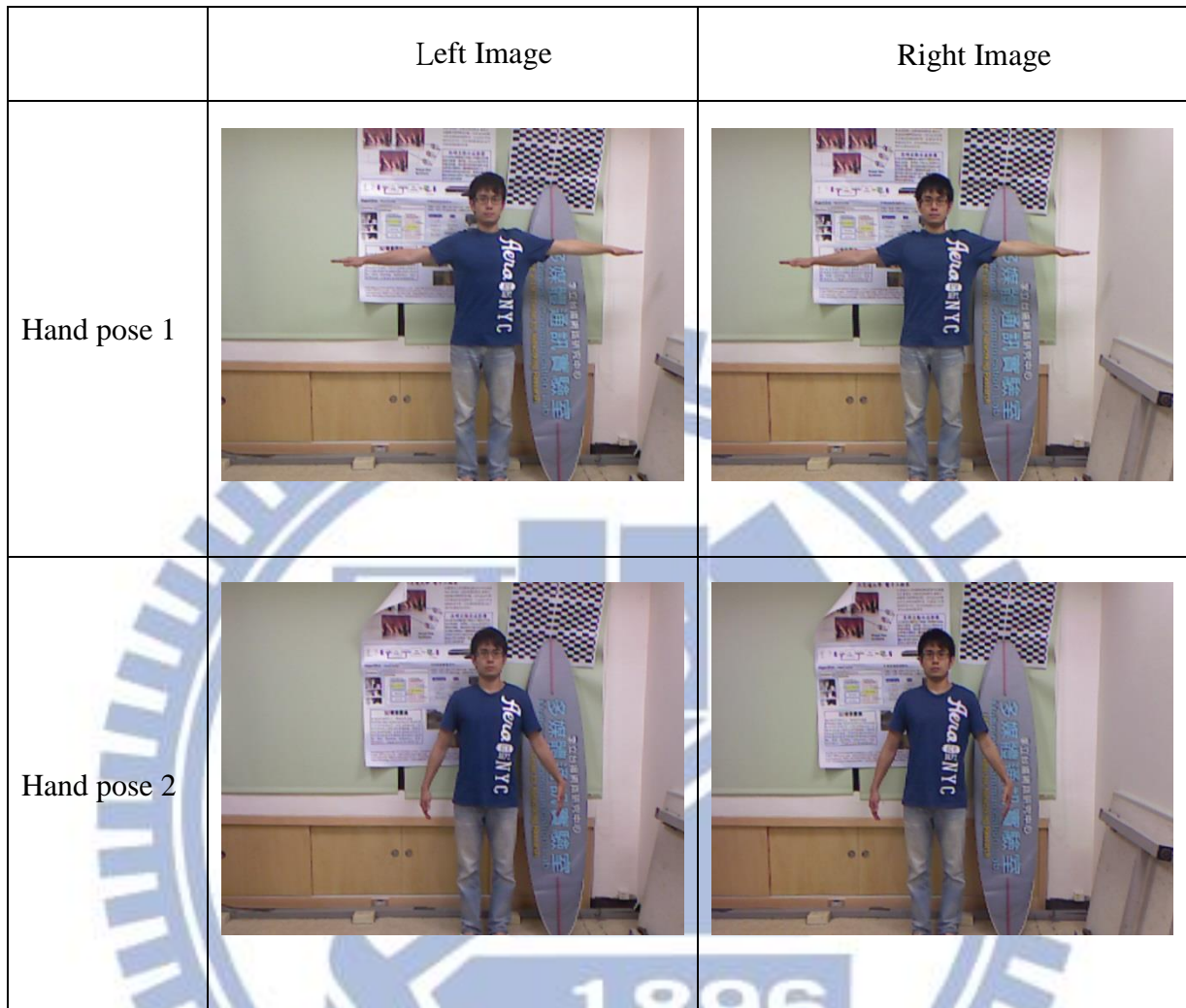


Figure 35: Pictures with different hand poses

Some applications extract the foreground based on the depth map. Hence, the estimated disparity in the foreground is more important than in the background. When the foreground is a person, we want to know whether the patterns on the clothes will influence the depth estimation algorithms. As a result, we take a set of pictures in which the person wears clothes with three different patterns. There are clothes with English alphabet, T-shirts with unicolor plaid pattern and T-shirts with multicolor plaid pattern. Figure 35 shows the T-shirt with letters. Figure 36 shows the other two patterns.

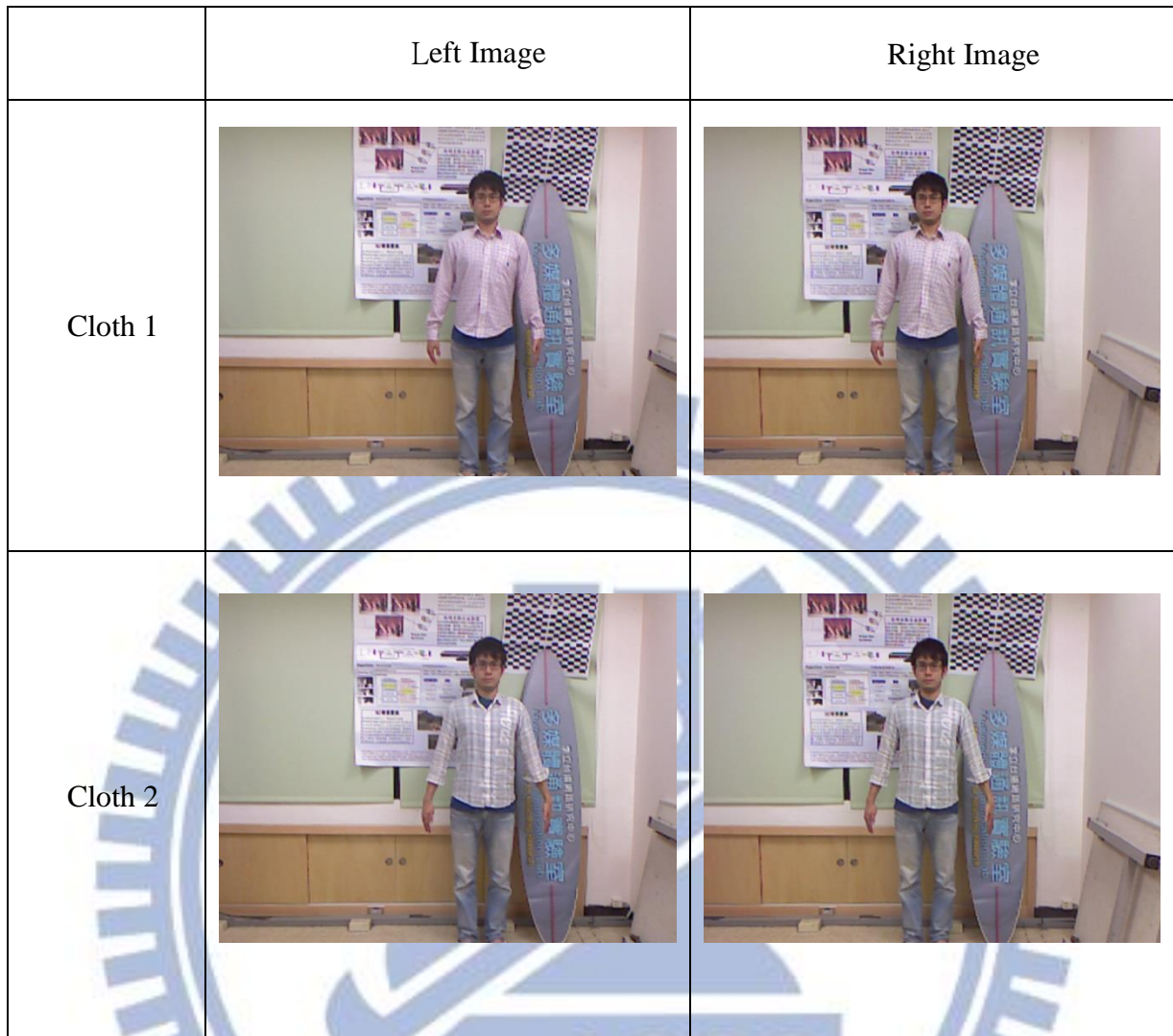


Figure 36: A person wears clothes with different patterns

#### 4.5.2 Image Quality Factor

In image quality factor category, we want to see how the image quality such as noise would affect the depth estimation algorithms. The factors include Gaussian noise and rectification error.

We add Gaussian noise of different variance to the simple background image. The noise level is labeled by PSNR. The independent Gaussian noise is added to the R, G, and B color components separately. With some pre-experiment, we pick up three levels of PSNR: 30, 35 and 40dB. In Figure 37, we show our images of different PSNR levels.







	Left Image	Right Image
PSNR=30		
PSNR=35		
PSNR=40		

Figure 37: Images with different PSNR

Next, we introduce rectification error into images. Rectification error means the corresponding pixels are not in the same horizontal line of the right and the left images, which

may cause some problem in stereo matching. Here we have two parts. First part is to compare the original images from Kinect and the rectified images using our rectification method.

Figure 38 shows the pictures.

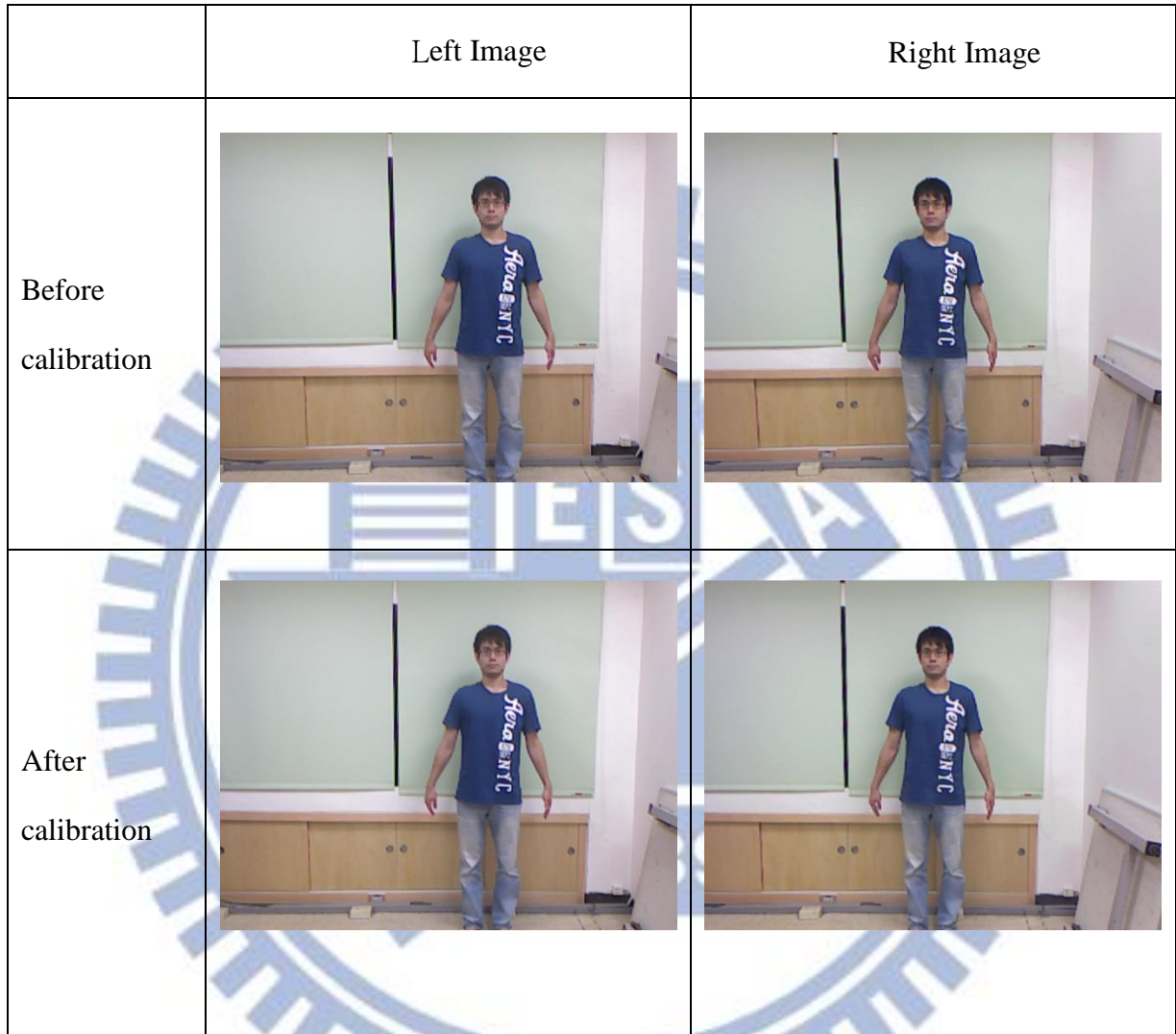
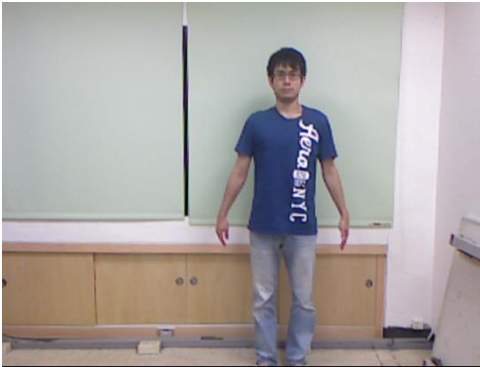







Figure 38: Unrectified and rectified images

In part two, if one rectified picture is artificially shifted by one to five lines with respect to the other picture, how the shift affect depth estimation algorithms. The idea is to simulate rectification errors. We assume that our rectification method is perfect. Pixels should be in the same horizontal line. And then we move the left image upward 1 line to 5 lines to simulate the shifted rectification errors. Figure 39 shows the images with shifts.

	Left Image	Right Image
1-line recification error		
2-line recification error		
3-line recification error		



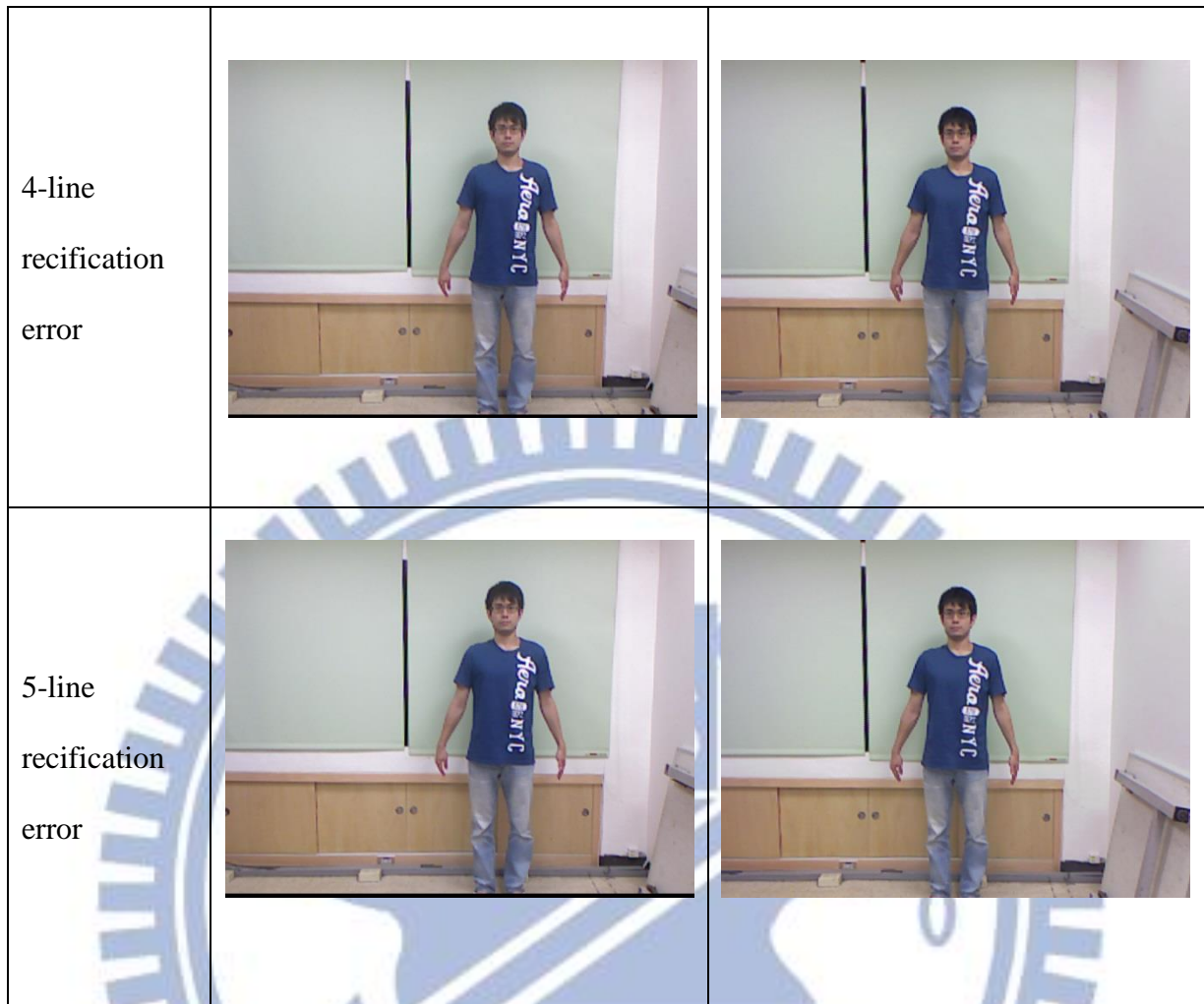


Figure 39: Image with rectification error of 1-5 lines

### 4.5.3 Error Metrics

In this study, we want to evaluate the stereo matching algorithm for the use of human-computer interaction applications. These applications do not need very accurate depth map. Often, extracting the foreground successfully from the background is more important. Consequently, we evaluate the disparity map mostly in the foreground and background areas, but the edge and the occlusion regions are ignored. Here we adopt the trimap concept in doing the evaluation. Trimap consists of three regions: a definite foreground, a definite background and a blended region, which is in-between the foreground and the background regions. In a trimap, we mark the definite foreground region by 255, and the definite background region by

0. The map value of the blended region is 125. Figure 40 shows an example of the trimap.



Figure 40: RGB image and its trimap

According to the trimap, we identify the region we want to evaluate. Due to the disparity map errors around the object boundaries, we treat blended regions as “unknowns”. We generate a trimap by the following steps:

- (a) We extract the foreground using a threshold from the depth map. The threshold is determined experimentally.
- (b) Dilating the original foreground in a depth map, we get the dilated foreground. The region which is outside dilated foreground is the definite background. And then we erode the original foreground to get the eroded foreground. The region which is inside the eroded foreground is the definite foreground. The blended region will be the subtraction of dilated foreground and eroded foreground. The morphological structural element we use is  $5 \times 5$  square.

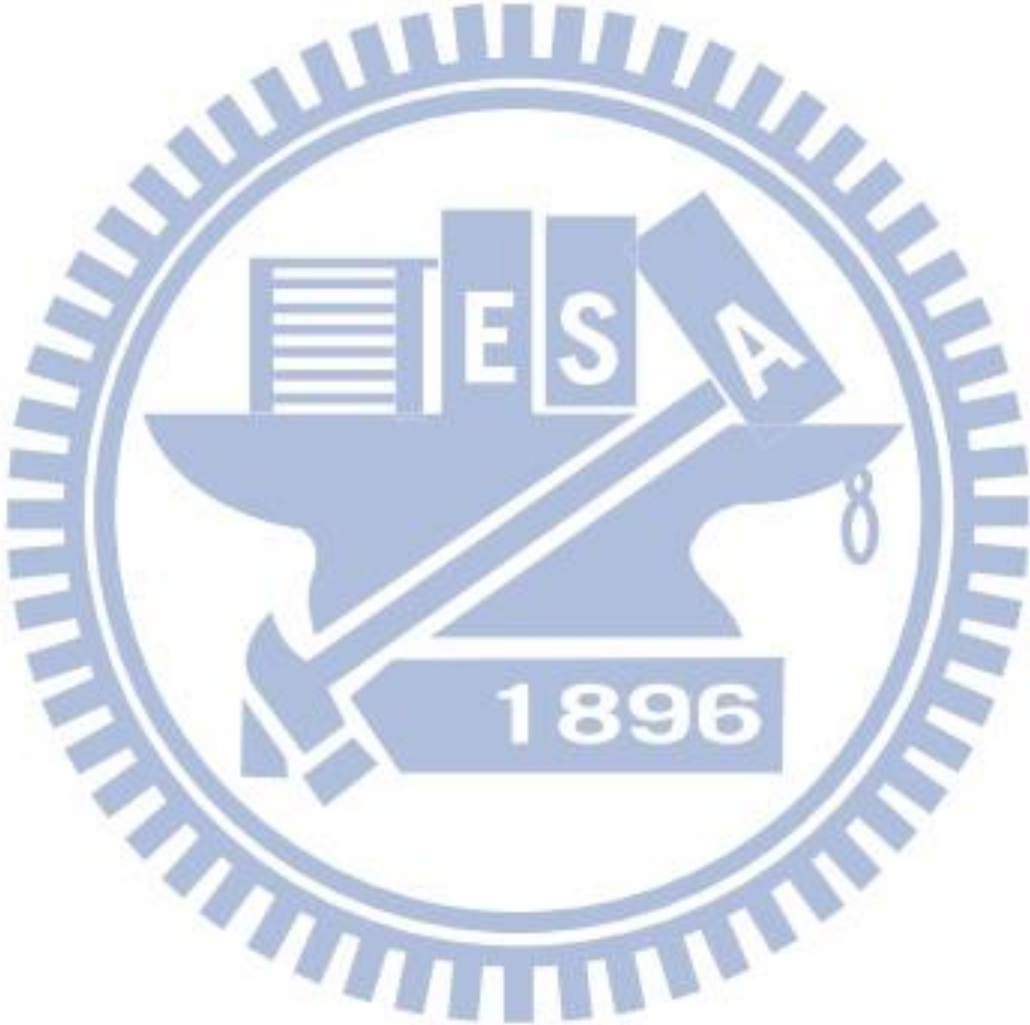
After determining the evaluating regions, we select the quality metrics. In this study, we use two metrics: mean squared value (MSE) and bad matching pixel rate (BPR). Assume that the computed disparity map is  $d_C(u, v)$  and the ground truth disparity map is  $d_{GT}(u, v)$ . MSE is defined as the following equation

$$\text{MSE} = \frac{1}{N} \sum_{(u, v) \in \text{evaluation region}} |d_C(u, v) - d_{GT}(u, v)|^2, \quad (20)$$

where  $N$  is the total number of pixels in the evaluation region. BPR is defined as follows.

$$\text{BPR} = \frac{1}{N} \sum_{(u,v) \in \text{evaluation region}} (|d_C(u,v) - d_{GT}(u,v)| > \delta_d), \quad (21)$$

where  $\delta_d$  is called disparity error tolerance. In this thesis, we use  $\delta_d = 10$ . If we choose a small value, there will be too many errors, which will make it harder to analyze the evaluation results. We choose 10 through many experiments.



## chapter 5 Experimental Results and Discussion

In this chapter, we evaluate three stereo matching algorithms using our dataset introduced in chapter 4, and show the estimated disparity map and the statistical results.

### 5.1 Experimental Environment

We use two Kinects for XBOX 360 to take the images of left and right view. They are set in parallel as Figure 41 shows.



Figure 41: Experiment set up

In the Figure 41, there is a halogen light besides the Kinects, and it is used to do the camera calibration for IR camera. We use Kinect-v1.0-beta2 and OPENNI to extract the raw RGB image, the raw depth map and the IR image.

## 5.2 Three Stereo Matching Algorithms

In this section, we briefly described the three stereo matching algorithms used in our experiments. They are the non-local cost aggregation method for stereo matching [24], the stereo matching with nonparametric smoothness priors in feature space [25] and the depth estimation reference software (DERS) [26]. They are chosen to represent different types of disparity estimation algorithms.

In [24], the authors propose a non-local cost aggregation method for stereo matching. The basic flow is the same as in the introduction in section 2.3. They use winner-take-all to find the disparity, so their method can finish in about 90 milliseconds when applied to the Middlebury dataset. The feature of this paper is that they use a non-local cost aggregation method instead of aggregating in a fixed local window. A tree structure is used to aggregate the cost function. The nodes in the tree are all the pixels in the image, and the tree branches between the two nearest neighboring pixels have a value representing their similarity. The paper then branches the edges with largest dissimilarities, and thus they get the minimum spanning tree (MST). MST can be seen as a natural image pixel similarity measurement. Aggregation operation is done with all the nodes in the MST, so it is called a non-local cost aggregation method.

DERS is developed by the 3D video Coding Team of Moving Pictures Experts Group (MPEG) for video (3DV) and free viewpoint Television (FTV). DERS needs three input views: left, central and right view. By computing the cost functions from left to central view and right to central view, DERS chooses the smaller cost out of them to improve the disparity accuracy in the occlusion areas. To compare the DERS with the other stereo matching algorithms which only use two views, we modify the software so that it computes the disparity by using two views. There are three modes that we can choose in DERS: automatic

mode, segmentation mode and semi-automatic mode. In this thesis, we use the automatic mode, and the following figure shows its operation.

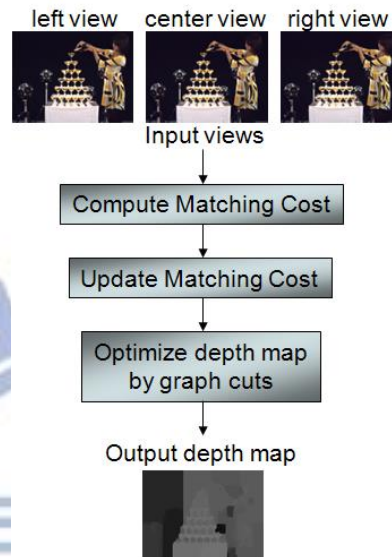


Figure 42: Flow diagram of the automatic mode [26]

The basic flow is nearly identical to our introduction in section 2.3. The matching block here is  $3 \times 3$ . The cost energy function containing a similarity term and a smoothing term, which are calculated at every pixel, and then the graph-cuts algorithm is used for the energy minimization.

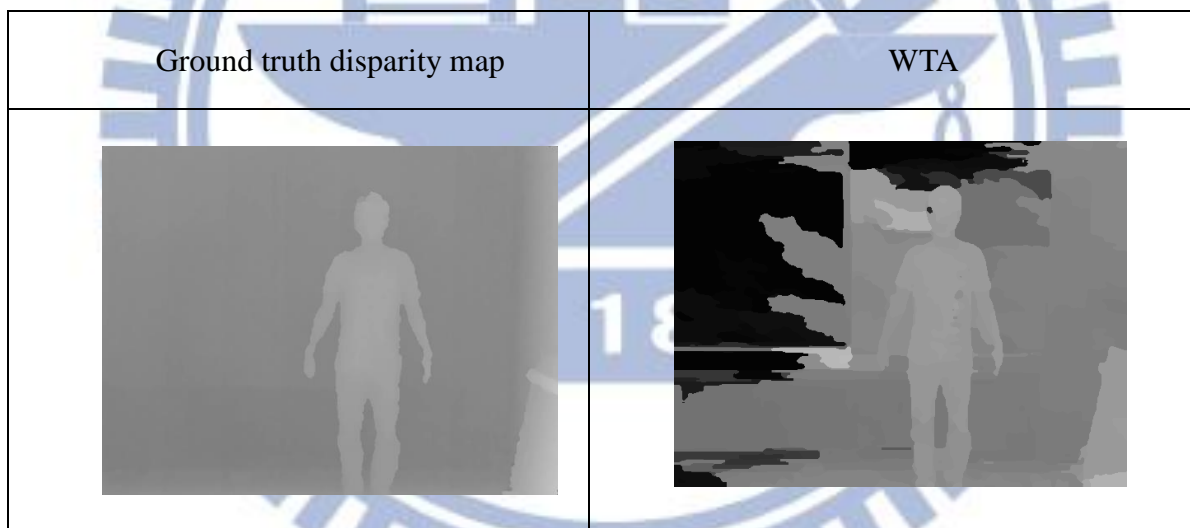
In [25], a stereo matching algorithm with nonparametric smoothness priors in feature space is proposed. They treat every pixel as a feature vector containing information of R, G, B value and location. They build a nonparametric depth smoothness model in this feature space that combining the features and depth values. The pixels with similar features can be connected by the model, and become a connected network. This network is just like a neighborhood system that grouping the similar pixels together without performing image segmentation. The Graph-cuts algorithm is applied to the neighborhood system to find the optimal disparity.

## 5.3 Experimental Results on Image Content Factors

In this section, we show the evaluation results using our stereo pair images with different image content factors. First, we show the computed disparity and their ground truth disparity map, and then we compute and show the bad-matching pixel rate and MSE. Here, we call the stereo matching using non-local aggregation method, Winner-take-all method (WTA) because of its optimization, and we call stereo matching with nonparametric smoothness priors in feature space method graph-cuts in neighborhood system (GC-NS).

### 5.3.1 Background Complexity

We first show the disparity maps that use images of textureless background.



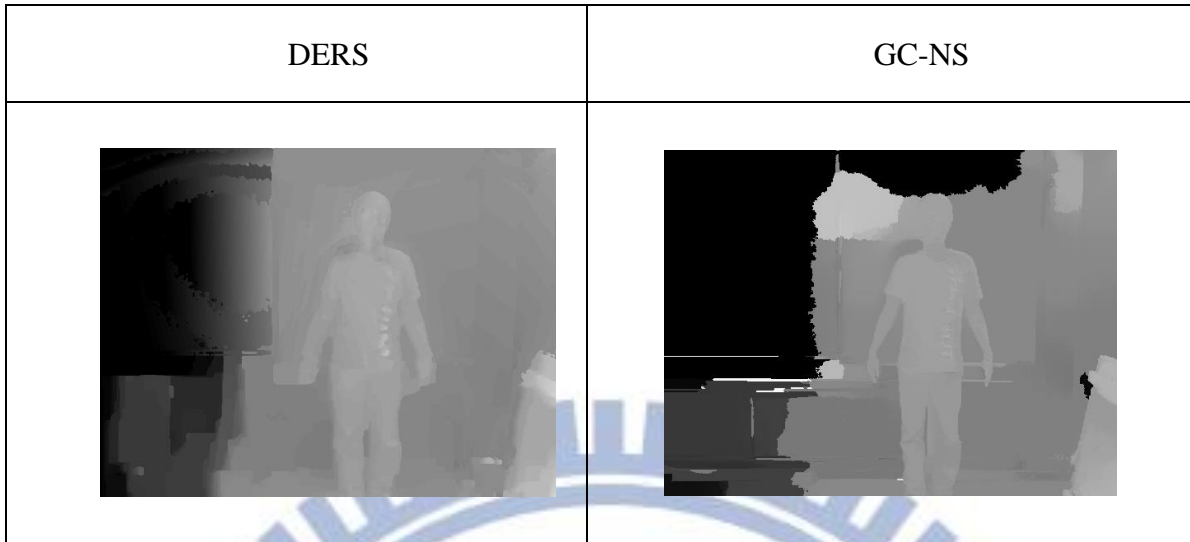


Figure 43: Disparity maps on the simple background test images

Figure 44 shows the disparity maps on the complex background test images (Figure 32).

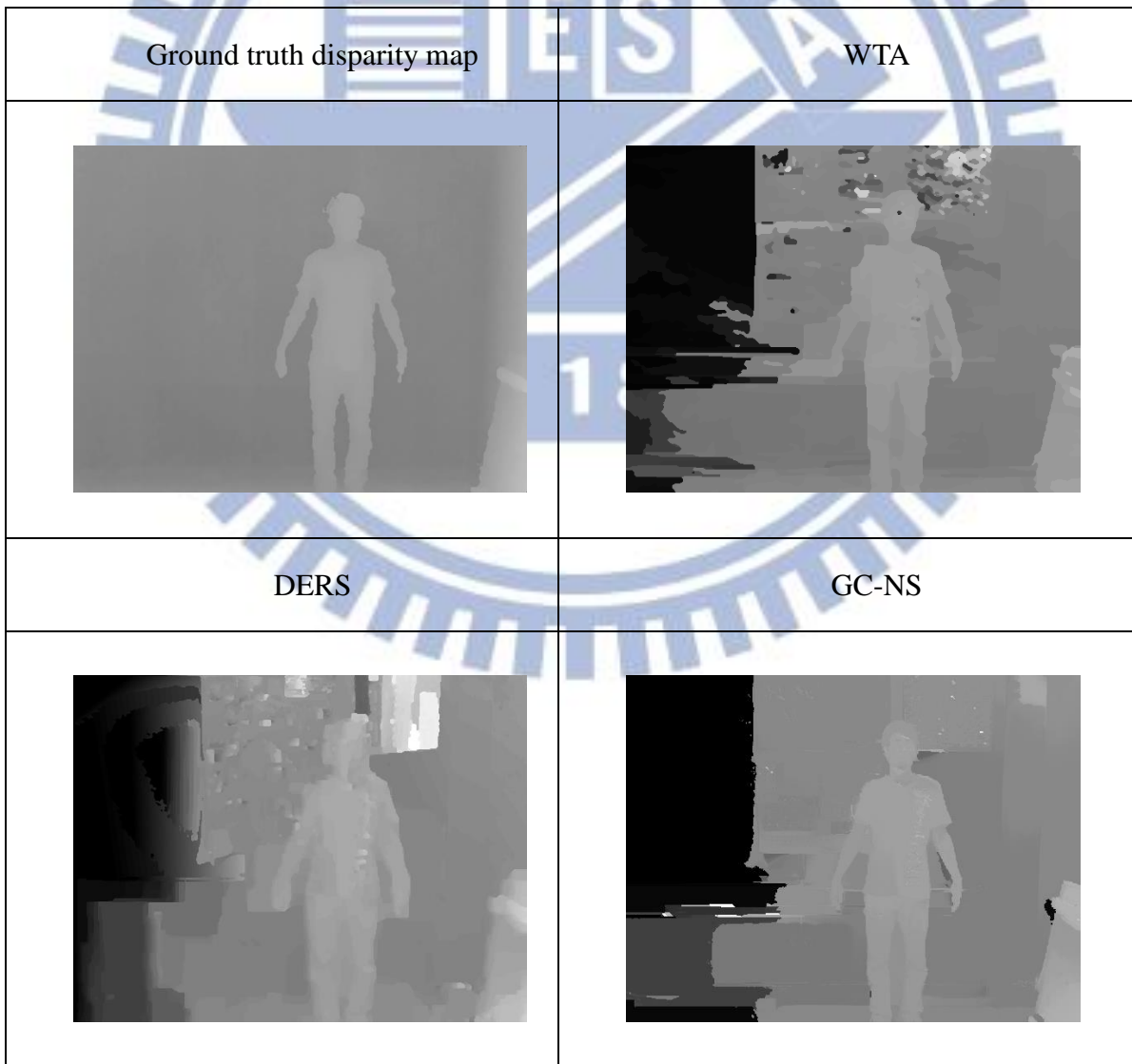




Figure 44: Disparity on complex background test images

Table 2 and Table 3 compare BPR and MSE of all methods on these test images. Note that the error metrics are calculated only on the foreground region or only on the background region, which are marked in the “Region” column (as explained in sec. 4.5).

Table 2: BPR of disparity maps on simple and complex background test images

	Region	WTA	DERS	GC-NS
simple	foreground	0.1065	0.0180	0.0162
	background	0.6611	0.4627	0.5693
complex	foreground	0.0115	0.0735	0.0162
	background	0.4692	0.4571	0.4035


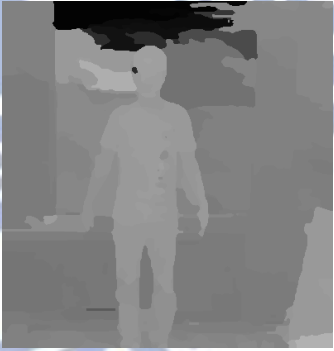
Table 3: MSE of disparity map on simple and complex background test images

	Region	WTA	DERS	GC-NS
simple	foreground	100.99	16.39	24.88
	background	5234.6	4258.8	6839.5
complex	foreground	38.81	49.22	16.86
	background	4087.3	3476.3	4688.3

Examining the above images and tables, we have a few observation. For WTA, its BPR and MSE in foreground and background both decrease from the simple background image to the complex background image. For DERS, BPR and MSE decrease from the simple background to complex in background, but increase in foreground. For GC-NS, BPR shows no increasing and decreasing in foreground from simple to complex background, but BPR in background and MSE in both foreground and background decrease. In the following discussions, we focus on the foreground. When the background is simple, WTA has the worst result in BPR. However, when the simple background changes to the complex background, the result of DERS becomes worse than that of WTA. WTA improves estimation results the most among the three methods from the simple to the complex background.

The design principles of these disparity estimating algorithms may help explaining their behaviors. Although WTA and GC-NS do not use image segmentation explicitly, their grouping method is very similar to that of image segmentation. As a result, the edge can be preserve better, and the estimated disparity in the foreground can also benefit from it. For DERS, it does not use any concept of grouping, so the foreground get worse when the background become complex.

Next, we want to see whether the results will be better if we cut off the textureless regions. Figure 45 shows that we first cut off the left part of image, and we call it CUT1 (Figure 33). Figure 46 are the images cutting off the left and top parts, and we call it CUT2 (Figure 33). Table 4 and Table 5 are the comparisons of full background, CUT1 and CUT2 in BPR and MSE.

Ground truth disparity map	WTA
	

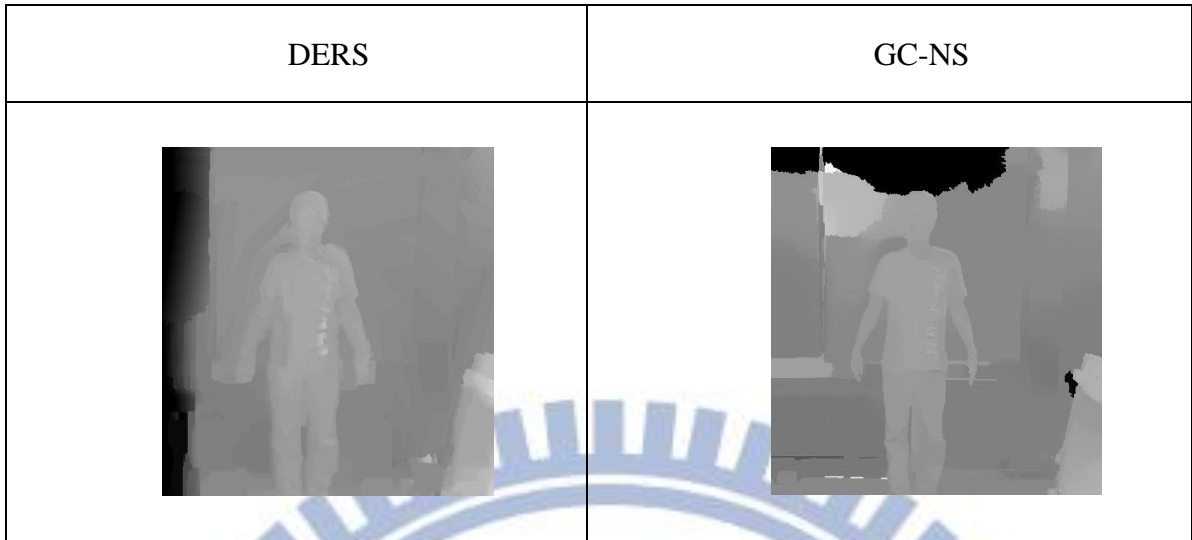


Figure 45: Disparity maps on CUT1 images

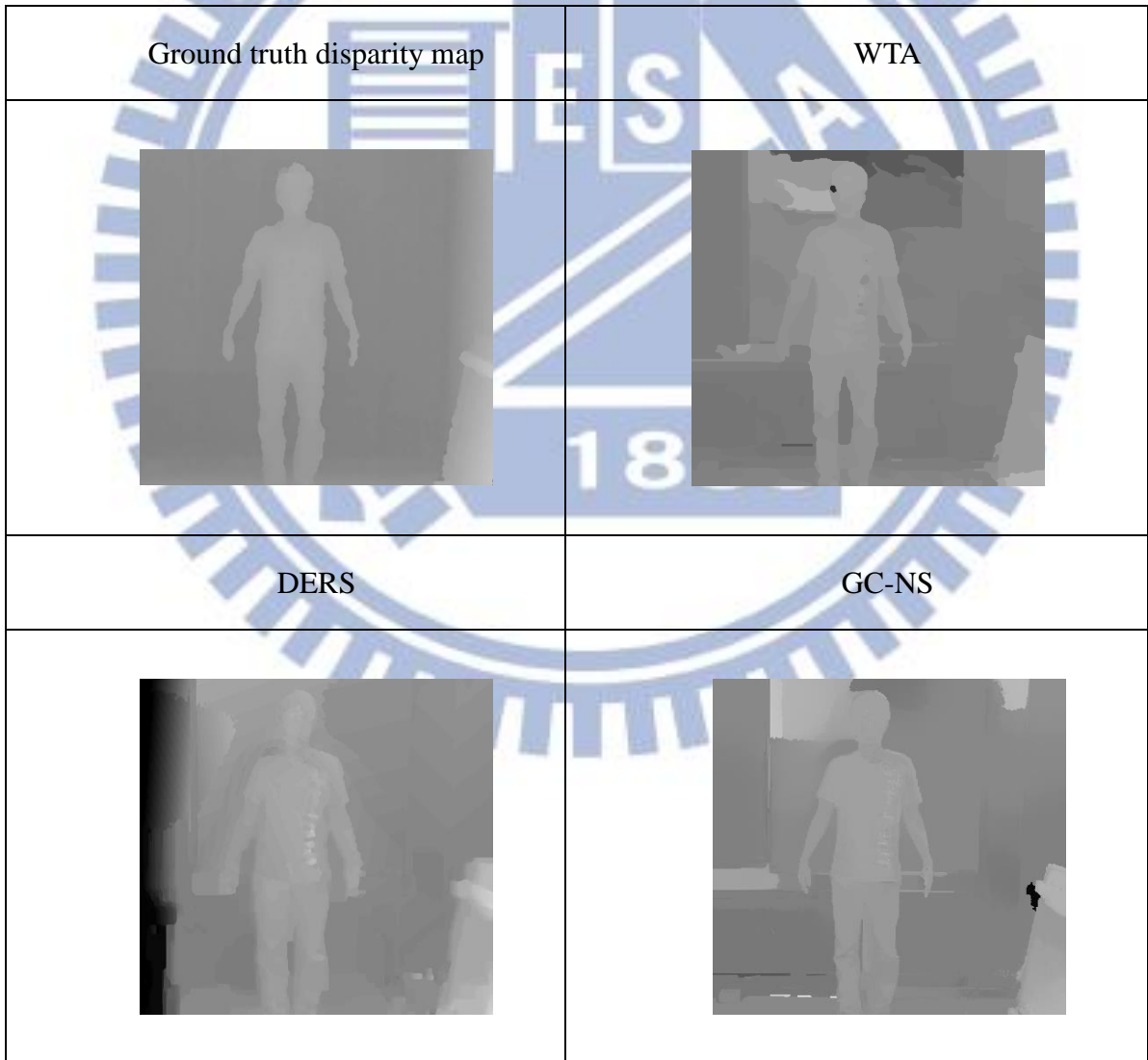


Figure 46: Disparity maps on CUT2 images

Table 4: Compison of BPR of computed disparity maps on different size of background

Type	Region	WTA	DERS	GC-NS
FULL	foreground	0.1065	0.0180	0.0162
	background	0.6611	0.4627	0.5693
CUT1	foreground	0.1065	0.0213	0.0160
	background	0.5071	0.2448	0.3066
CUT2	foreground	0.1065	0.0216	0.0160
	background	0.4916	0.3293	0.2455

Table 5: Compison of MSE of computed disparity maps on different size of background

Type	Region	WTA	DERS	GC-NS
FULL	foreground	100.99	16.39	24.88
	background	5234.6	4258.8	6839.5
CUT1	foreground	100.98	16.47	24.95
	background	297.6909	1444.7	232.48
CUT2	foreground	100.98	17.13	24.95
	background	1382.7	1561.8	1957.7

For WTA, cutting off textureless background does not change the disparity results in the foreground, but it helps to decrease the BPR and MSE in the background. The black regions in the depth maps become less in CUT1 and CUT2. For DERS, there is an obvious improvement in the background in BPR and MSE when the full size background changes is reduced to CUT1. There are still some errors on the left side. This is because the pixels on the picture left border of the left image have no corresponding pixels on the right image. For GC-NS, the accuracy improves the most among the three methods in the background when the full size background is reduced to CUT2. For all of these three methods, cutting off the textureless does not affect the foreground much.

We thus conclude that cutting off the textureless region can improve the performance of stereo matching algorithms as we expect. We found that estimation error of DERS on the left side of image is caused not by both the textureless region and the missing part on the right

view. But why we do not see this phenomenon in the other two methods? The reason is that the two methods do cross checking between left and right estimated disparity. In fact, DERS does cross checking by using three input images, but our modification uses only two input images.

Next, we look at repeated-pattern background. Tables 6~9 show BPR and MSE comparison between the repeated-pattern background and the textureless background, and between the complex and the textureless background.

WTA has better disparity estimation results when the texture is full of repeated pattern or the texture is complex. The BPR and MSE of DERS indicate that it cannot handle the repeated-pattern area well. But when the texture is complex, DERS performs better than when the texture is simple. The performance of GC-NS, the performance improves when the background contains repeated pattern or complex pattern.

WTA and GC-NS use adaptive windowing in their grouping methods. In contrast, DERS uses a fixed window. We believe this explains why DERS has poor performance in the repeated-pattern area.

Table 6: BPR focusing on the repeated-pattern area

Region	Type	WTA	DERS	GC-NS
Foreground	Repeated pattern	0.5341	0.9068	0.0285
	textureless	0.8650	0.0108	0.5019

Table 7: MSE focusing on the repeated-pattern area

Region	Type	WTA	DERS	GC-NS
Foreground	Repeated pattern	1053.2	4645.5	51.46
	textureless	6545.7	16.29	9838.6

Table 8: BPR focusing on the complex area

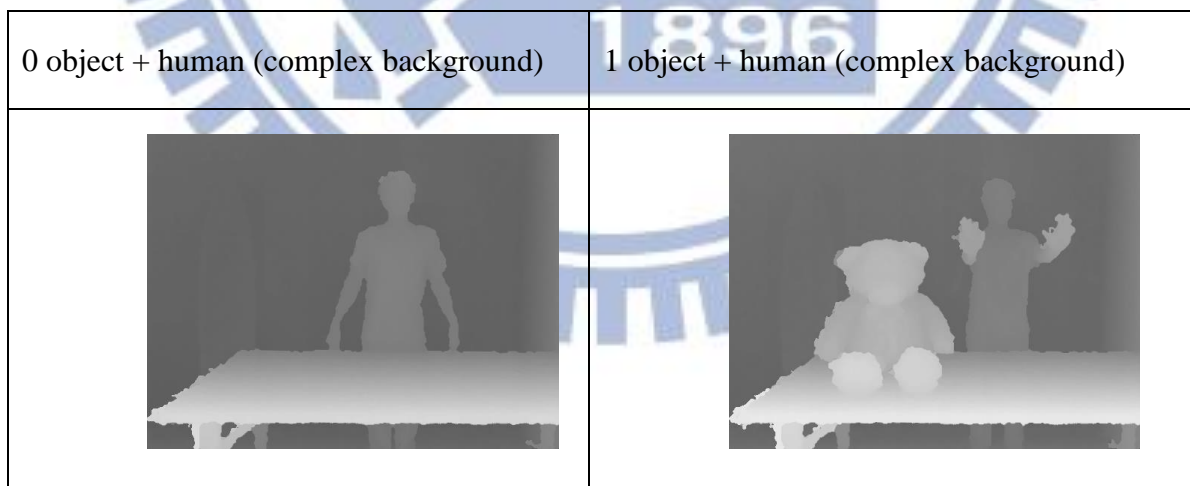
Region	Type	WTA	DERS	GC-NS
Foreground	Complex	0.2006	0.1565	0.0169
	simple	0.7323	0.3736	0.6693

Table 9: MSE focusing on the complex area

Region	Type	WTA	DERS	GC-NS
Foreground	Complex	465.34	136.57	16.50
	simple	6521.8	703.74	8172.2

### 5.3.2 Different numbers of Objects

In this section, our focus is the number of objects if this factor affects the performance of stereo matching algorithms. We start with a person and a table in the scene, and then we add 3 objects gradually (Figure 34). Note that here we only focus on the estimated disparity value of the person.






2 object + human (complex background)	3 object + human (complex background)
	
3 object + human (simple background)	
	

Figure 47: The ground truth disparity maps

Also, we show the trimaps of these cases.

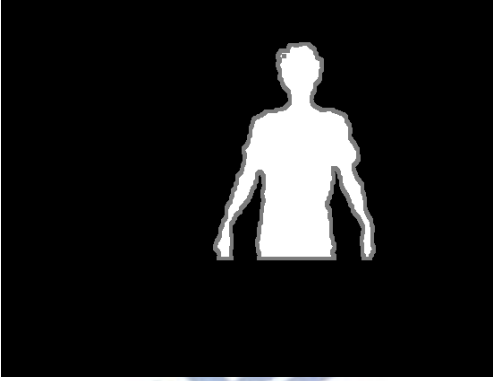
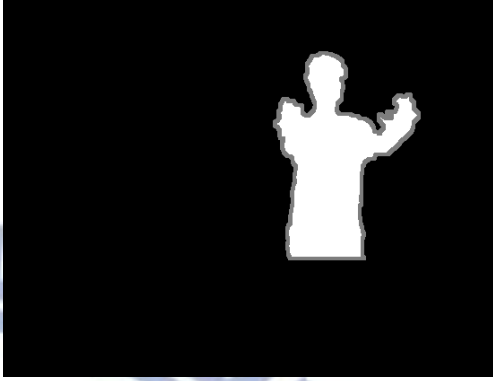



0 object + human (complex background)	1 object + human (complex background)
	
2 object + human (complex background)	3 object + human (complex background)
	
3 object + human (simple background)	
	

Figure 48: The trimaps in this evaluation








0 object + human (complex background)	1 object + human (complex background)
	
2 object + human (complex background)	3 object + human (complex background)
	
3 object + human (simple background)	
	

Figure 49: Computed disparity map using WTA


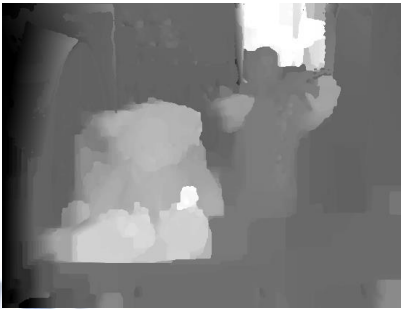



0 object + human (complex background)	1 object + human (complex background)
	
2 object + human (complex background)	3 object + human (complex background)
	
3 object + human (simple background)	
	

Figure 50: Computed disparity map using DERS






0 object + human (complex background)	1 object + human (complex background)
	
2 object + human (complex background)	3 object + human (complex background)
	
3 object + human (simple background)	
	

Figure 51: Computed disparity map using GC-NS

Table 10: BPR of computed disparity maps using images consisting of different amount of objects in the complex background

Type	Region	WTA	DERS	GC-NS
0 object	human	0.9674	0.0185	0.0193
1 objects	human	0.8821	0.0955	0.0412
2 objects	human	0.9742	0.1961	0.0457
3 objects	human	0.9575	0.1857	0.0849

Table 11: BPR of computed disparity maps using images consisting of 3 objects in the simple

background

Type	Region	WTA	DERS	GC-NS
3 object	human	0.9837	0.2280	0.0870

All of the results of WTA are poor for these test images. We thus skip them. The results of DERS become worse when we increase objects in the scene, but the third object which is the closest to the camera does not worsen the results of the human depth map. When the complex background is replaced by the simple background, the result does not change too much. The results of GC-NS also get worse when the object number increases, though the second object has little impact on the results. Whether it is a complex background or a simple background, the results remain about the same.

Among these three methods, the performance of GC-NS is the best, and the performance of WTA is the worst. The complexity of the algorithms may explain the results. Compare DERS with GC-NS, the behaviors are similar. But GC-NS is better at edge preservation because it uses an adaptive grouping method.

### 5.3.3 Hand Poses

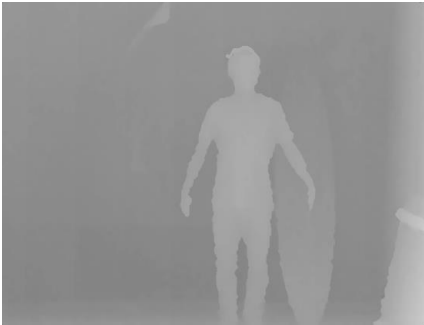





Ground truth disparity map	WTA
	
DERS	GC-NS
	

Figure 52: Disparity maps on hadns down

Ground truth disparity map	WTA
	

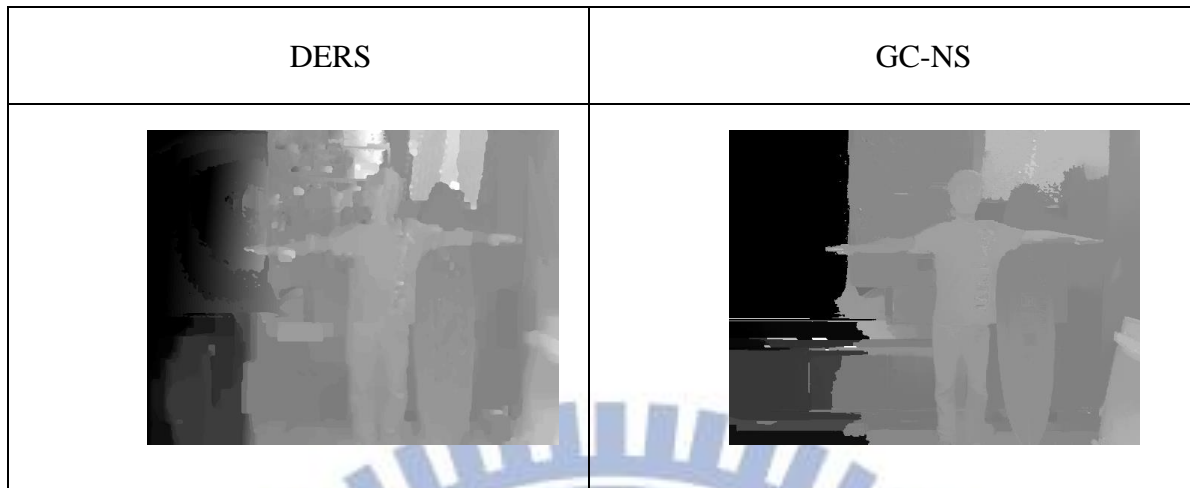


Figure 53: Disparity maps on horizontal arms

Table 12: BPR of computed diparity maps with vertical arms

Type	Region	WTA	GC	GC-NS
Foreground	Vertical pose	0.7808	0.0326	0.1621
	Horizontal pose	0.8996	0.0585	0.3367

Table 13: MSE of computed diparity maps with horizontal arms

Type	Region	WTA	GC	GC-NS
Foreground	Vertical pose	154.28	21.25	76.18
	Horizontal pose	196.06	32.26	97.70

When we put our arms horizontally, finding corresponding pixels may be more difficult because there are similar skin color pixels in the path of finding correspondence. Tables 12 & 13 show the results that agree to our speculation. On the foreground region, all of the BPR and MSE of all three methods increase when the arm position is changed from vertical to horizontal. To check whether the skin color region increases the errors, we show the computed disparity maps (Figure 54) from DERS and GC-NS, where we use red color to mark the bad matching pixels. We do not show the results from WTA because its errors are too high to tell the difference. From Figure 54, we can clearly see the error increase in skin color region.

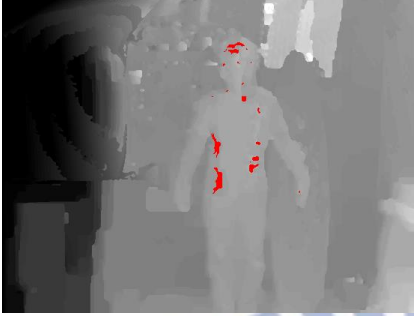
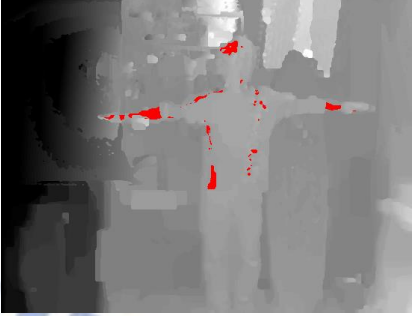


DERS (vertical hand pose)	DERS (horizontal hand pose)
	
GC-NS (vertical hand pose)	GC-NS (horizontal hand pose)
	

Figure 54: Disparity maps of two different hand poses with bad matching pixels marked on them as red

### 5.3.4 People in Different Clothes

In this section, we show the experimental results of the images with different clothes. As discussed earlier in sec. 4.5, three types of patterns on clothing are tested: T-shirts with letters, with multicolor plaid pattern and with unicolor plaid pattern. The results of T-shirt with letters are from

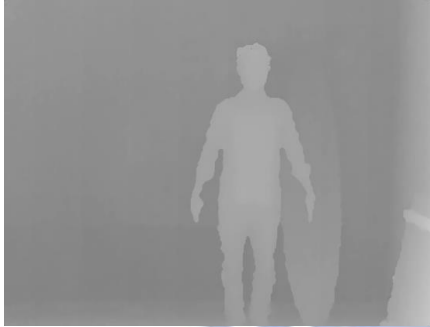



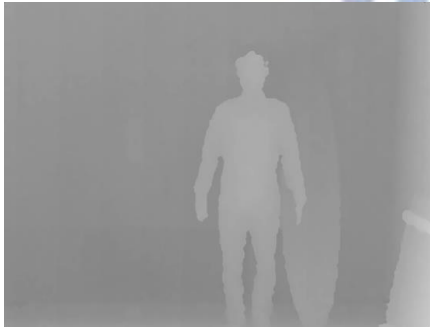

Ground truth disparity map	WTA
	
DERS	GC-NS
	

Figure 55: Disparity maps on images of multicolor plaid pattern T-shirt

Ground truth disparity map	WTA
	



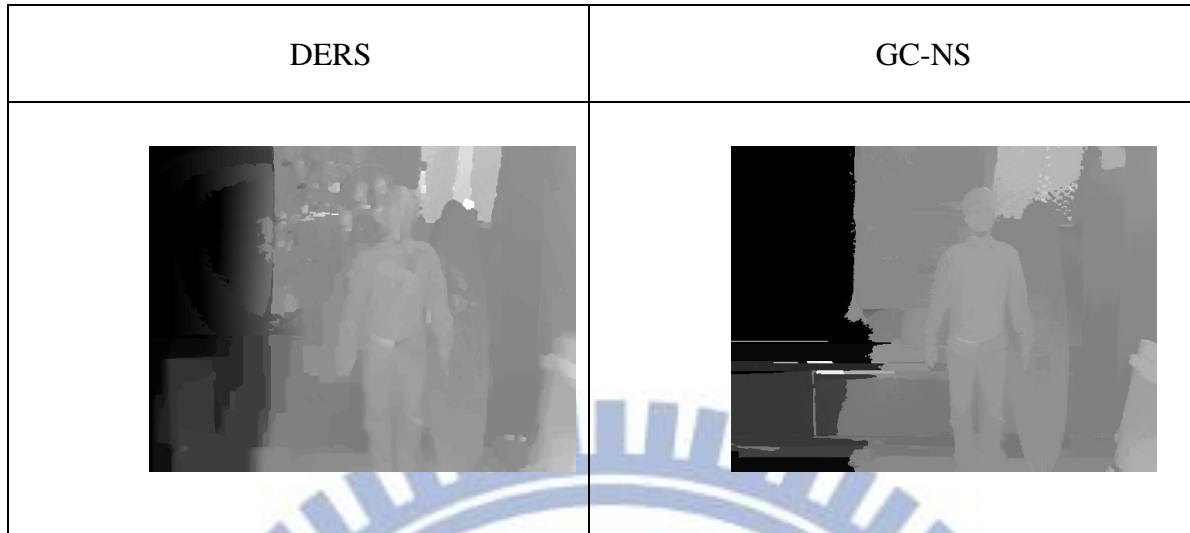


Figure 56: Disparity maps on images with unicolor plaid pattern T-shirt

Table 14: BPR of the estimated disparity maps in Figure 55 and Figure 56

Region	Type	WTA	DERS	GC-NS
Foreground	Letters	0.7808	0.0326	0.1621
	Multicolor	0.7414	0.0132	0.0094
	Unicolor	0.9479	0.0616	0.0122

Table 15: MSE of the estimated disparity maps in Figure 55 and Figure 56

Region	Type	WTA	DERS	GC-NS
Foreground	Letters	196.83	31.43	105.12
	Multicolor	149.38	13.37	9.61
	Unicolor	352.01	44.16	12.88

The results of all three methods become worse when changing cloth from multicolor plaid pattern to unicolor plaid pattern. The results of T-shirt with letters using WTA and DERS are better than the results of T-shirt with unicolor plaid pattern, but worse than that of T-shirt with multicolor plaid pattern. However, the results of T-shirt with letters using GC-NS are the worst in the three types of patterns. Among the three methods, GC-NS has the best performance, and WTA has the worst performance.

## 5.4 Experimental Results on Image Quality Factors

In this section, we show the experimental results of computing disparity maps by using images of different PSNRs and different rectification errors.

### 5.4.1 PSNR

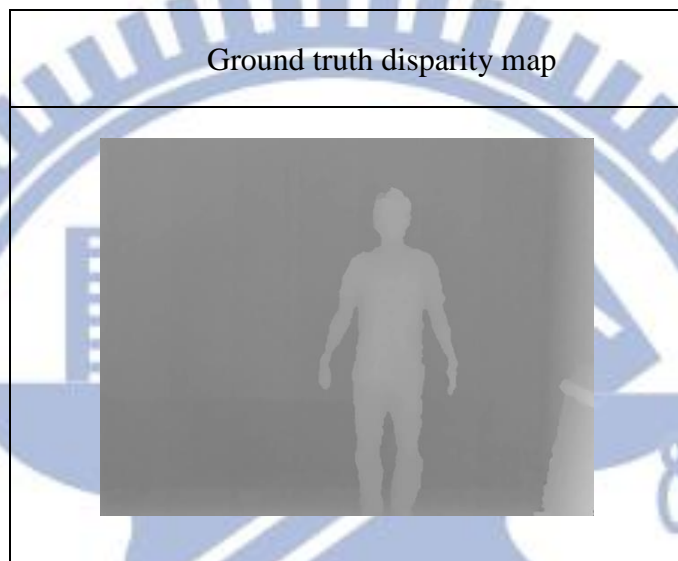
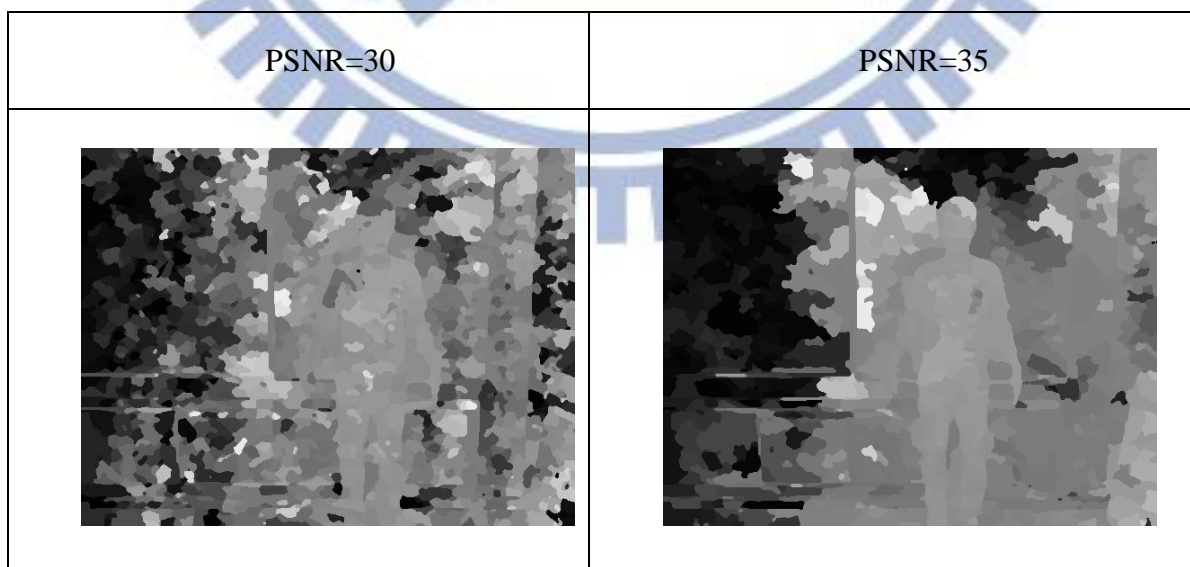


Figure 57: The ground truth disparity map



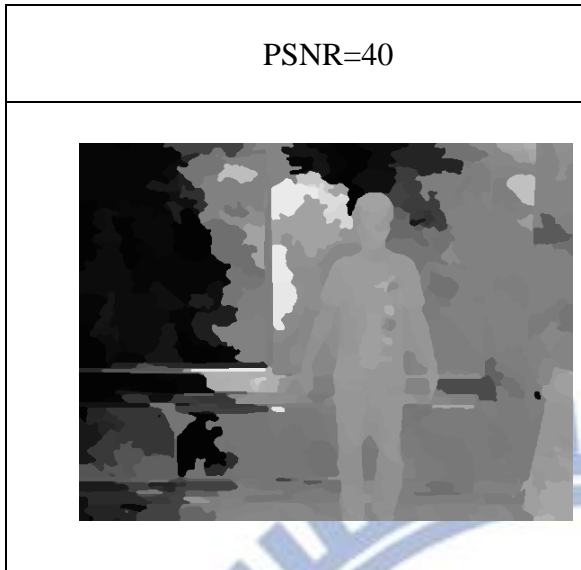


Figure 58: Computed disparity maps using WTA on gaussian noise images of different PSNR

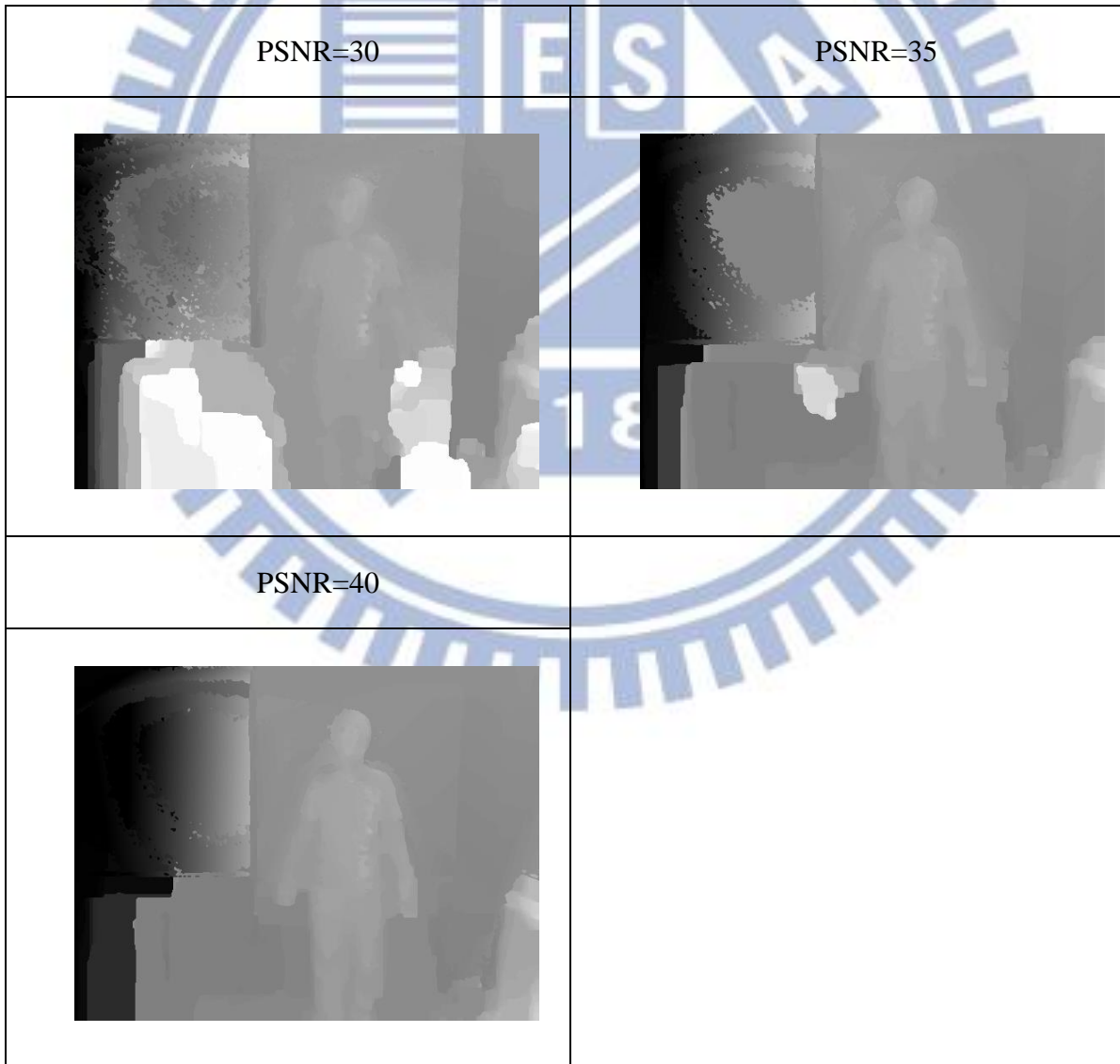


Figure 59: Computed disparity maps using DERS on Gaussian noise images with different

PSNR

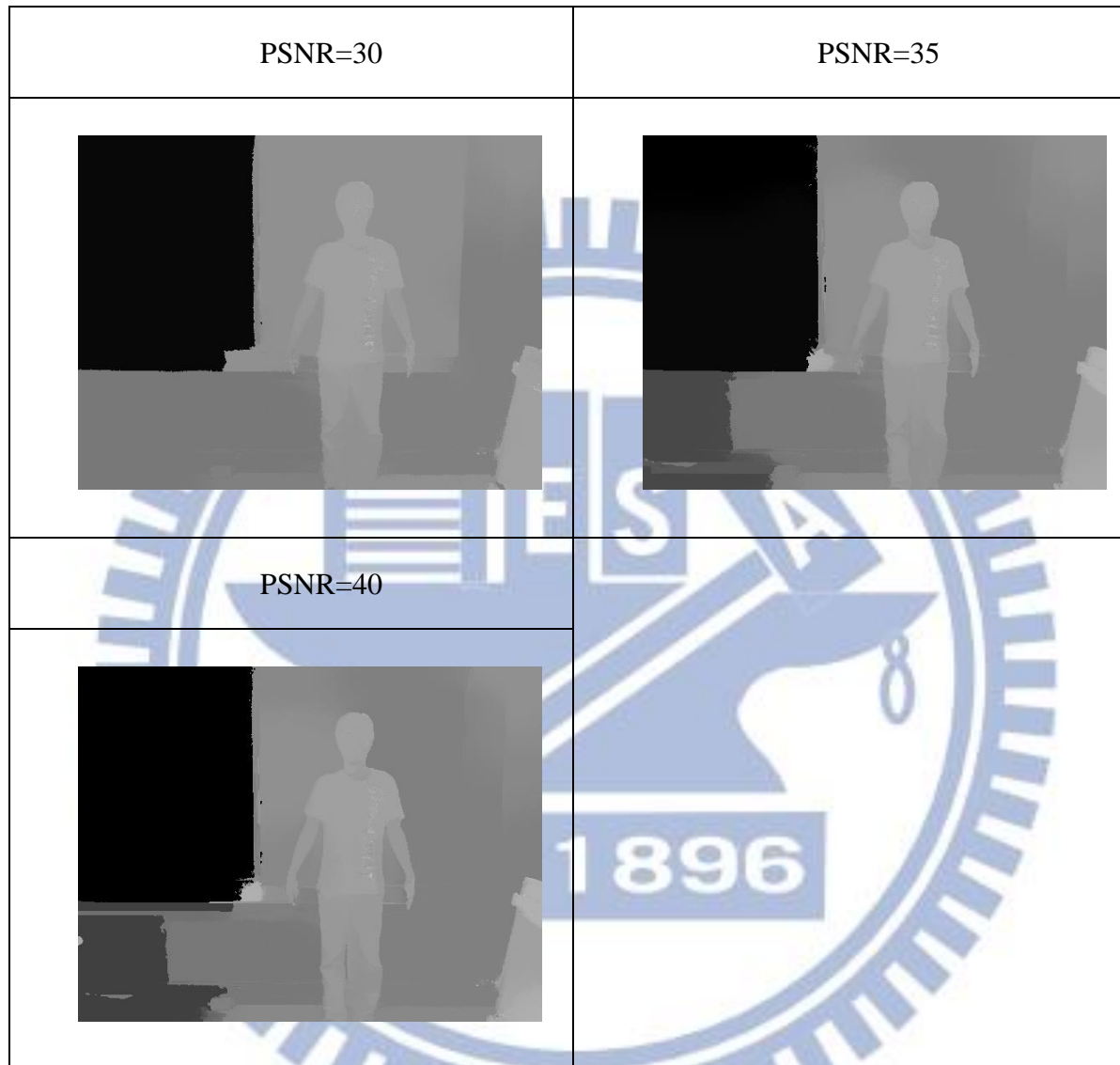


Figure 60: Computed disparity maps using GC-NS on Gaussian noise images with different

PSNR

Table 16: BPR of computed disparity map from Figure 58, Figure 59, and Figure 60

Type	Region	WTA	DERS	GC-NS
PSNR=30	foreground	0.4172	0.0132	0.0203
PSNR=35	foreground	0.2067	0.0022	0.0189
PSNR=40	foreground	0.1541	0.0025	0.0169
Original	foreground	0.1065	0.0180	0.0162

Table 17: MSE of computed disparity map from Figure 58, Figure 59, and Figure 60

Type	Region	WTA	DERS	GC-NS
PSNR=30	foreground	315.46	24.26	25.33
PSNR=35	foreground	91.81	8.45	23.11
PSNR=40	foreground	77.93	9.11	25.32
Original	foreground	100.99	16.39	24.88

The BPR and MSE of WTA get worse when PSNR drops. The results of DERS have the same tendency as WTA. But an interesting phenomenon appears, when we compare the disparity map using the original images with the disparity map using images of PSNR = 40. The one with noise has better result. To check why this phenomenon happens, we use other two sets of images to repeat experiment. They are the images of 0 object in Figure 34, and the images with hand down in Figure 35. Table 18 shows the results:

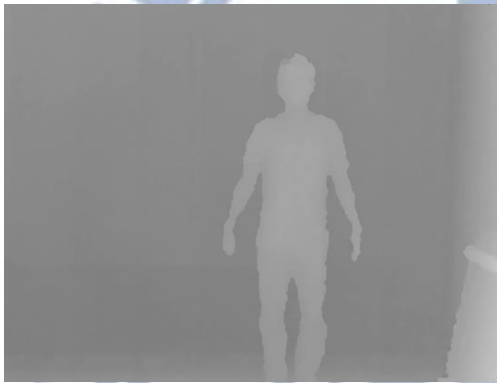


Table 18: Comparison of estimated disparity for images of no noise and images of PSNR=40

Set	Type	Region	DERS
First set	PSNR=40	foreground	0.0254
	Original	foreground	0.0185
Second set	PSNR=40	foreground	0.0252
	Original	foreground	0.0210

The table shows that using the original images has better performance. A possible explanation is that the depth estimation methods are fairly complicated. The high PSNR pictures may occasionally produce better depth maps. But, in general, the original is better.

The BPR and MSE of GC-NS become worse when PSNR drops. Comparing the disparity map using the original images with the disparity map using images of PSNR = 40, the result does not change much.

### 5.4.2 Rectification Errors

Ground truth disparity map	
	
WAT on unrectifid images	WAT on rectifid images
	

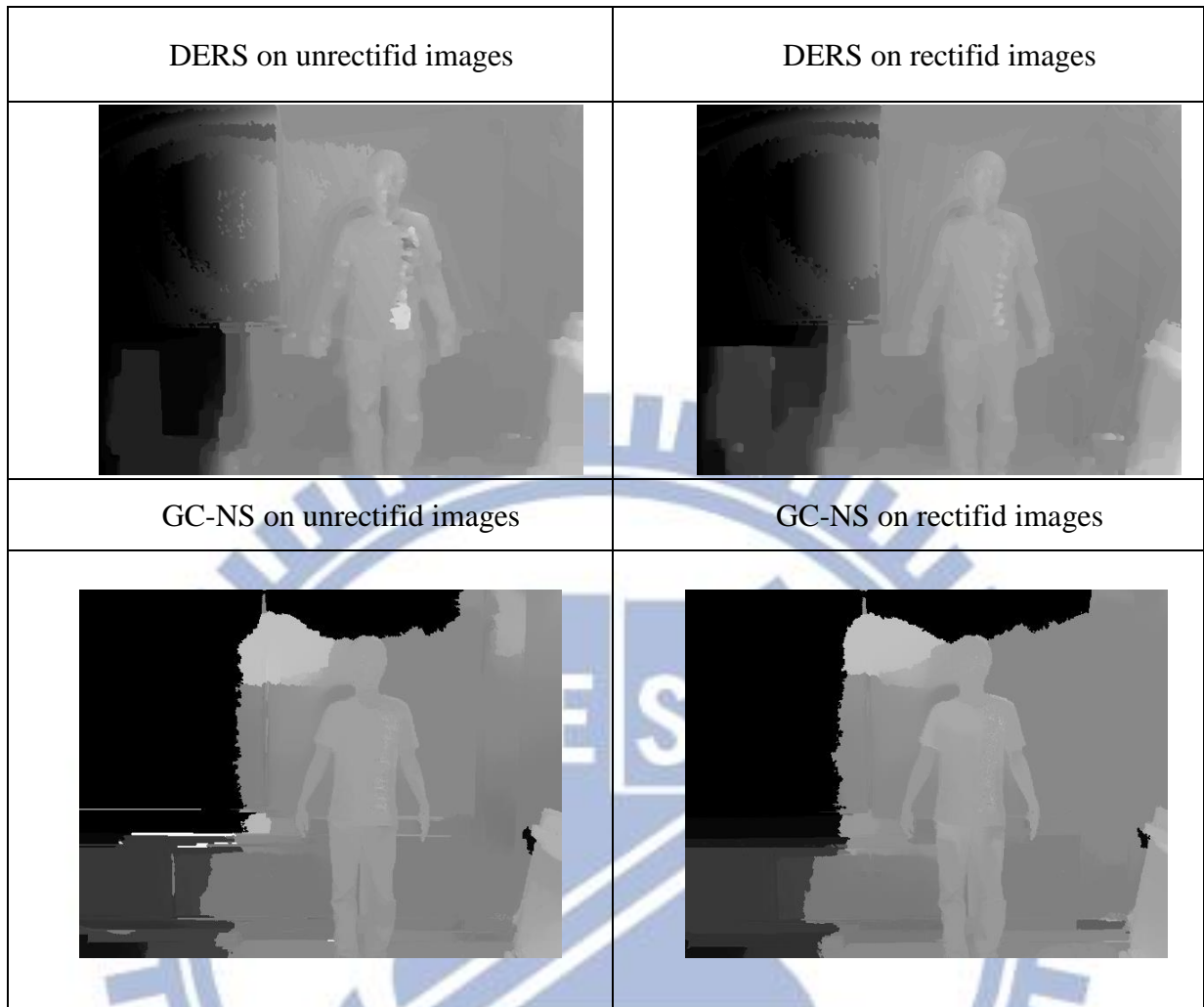


Figure 61: Computed disparity maps on the rectified and unrectified images

Table 19: BPR of computed disparity map on the rectified and unrectified images

Type	Region	WTA	DERS	GC-NS
rectified	foreground	0.1065	0.0180	0.0162
	background	0.6611	0.4627	0.5693
unrectified	foreground	0.1272	0.1039	0.0326
	background	0.5528	0.5083	0.5613

Table 20: MSE of computed disparity map using rectified and unrectified images

Type	Region	WTA	DERS	GC-NS
rectified	foreground	101.25	110.87	27.78
	background	5430.7	4423.6	6769.1
unrectified	foreground	100.99	16.39	24.88
	background	5234.6	4258.8	6839.5

From Table 19 and Table 20, the BPR of three methods all increase in the foreground when the two images are not rectified, and this is what we expected.

Without rectification, the performance of DERS decreases the most in BPR. This is because DERS uses camera parameters to help finding the corresponding pixel when it is not 1D parallel, and the other two methods take every pixel into consideration.

Table 21 and Table 22 show the results of the experiments, where a 1-line to 5-line rectification errors are inserted into test images. The performances of three methods all become worse when the number of error line increases. Focusing on the foreground, GC-NS has the best performance. Although WTA has the worst results, DERS is the one whose performance drops the most when the error line changes from 1 to 5. We believe the reason is that DERS addresses the non-1D parallel situation by using camera parameters, which is not effective as taking every pixel into consideration.



Table 21: BPR of computed disparity map on the iamges of different rectification errors

Type	Region	WTA	DERS	GC-NS
rectified	foreground	0.1065	0.0180	0.0162
	background	0.6611	0.4627	0.5693
1-line error	foreground	0.1971	0.0797	0.0216
	background	0.6625	0.4661	0.5709
2-line error	foreground	0.2701	0.1448	0.0267
	background	0.6272	0.4897	0.5870
3-line error	foreground	0.2865	0.1756	0.0261
	background	0.6203	0.5012	0.5474
4-line error	foreground	0.3364	0.2159	0.1182
	background	0.6715	0.4727	0.5598
5-line error	foreground	0.3301	0.2743	0.1399
	background	0.6492	0.4869	0.5630

Table 22: MSE of computed disparity map on the iamges of different rectification errors

Type	Region	WTA	DERS	GC-NS
rectified	foreground	100.99	16.39	24.88
	background	5234.6	4258.8	6839.5
1-line error	foreground	102.49	60.63	25.70
	background	5512.4	4303.6	6946.0
2-line error	foreground	236.86	100.17	28.15
	background	5185.5	4329.8	6894.1
3-line error	foreground	250.06	112.38	28.52
	background	5661	4088.4	6803.3
4-line error	foreground	198.72	198.72	93.32
	background	59121.1	59121.1	6481.1
5-line error	foreground	414.69	414.69	65.77
	background	5579.6	5579.6	5967.1

## chapter 6 Conclusions and Future Work

### 6.1 Conclusions

In this thesis, we propose a method to evaluate the stereo matching algorithms by using our dataset consisting of stereo pair images. These images designed to include many factors that may affect the performance of stereo matching algorithms. Our evaluation focuses on the foreground, because we assume that the depth map is used for human-computer interaction applications. With this set of evaluation dataset and procedure, we like to know the behavior of a specific stereo matching algorithm. Is it robust to certain disturbance factors?

We summarize the characteristics of the three disparity estimation algorithms test in this thesis.

WTA (stereo matching using non-local aggregation method): When the background is complex, the accuracy of WTA increases. No matter the background has repeated patterns or irregular complex patterns, WTA has better results than the simple background. Reducing the textureless background region can improve its performance. When there are several objects in the scene, WTA has very bad estimation results. A person with arms up horizontally or a person in the T-shirt with unicolor plaid pattern makes the performance worse. When the PSNR=40, WTA produces similar results as the cases without noises in the images. Rectification error has little impact on WTA.

DERS (Depth Estimation Reference Software): When the background is complex, the estimated disparity in the foreground has more errors. Cutting off textureless region helps the estimation accuracy. We found that without left-right cross check, DERS does not handle the occlusion region well. DERS has very poor results in the repeated-pattern regions, but it works well in the complex background. The increase of object number in a scene decreases

the performance of DERS. A person with arms up horizontally or a person in the T-shirt with unicolor plaid pattern increases the errors. PSNR=40 is good enough for DERS to do the depth estimation. Rectification error has huge impact on DERS that the errors increase a lot. GC-NS (stereo matching with nonparametric smoothness priors in feature space): The estimated disparities in the foreground has little change when the background becomes complex. Cutting off the textureless region can be useful to improve the performance. The performance increases a lot in the repeated-pattern region and irregular complex region. When the number of objects increase, its performance gets worse. GC-NS cannot do well using the images where a person with arms up horizontally or a person in the T-shirt with unicolor plaid pattern. The Gaussian noise has little impact on GC-NS when the PSNR = 40. Rectification error has some influences on the performance but not much.

## **6.2 Future Work**

For the proposed dataset, it takes time to generate a dataset consisting of stereo pair images and its ground truth, and we have tried our best to cover all of the factors in the dataset. However, there's still some factor can be added into the dataset, such as illumination, motion blur and shape complexity. Moreover, we can do better to quantize the factors like background complexity. It will be great if we can use sequence instead of single image. For ground truth disparity map, we can find a better active sensor that the black holes can reduce to make the ground truth more reliable.

For the evaluation part, we use BPR and MSE to see the performance. Since our purpose is to evaluate the algorithm for novel applications, we should choose one of the applications to help us complete the evaluation.

## Bibliography

- [1] D. Scharstein, and R. Szeliski, <http://vision.middlebury.edu/stereo/eval/>
- [2] S. Morales, T. Vaudrey, and R. Klette, "Robustness evaluation of stereo algorithms on long stereo sequences," *Intelligent Vehicles Symposium*, IEEE, pp.347-352, 3-5 June 2009.
- [3] J. Chai, X. Tong, S. Cha, and H. Shum : Plenoptic Sampling. In: *Proceeding of ACM SIGGRAPH*, pp. 307–318, 2000.
- [4] H. Shim, and S. Lee, "Performance evaluation of time-of-flight and structured light depth sensors in radiometric/geometric variations," *SPIE Optical Engineering*, vol. 51, 2012.
- [5] LaserFocusWorld: GETURE RECOGNITION: Lasers bring gesture recognition to the home  
[Online] Available:  
<http://www.laserfocusworld.com/articles/2011/01/lasers-bring-gesture-recognition-to-the-home.html>
- [6] Depth Biomechanics: Background  
[Online] Available: <http://www.depthbiomechanics.co.uk/?cat=18>
- [7] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *IEEE Workshop on Stereo and Multi-Baseline Vision*, pp.131-140, 2001.
- [8] M. Humenberger, and C. Zinner "A fast stereo matching algorithm suitable for embedded real-time systems," *Journal on Computer Vision and Image Understanding*, pp.1180-1202, 2010.
- [9] A. Fusiello, V. Roberto, and E. Trucco, "Efficient stereo with multiple windowing," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 858–863, June 1997.
- [10] M. F. Tappen, and W. T. Freeman, "Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters," *Ninth IEEE International Conference on Computer Vision. Proceedings*, vol.2, pp.900-906, Oct. 2003.
- [11] C. Kim, K. M. Lee, B. T. Choi, and S. U. Lee, "A dense stereo matching using two-pass dynamic programming with generalized ground control points," in *Proc. IEEE Conference on Computer Vision and*

*Pattern Recognition*, vol.2, pp.1075-1082, June 2005.

- [12] A. Rehman, and Zhou Wang, "Reduced-reference SSIM estimation," *17th IEEE International Conference on Image Processing*, pp.289-292, 26-29 Sept. 2010.
- [13] H. Hirschmuller, and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1,8, 17-22 June 2007.
- [14] R. Hartley, and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2th edition, 2003.
- [15] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Machine Vision and Applications*, vol. 12, pp. 16–22, 2000.
- [16] Na-Eun Yang, Yong-Gon Kim, and Rae-Hong Park, "Depth hole filling using the depth distribution of neighboring regions of depth holes in the Kinect sensor," *IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC)*, pp.658-661, 12-15 Aug. 2012.
- [17] Zhengyou Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol.1, pp.666,673, 1999.
- [18] Camera Calibration Toolbox for Matlab. Available together with the software at [http://www.vision.caltech.edu/bouguetj/calib\\_doc/](http://www.vision.caltech.edu/bouguetj/calib_doc/)
- [19] J. Smisek, M. Jancosek, and T. Pajdla, "3D with Kinect," *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp.1154-1160, 6-13 Nov. 2011.
- [20] GIZMOWATCH: What you need to know about the Kinect for XBOX 360  
[Online] Available:  
<http://www.gizmowatch.com/entry/what-you-need-to-know-about-the-kinect-for-xbox-360/>
- [21] Kinect for Windows: DOWNLOAD KINECT FOR WINDOWS SDK  
[Online] Available: <http://www.microsoft.com/en-us/kinectforwindows/develop/overview.aspx>
- [22] G. Danciu, S. M. Banu, and A. Caliman, "Shadow removal in depth images morphology-based for Kinect cameras," *International Conference on System Theory, Control and Computing (ICSTCC)*, pp.1-6, 12-14

Oct. 2012.

[23] Contour Tracing: Defining Connectivity

[Online] Available:

[http://www.imageprocessingplace.com/downloads\\_V3/root\\_downloads/tutorials/contour\\_tracing\\_Abeer\\_George\\_Ghuneim/connectivity.html](http://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/connectivity.html)

[24] Qingxiong Yang, "A non-local cost aggregation method for stereo matching," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1402-1409, 16-21 June 2012.

[25] B. M. Smith, Li Zhang, and Hailin Jin, "Stereo matching with nonparametric smoothness priors in feature space," *IEEE Conference on Computer Vision and Pattern Recognition*, pp.485-492, 20-25 June 2009.

[26] ISO/IEC JTC1/SC29/WG11, N11631, "Report on Experimental Framework for 3D Video Coding," October 2010.

