3D Sensing and Tracking of Human Gait

by

Lin Yang

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc degree in
Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Motion capture technology has been applied in many fields such as animation, medicine, military, etc. since it was first proposed in the 1970s. Based on the principles applied, motion capture technology is generally classified into six categories: 1) Optical; 2) Inertial; 3) Magnetic; 4) Mechanical; 5) Acoustic and 6) Markerless. Different from the other five kinds of motion capture technologies which try to track path of specific points with different equipment, markerless systems recognize human or non-human body's motion with vision-based technology which focuses on analyzing and processing the captured images for motion capture. The user doed not need to wear any equipment and is free to do any action in an extensible measurement area while a markerless motion capture system is working. Though this kind of system is considered as the preferred solution for motion capture, the difficulty for realizing an effective and high accuracy markerless system is much higher than the other technologies mentioned, which makes markerless motion capture development a popular research direction. Microsoft Kinect sensor has attracted lots of attention since the launch of its first version with its depth sensing feature which gives the sensor the ability to do motion capture without any extra devices. Recently, Microsoft released a new version of Kinect sensor with improved hardware and and targeted at the consumer market. However, to the best of our knowlege, the accuracy assessment of the sensor remains to be answered since it was released. In this thesis, we measure the depth accuracy of the newly released Kinect v2 depth sensor from different aspects and propose a trilateration method to improve the depth accuracy with multiple Kinects simultaneously. Based on the trilateration method, a low-cost, no wearable equipment requirement and easy setup human gait tracking system is realized.

Keywords. Motion capture, multi-Kinect trilateration, depth, accuracy, Kinect v2, gait tracking.

Acknowledgements

My deepest gratitude goes first and foremost to Professor Abdulmotaleb El Saddik, my supervisor, for his constant encouragement and guidance. Without his consistent and illuminating instruction, this thesis could not have reached its present form.

Second, I would like to express my special heartfelt gratitude to Dr. Haiwei, who introduces me the topic and provides me many insightful suggestions. Through the entire research work, his erudition and preciseness guide me not only in academic level but also in my personal life.

I would also like to thank all colleagues in MCRLab and all my friends for giving their time to help work out the problems existing in my project. Last thanks would go to my family for their loving considerations and great confidence in me all through these years.

Table of Contents

List of Tables

List of Figures

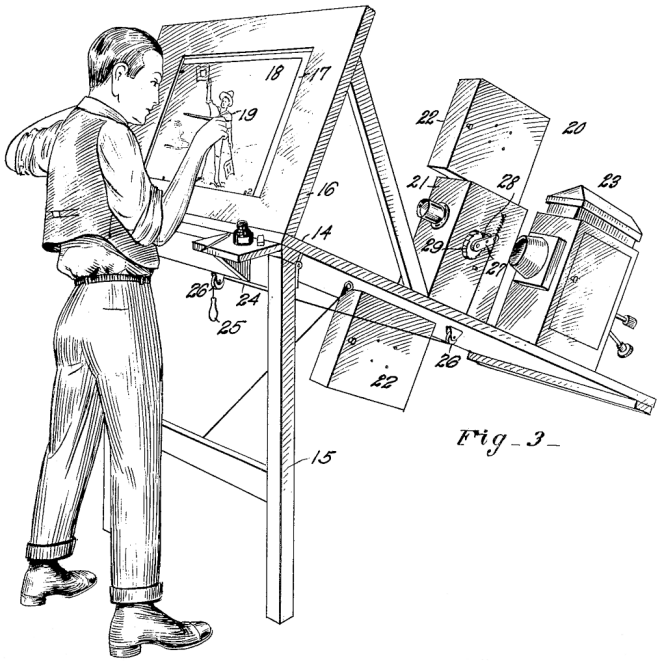
Chapter 1

Introduction

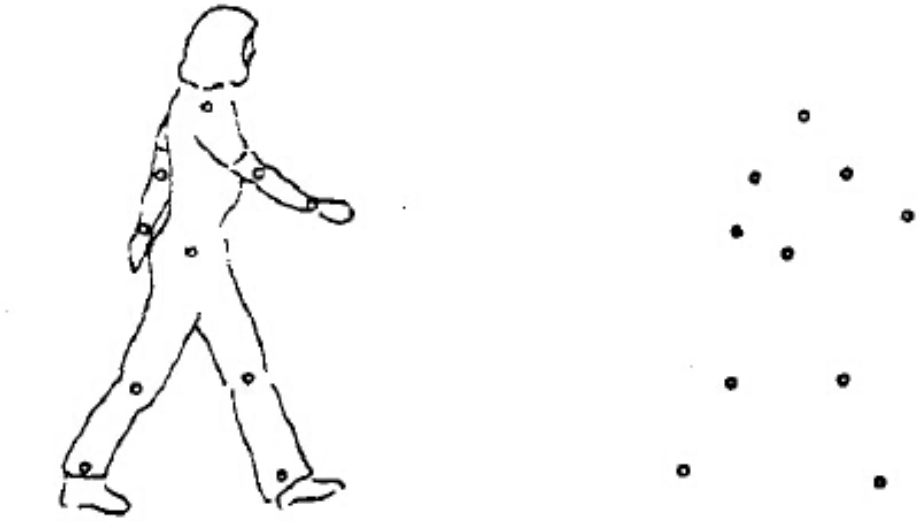
1.1 Background and Motivation

1.1.1 Motion Capture

In the late 1970s, the motion capture technique was first proposed by Johansson [2] in a moving light display (MLD) experiment. Different from the older techniques applied in animation, such as rotoscoping or cel animation, the movements of a human or nonhuman body part is recorded for the first time while ignoring the human or nonhuman body's visual appearance. In rotoscoping, the animator is required to trace the film's specific frames one by one and redraw the specific human body or nonhuman body with a piece of equipment called a rotoscope [1] (Fig. 1.1a). Finally, the film is generated by compositing the original frames and the redrawn frames. In animation with motion capture techonogies, the recorded movements (usually movements at specific points visualized from the human body or nonhuman body's motions) can be mapped to different 3D models such that these 3D models simulate the same actions as the original human or nonhuman body performs (Fig. 1.1b). The development of the motion capture technique was promoted in the 1980s by researchers such as Carol from MIT [4], who addressed the challenges in person-modeling 3D animation systems, and Robertson [5], who developed an application able to visualize facial expressions by capturing human facial expressions in reality. In the 1990s, motion capture technologies continued to mature and develop new processes, such as a CCD-camera based system capable of biomechanical motion analysis [6], a knowledge-based framework for capturing a human walker's movements and reconstructing the corresponding models in real-life video [7], and a visual motion estimation technology for recovering complex motions [8]. In recent years, motion capture technologies have developed even more rapidly, they can generally be divided into several categories: 1) optical systems that apply several



(a)



(b)

Fig. 1.1: Traditional rotoscoping and the virtualized human body with motion capture technology. (a) In rotoscoping procedure, the animator is required to stand infront of a projector, re-drawing the live actions in a film [1]. (b) In motion capture, the movement of human body is recorded while the visual appearance is ignored [2].

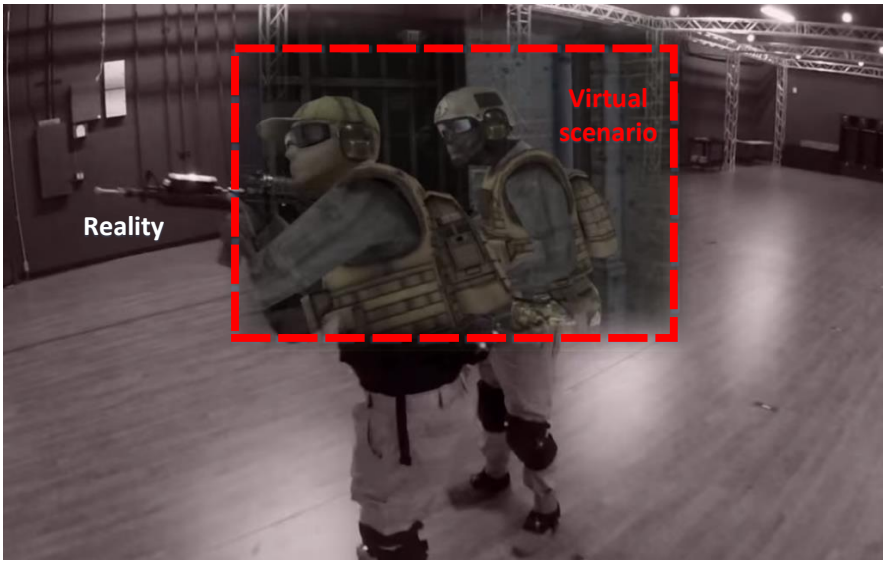


Fig. 1.2: Visualized scenario by Motion Reality, Inc. (MRI)[1]. For military purpose, the gears (such as guns) are closely simulated in both reality and virtual scenarios.

cameras positioned accordingly for motion capture; 2) inertial systems based on acceleration and rotational measurements from MEMS (micro-electro-mechanical system) chips with specific sensors attached on a body's strategic points; 3) magnetic systems that require the user to perform actions in a specific magnetic field while wearing specific sensors able to detect the magnetic field; 4) mechanical systems that require an exoskeleton to be equipped by users for motion capture; 5) an acoustic system that applies ultrasonic transmitters and microphones to determine the location of human joints and relative positions between each pair of microphones; and 6) markerless systems that track a human or nonhuman body's motion based on computer vision technologies without requiring specific markers to be attached on the human or nonhuman body [9, 10, 11]. Nowadays, many motion capture systems have been promoted in the marketplace, such as Vicon motion capture system and Xsens Moven, and motion capture technologies related to CG (Computer Graphics) and VR (Virtual Reality).

In the 1970s, the first motion capture technology' application was developed for military purposes. This application actually tracks the pilot's head movement through a process based on magnetism. As motion capture technology has advanced, more military applications have been developed. Nowadays, motion capture systems research for military purposes mainly focus on training for soldiers or simulating specific battle scenarios. As a small mistake might lead to a serious consequence—and as live drills in the field are costly

[1] http://www.motionreality.com/index

Fig. 1.3: Na'vi's face expression is created by visualizing the actor's face expression in Avatar[2].

in terms of time and money— motion capture technology has potential value if relative real scenarios or soldiers' motions can be captured and visualized in real time. Usually, the applications visualize real battlefield elements (buildings, villages, etc.), human body movement, and the equipment involved (Fig. 1.2). According to how the soldiers perform in these scenarios, the examiners could design further training plans.

Motion capture technology has been applied for entertainment purposes since the 1980s. There are three main fields that apply motion capture technologies: animation (or film making), VR (virtual reality), and gaming. Around the year 2000, motion capture technology started to be applied in animation with better performance and lower cost compared to traditional animation methods, such as Cel animation [12]. Usually, in animation or film-making, the actor's body movement and facial expression are recorded by visualizing the human body as joint combinations for body motion and visualizing facial expression, as a mesh with specific points representing eyes, nose, mouth, and so on (Fig. 1.3). Thus, in CG animation, visualized creatures (e.g., the orcs in The Lord of the Rings and the Na'vi in Avatar) move like humans with human expressions. The first well-known movie made primarily by motion capture technology was Final Fantasy: The Spirits Within, and the most successful one was Avatar by James Cameron. Nowadays, in almost every animation or fiction movie, motion capture technology is applied.

In game development, motion capture technology is mainly applied in action-adventure games (A-AVG) because of their requirement for a large quantity of human motion. Other than CG video creation in games, there are typically two methods to apply motion capture

[2]http://www.develop-online.net/tools-and-tech/motion-capture-moving-with-the-times/0117523

Fig. 1.4: Capture actor's motion in reality and map it to character in game by XSENS[3].

technology in game development. The first involves designing and recording specific movements or actions in reality with motion capture technology and mapping the movements or actions to the 3D models of the characters (controlled by players) in the games (Fig. 1.4), which provides a close simulation of human motion when the player is controlling the character to do specific actions. The second involves capturing the player's motions, mapping the motions to the characters in the game, and showing how the players interact with the video games accordingly in real time (e.g., Kinect Sports) (Fig. 1.5). In the second method for applying motion capture technology in game development, specific hardware is required to help record movements, such as walking for a long distance, and VR technology is usually applied to help visualize game scenarios (e.g., a city or a jungle). As Fig. 1.6 shows, for achieving a better sense of immersion or more of a purpose for exercising, the player needs to perform real actions in real time. For the motion capture and VR technology to provide perfect technical support, the player needs to wear a special belt for calculating the waist's rotation angle, stand on a specific reel for calculating the moving speed, and be attached with some specific markers for tracking hand movements. The player is able to be visualized in real size and free to do any actions in reality while playing the game.

In biomechanics, motion capture technology is mainly applied for observing and recording the human body's biomechanical data. With these data, many applications for understanding human motion, improving the physical quality of the human body, and detecting disease have been developed. One of the most popular applications with motion capture technology is sports analysis. Almost every kind of sport analyzes skill levels to determine

[3]http://www.develop-online.net/tools-and-tech/motion-capture-moving-with-the-times/0117523

Fig. 1.5: Sports game with motion capture technology by Kinect Sports[4]. A Kinect sensor is positioned accordingly for capturing the users' motion and map the motions to the 3D character in game, which realize real-time interactive gaming.

Fig. 1.6: Shooting game with motion capture and virtual reality technology by Virtuix Omni[5] gives a solution for free movement entertainment environment.

how players can increase performance while decreasing the potential for injuries. Sports analysis applications apply motion capture technology to collect biomechanical data, analyze the data from a clinical perspective, define the user's skill level, observe potential injury causes, and report about how to exercise correctly or how to directly operate sports equipment (e.g., adjusting the speed of a treadmill, controlling the duration of running, and giving running posture advises), as shown in Fig. 1.7. Rehabilitation analysis is also

[4]http://www.xbox.com/en-CA/xbox-one/games/kinect-sports-rivals
[5]http://www.virtuix.com/

Fig. 1.7: Running analysis solution by Qualisys[6] help improve sports skills and reduce potential injuries with biomechnical data captured by motion technology while a person runs.

one of the most popular application categories with motion capture technology. By accurately assessing gait movement and electromyography (EMG) data, the current condition of a patients' gait can be analyzed based on clinical knowledge, and more precise rehabilitation plans can be made for specific patients (Fig. 1.8). One interesting application that has been developed with motion capture technology is music and movement analysis. By capturing the motions of musicians and dancers while they are performing with different kinds of music, these kind of application analyze how people perceive music and generate sound (the specified pitch and instrument) according to the movement created by the user. With these kinds of applications, people can interact with music and create music in a creative way. One of the most popular systems in music analysis is the Qualisys system [13].

Motion capture technology is gaining in popularity due to its high performance in entertainment, the military, biomechanics, and other areas. By assessing human motion, motion capture technology helps people understand how human beings perceive the world and benefits people by providing a more creative and healthier environment. However, there are several general problems with the current motion capture systems.

- ***They are expensive.*** Usually, a motion capture system requires professional hardware devices for reaching high accuracy. For example, for gait analysis, a Vicon system [14] has a starting price of $12,000, excluding training fees, calibration fees,

[6]http://www.qualisys.com/applications/biomechanics/running/
[7]http://www.qualisys.com/applications/customer-cases/sound-and-motion-at-fourms-lab/

Fig. 1.8: Music and motion analysis by Qualisys[7] access how people perceive music while musician is performing and try finding a new way for music creation with human motion.

installation fees, and so on[8].

- ***They require wearable gear.*** Most motion capture technologies (except vision-based motion capture technologies) require specific equipment to be worn on the human body, including 1) optical systems that require markers to be attached to the body such that the specific cameras can detect the joints from a background, 2) inertial systems that require an accelerometer and gyroscope to be attached to the body while tracking movements, 3) mechanical systems that require users to wear an exoskeleton for tracking motion, 4) magnetic systems that require a magnetic field to be generated (it also requires users to act in the magnetic field such that the motion can be tracked by the magnetic sensors attached to the body), and 5) ultrasonic systems that also require users to wear specific receptors such that the motion can be tracked by the ultrasonic generator.

- ***A complex setup is required.*** Different motion capture technologies are based on different locating theories, which means that the requirements of the system environment are different. Usually, for motion capture, more equipment (more wearable gear, more observing cameras, etc.) indicates better accuracy and results in a more complex setup. For example, in a Vicon system, tens of cameras need to be positioned accordingly, and several calibration procedures need to be repeated for better accuracy (this is why there are several costly courses for Vicon system's installation, debugging, calibration, etc.).

In this thesis, we propose a vision-based technology with the newly released Kinect for Windows v2 sensors. The system has a low cost and is easy to use, and no wearable

[8]http://www.peppm.org/Products/viconindustries/price.pdf

equipment is required for gait tracking.

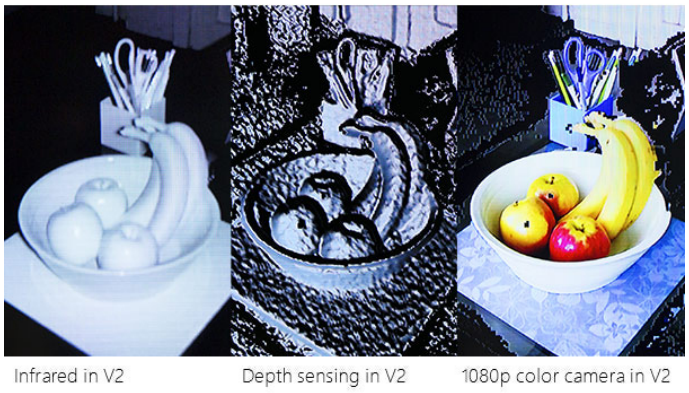
1.1.2 Kinect for Windows sensor

In the past few years, the areas involving computer vision have been developing quickly due to the improvements in computer and camera capabilities. While RGB cameras capture the color information, depth cameras measure the range information between the camera and the object, which offers more convenience for three-dimensional (3D) model construction, object tracking, movement detection, and so on, as shown in Fig. 1.9. Both industrial and consumer-grade depth sensors have been developed to satisfy the requirements demanded by researchers and common users. Typical devices include the Cyberware laser head scanner [15], MESA Imaging SRFamily Time-of-Flight (ToF) camera [16], and PrimeSense Carmine structured-light cameras [17].

The launch of Microsoft Kinect for Windows sensor v1 [18] in November 2010 further enriched the depth-camera device options with its low consumer price, compact size, and the capability to capture depth and image data at video rate. Kinect v1 projects patterns consisting of many stripes at once and allows the acquisition of a multitude of samples simultaneously. It has several advantages: it can capture color and depth images at a video rate well in low light levels and resolve silhouette ambiguities in pose. It is also color and texture invariant. Additionally, its operation has no difference compared with video cameras, so it can be easily operated by non-expert users [19].

On July 15th, 2014, Microsoft has released Kinect v2, which has significantly improved the depth measurement accuracy. Regarding the depth-sensing principle, Kinect v1 adopts a structured-light method, which projects patterns consisting of many stripes (or arbitrary fringes) at once, and allows the acquisition of a multitude of samples simultaneously [20]. Compared with v1, Kinect v2 utilizes a time-of-flight method, which uses active sensors to measure the distance of a surface by calculating the round-trip time of a pulse of light [21]. The captured depth images from Kinect v2 have better quality, and in our work, we mainly focus on evaluating the depth-sensing capability of Kinect v2.

Besides evaluating Kinect v2's depth accuracy, we also propose a method to improve its accuracy by applying the trilateration principle with multiple Kinect v2 devices. The trilateration principle has been widely used in Global Positioning Systems (GPS), Wireless Sensor Networks (WSN), and other localization research areas [22]. In our experiment, we put three Kinect devices at different locations while measuring the same target and then apply the proposed trilateration method to achieve the optimal position of the target. Compelling results can be ascertained with our method. To the best of our knowledge,



(a)

(b)

Fig. 1.9: Images from Kinect for Windows v2 sensor. (a) Besides the color vision at right, depth sensor gives depth frames. Based on infrared vision at left, depth sensing can be realized with TOF technology[9]. (b) Based on depth sensing, human motion capture is realized in 3D space[10].

there have been no existing studies evaluating the accuracy of Kinect v2 until now.

1.2 Objective and Contribution

Though the accuracy of Kinect v2s might not be as good as the professional systems developed by Vicon [23], Qualisys [14] or Northern Digital Inc. [24], its current price is much lower than commercial systems. By applying the multi-Kinect trilateration method, the system proposed in this thesis improves the overall accuracy of the Kinect sensors applied in our system, which is low cost, requires no wearable equipment, and is an easy-

[9]https://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx
[10]http://rfilkov.com/2014/05/13/kinect-v2-whats-new/

to-set-up human gait-tracking solution.

The main contributions can be summarized in th following points:

- ***Development of a method to access the accuracy of Kinect for Windows v2 sensor.*** Before proposing the gait-tracking system, we evaluated the accuracy of the Kinect v2 for Windows sensor. As the key features, such as the joint tracking and 3D data fusion, are based on depth images, the accuracy evaluation in this thesis mainly focuses on the depth camera embedded in Kinect v2. A total of five attributes are evaluated from different aspects: depth accuracy distribution, depth resolution, depth entropy, edge noise, and structural noise.

- ***Design two multi-Kinect trilateration methods to improve Kinect v2's accuracy.*** According to the accuracy evaluation results, the Kinect sensor's depth accuracy varies when the target object is placed at different positions. The gait tracking requires a relative large range of human body movements, so the Kinect sensors might not be able to generate data accurately when the Kinect v2 sensors are positioned relatively far from the target object. We propose two multi-Kinects solutions based on trilateration for improving the overall accuracy when the Kinect sensors are positioned where they are not able to obtain data accurately. The first trilateration method is based on geometry theory and requires three Kinect sensors. The second method is based on the least square theory and minimizes the accuracy error when more than three Kinect sensors are applied.

- ***Developing a cost-effective, highly accurate system for human gait tracking.*** Based on the multi-Kinect trilateration method, a human gait-tracking system with three Kinect sensors is realized. The user is required to walk for several steps while the system is tracking six joints of the user's legs.

Publications:

Lin Yang, Longyu Zhang, Haiwei Dong, Abdulhameed Alelaiwi and Abdulmotaleb El Saddik. Evaluating and improving the depth accuracy of Kinect for Windows v2. *IEEE Sensors Journal*, 8(15):4275-4285, 2015.

1.3 Thesis Organization

This thesis is organized as follows:

- ***Chapter 2:*** In this chapter, we presents a review of the literature in recent years on related topics, including the principles of the popular motion capture technologies, depth-sensing technologies, and calibration methods for improving a Kinect sensor's accuracy.

- ***Chapter 3:*** This chapter assesses the accuracy of Kinect for the Windows v2 sensor and defines five attributes for evaluating the accuracy from different aspects. The methods behind the experiments for evaluating specific attributes are introduced, and the results are discussed in Chapter 5.

- ***Chapter 4:*** The geometry method and least square method, which are based on the trilateration principle, are described in this chapter. To verify that the two methods improve the overall accuracy of the three Kinect sensors, specific experiments with specific configurations are conducted. The results are shown in Chapter 5.

- ***Chapter 5:*** This chapter shows the results of the experiments (conducted in Chapter 3) for accessing the accuracy of the newly released Kinect sensor and the results of experiments based on the trilateration principle (in Chapter 4) for verifying that the locating methods improve the overall accuracy of the three Kinect sensors.

- ***Chapter 6:*** Based on the experimental results, a human gait-tracking system was realized. The details of the implementation are described in this chapter, including system requirements, overall system architecture, an introduction of each component, and the system results.

- ***Chapter 7:*** This chapter concludes the thesis and discusses future work.

Chapter 2

Related Work

2.1 Motion capture technologies

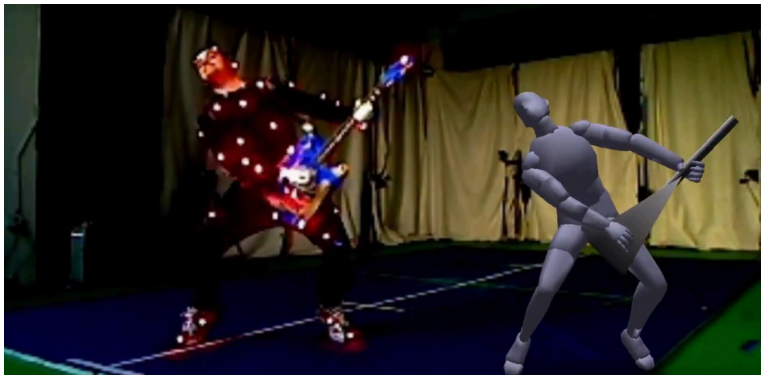
At present, motion capture technologies can be classified into six categories: 1) optical systems, 2) inertial systems, 3) magnetic systems, 4) mechanical systems, 5) acoustic systems, and 6) markerless systems. In this section, the principles of each kind of motion capture technology are described briefly, as well as their advantages and disadvantages.

2.1.1 Optical System

Optical systems can be divided into two categories: passive marker and active marker.

Passive marker: In optical motion capture technology with passive markers, multiple high-speed cameras are positioned at fixed positions around the measurement area. Several markers that are able to reflect the special light beams from the cameras are required to be attached on the joint positions. Thus, if one specific joint can be observed by any two cameras, the joint's position can be located in 3D space based on the triangulation principle [25]. By continuing to capture the frames with high-speed cameras, the joints are tracked. These systems can track the target accurately (usually, accuracy is measured as less than one millimeter with a distance larger than 10 meters). As the markers are attached without any cables or batteries, the user is free to perform any actions. These systems are easily affected by intense light source nearby because the camera can be affected by noise. As the cameras are fixed around the measurement area, the workspace is limited. One other disadvantage is that the reflective markers are not identified for recognizing each marker at a specific position, which leads to occlusion or an incorrect tracking result. To solve this problem, additional post processing is required.

(a)



(b)

Fig. 2.1: (a) Motion capture with passive markers by Vicon[11]. (b) Motion capture with active markers by PhaseSpace[12].

Active marker: In optical motion capture technology with passive markers, the markers consist of infrared emitting diodes (IREDs). By dividing the frequency of the high-speed cameras, the markers can be recognized by a specific frequency band. Though the system is limited to the cameras' frequency and the number of markers, post-processing procedures are not required. As the markers need power, cables and batteries are required to be attached to the human body. Similarly, the measurement area is limited, since the cameras are located at fixed positions.

[11]http://www.vicon.com/
[12]http://www.phasespace.com/

Fig. 2.2: Inertial motion capture developed by XSENS[13].

2.1.2 Inertial System

As Microelectro Mechanical Systems (MEMS) have been developed, advanced, and applied in different areas (e.g., in mechanics and electronics), MEMS sensors, such as accelerometers and gyroscopes, are small enough to be attached to a human or nonhuman body while tracking movements. Inertial systems apply accelerometers and gyroscopes to obtain the speed and the rotation angle of one specific point in a 3D space. With the two parameters, the point displacements can be calculated during a specific period. As the sensors are actually estimating the position by obtaining velocity continuously, errors accumulate. The accuracy of the MEMS sensors are different when the sensors are attached at different positions or when a user is performing different actions. In one experiment, the angle error was more than 10 degrees RMS if the MEMS sensors are attached to an ankle joint when the user is rotating his ankle [26]. One of the most popular systems for applying this technology is Xsens MVN developed by XSENS [27].

[13]https://www.xsens.com/

Fig. 2.3: ULTRATRAK PRO system developed with magnetic motion capture by Polhemus[14].

2.1.3 Magnetic systems

A magnetic system consists of an electromagnetic emission source and electromagnetic receivers. The magnetic source builds a low-frequency magnetic field while the user is moving in the measurement area and wearing several electromagnetic receivers. Based on the Hall effect [28], the movement of the sensor can be estimated. For establishing a stable magnetic field while the user is moving, the electromagnetic emission source is usually mounted at the top of the measurement area. This kind of system is cost-effective and has no occlusion issues. One of the disadvantages is the limitation in the size of the measurement area because the size of the magnetic field is limited. If the data is transmitted wirelessly, transmission interference occurs because multiple sensors are working simultaneously. If the data is transmitted with wires, the wires limit the user's movement. One another disadvantage is that this kind of system requires that there is no high magnetic field nearby (i.e., no metallic object). One of the most popular systems applied in magnetic motion capture technology is Polhemus's electromagnetic system [29].

[14] http://polhemus.com/

Fig. 2.4: Gypsy 6 with mechanical motion capture by Animazoo[15].

2.1.4 Mechanical System

The most direct method of capturing an object's movement is to measure the object's orientation and displacement with electromechanical potentiometers embedded in the mechanical exoskeleton. There are no external limitations to the environment or occlusions, and it is an effective, real-time, accurate, and high-degree simulation method for motion capture. The disadvantage is obvious: the mechanical exoskeleton limits the range of the human body's motion, as human joints are much more flexible than the mechanical links. Also, the absolute positions of the joints need to be calculated from the rotation data obtained by the sensors embedded in the exoskeleton. One of the most popular mechanical motion capture systems is Gypsy 6 [30] developed by Animazoo.

2.1.5 Acoustic System

Acoustic systems consist of ultrasonic transmitters and ultrasonic receivers. By measuring the time cost of ultrasonic pulse sent from the transmitters to the receivers, the distance can be estimated with the ultrasonic pulse's speed. The position then can be calculated by triangulation with multiple receivers' distance measurements from the same transmitter. By constantly calculating the positions of transmitters attached to the human body, the user's motion is captured. This kind of system perfectly solves the issue of occlusions because the ultrasonic pulse steers clear of obstacles. Also, the price of hardware is relative low cost. As the ultrasonic pulse becomes weaker as distance increase, the size of the measurement area is limited. Limits to the features are the speed of the ultrasonic pulse and relative large latency. Acoustic systems can also easily be affected by environmental

[15]http://www.animazoo.fr/motion-capture-systems/

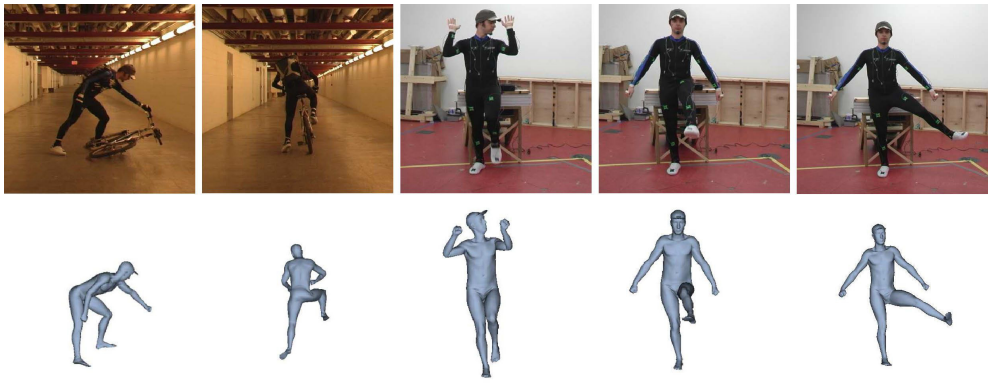


Fig. 2.5: Acoustic motion capture system developed by MIT[16].

Fig. 2.6: Hand's motion capture without markers developed by Leap Motion[17].

factors (e.g., temperature, humidity, and atmospheric pressure). Currently, Logitech is developing this kind of motion capture system [31][32].

2.1.6 Markerless System

Markerless or vision-based systems are realized based on the human vision principle. They use one camera to simulate human vision, which is able to capture body motion without any markers attached to the body. As these kinds of systems are based on the human vision principle, the hardware is low cost, and there are no limitations for the range of human motion and size of measurement area. Though this kind of system is considered a perfect solution for motion capture, the difficulty in realizing an effective and high-accuracy vision-based system is much higher than the technologies mentioned above, which makes markerless motion capture development a popular research topic. Nowadays, one

[16]http://publications.csail.mit.edu/abstracts/abstracts07/drdaniel_2007/drdaniel.html
[17]https://www.leapmotion.com/

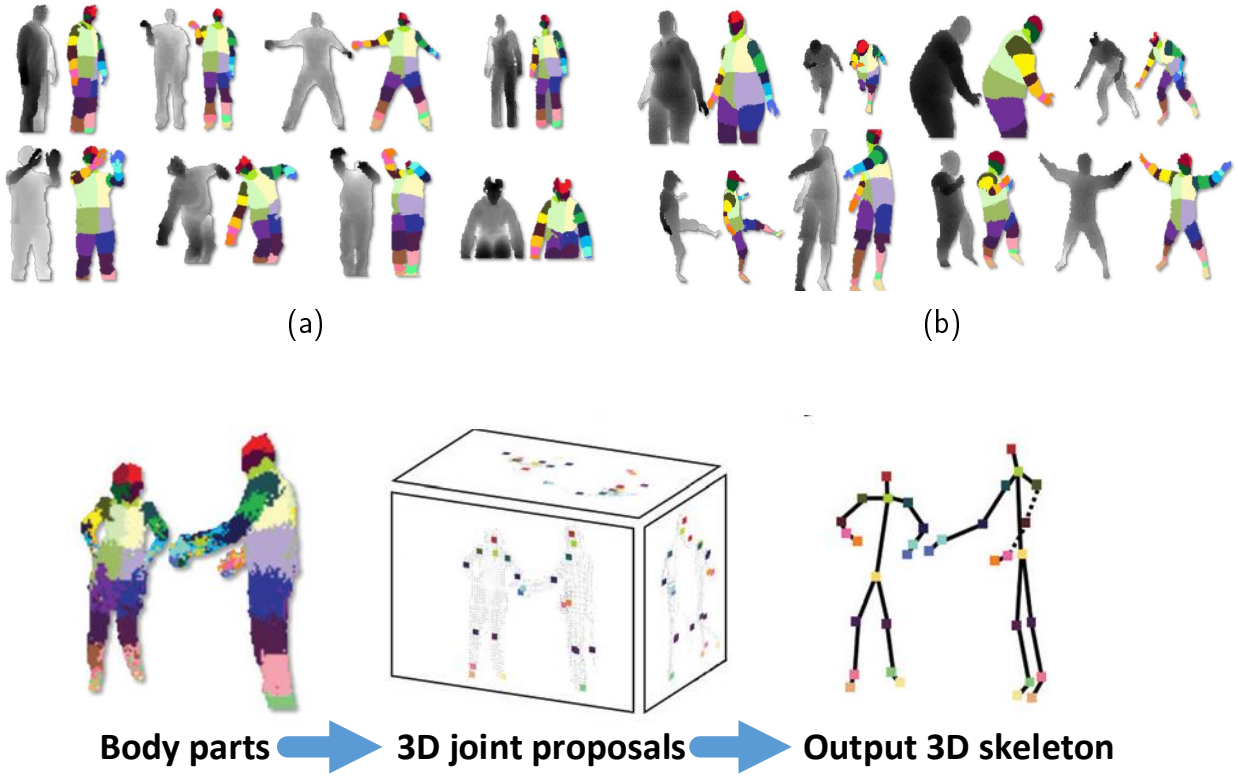


Fig. 2.7: Markerless motion capture technology of Kinect sensor based on depth images. (a) The training data (depth frames) with known body parts. (b) The test result from test data with unknown body parts. (c) Skeleton's generating with mean shift algorithm's result (3D joint proposals) [3].

of the most popular motion capture systems with a vision-based principle is Kinect [18] by Microsoft. Different from other markerless motion capture devices that capture motion with video cameras (generating color frames of scenes in reality), a Kinect sensor captures human motion with depth information and machine learning algorithms. Usually, the markerless systems with video cameras suffer from the quality of the training data: the intensity frames representing human motion that are converted from the color frames are badly affected by the huge color and texture variability of different people's appearance (hair, cloth, skin, etc.). The depth frames give distance information, which is color and texture invariant, between the objects and the camera. There are two steps for capturing human motions: 1) recognizing the body part from a depth frame and 2) finding the joint position within one specific body part. To infer the body part, a database is built consisting of over one million depth images from a known skeleton from another motion

capture system. For each real image, the body parts are rendered using computer graphic techniques (see Fig. 2.7a). A randomized decision forest is built with the training data, and hence, the body part can be recognized after a new depth frame with unknown body parts is obtained (test data) (see Fig. 2.7b). With the recognized body parts, a mean shift algorithm is applied for 3D joint proposals. With a start-depth pixel, mean shift algorithm helps find the modes in this density efficiently and finally estimates the sum of the pixel weights reaching each mode [3]. Another popular motion capture system with a similar principle is the Leap Motion Controller [33] by Leap Motion.

2.2 Depth sensing technology

Though the release of Microsoft's Kinect for Windows v1 sensor attracted lots of attention in the 3D sensing area, there are several kinds of 3D-sensing devices that apply different technologies and that existed before the Kinect sensor. In this section, we introduce the principles of these technologies. Usually, 3D-sensing devices reconstruct a 3D object or 3D environment by generating point clouds or polygon meshes after measuring the physical objects. These devices can be classified into two categories: traditional passive-image-based cameras, which reconstruct 3D objects based on 2D images (without depth information) with multiple cameras at fixed positions, and active 3D-sensing devices, which reconstruct 3D objects based on depth information (usually, a depth map of the object or environment is generated with different principles). In this thesis, we mainly introduce active 3D-sensing technologies using structured-light, time-of-flight (TOF), and X-ray computed technology and a triangulation-based laser.

2.2.1 Structured-light

Structured light has been studied since the 1970s [34]. It is based on stereo vision technology. Instead of using two cameras to observe one object, one camera is replaced with a light source (usually an infrared projector), and the other camera is a specific image sensor that is able to recognize the pattern designed by the light source (a video camera, CMOS sensor, etc.). The light beams for illumination of the scene are designed with a specific spatially varying intensity pattern [35]: the "structured light." Thus, if a planar surface is illuminated by the light source, the image sensor generating the frames and representing a scene captures the pattern (such as stripes with the same intervals) similar to the pattern projected. If an irregular 3D surface, in reality, is illuminated, the non-planar surface deforms the pattern projected by the light source. As the distortion of the pattern

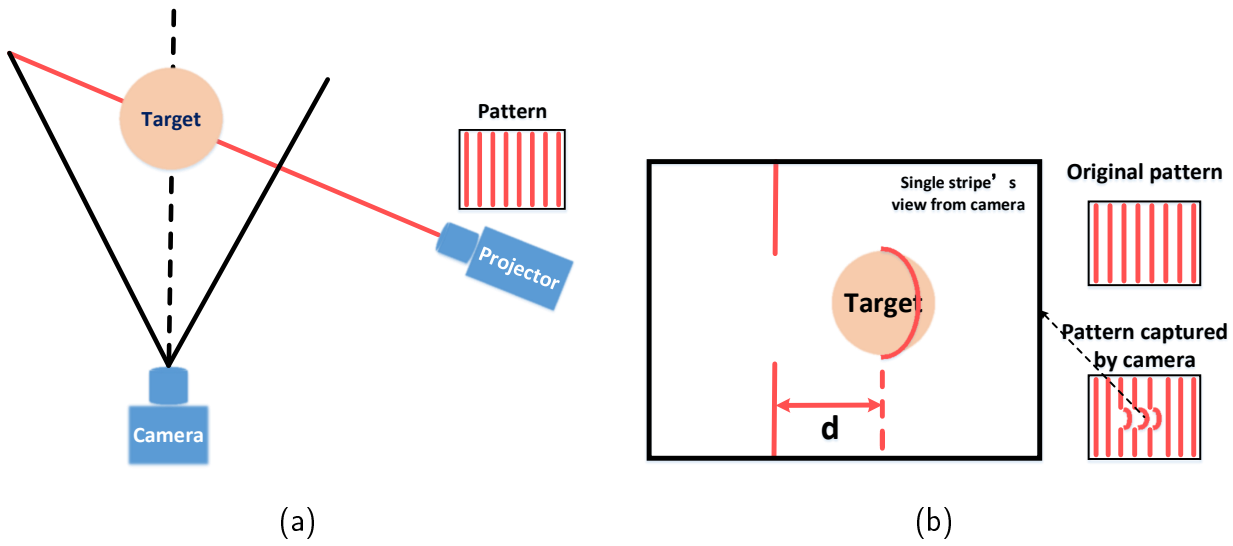


Fig. 2.8: Structured light principle. (a) The configuration of the projector, the camera and the pattern (stripes). (b) The camera's view of one single stripe.

is proportional to how far the non-planar part (the points of the non-planar surface) is from the camera, the depth of each non-planar point (relative to the reference surface) can be calculated based on the triangulation principle. For example, the projector and the camera are positioned at fixed locations to measure a 3D surface, such as a sphere (the target). The pattern is designed as stripes with specific intervals, as shown in Fig. 2.8a. The pattern shown in Fig. 2.8b, is distorted by the target's surface (the sphere's surface). From the two figures, we can see how this pattern works in the camera's view: the position of the single stripe is more to the right, and the position measured is closer to the camera. Based on how the pattern is designed and coded, the structured-light technology can be divided into several categories, including sequential projections, continuous varying patterns, stripe indexing, and grid indexing. Different from the traditional structured-light technology, the pattern applied by the light source does not use 2D image encoding, but 3D image encoding consisting of different 2D image encodings with "light coding" technology. The light source is called a "laser speckle," and the pattern is called a "speckle" pattern, which is actually named for the random speckles that appear after shining a laser at an object. Thus, the positions in a specific 3D space are all marked with different speckle images, as every position has a specific speckle image. With this 3D speckle pattern and a CMOS image sensor, the depth maps of the scene are generated at a low cost in real time [36].

Usually , structured-light sensors are not designed to be high in quality 3D sensors and have a consumer grade price, which makes this kind of sensor suitable for everyday applications. As a result, the sensors usually have a low X/Y resolution and high noise

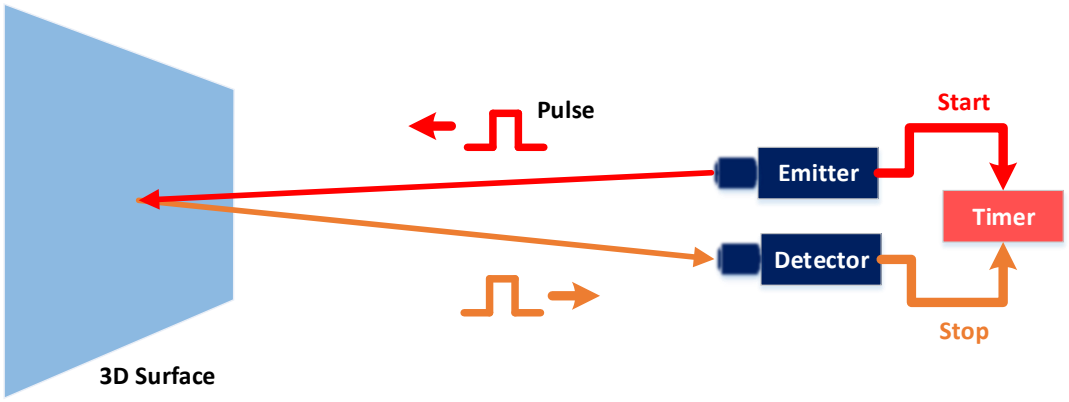


Fig. 2.9: ULTRATRAK PRO system developed with magnetic motion capture by Polhemus.

levels, which always affect their accuracy.

2.2.2 Time-of-Flight (ToF)

By measuring the absolute time cost of the light traveling from a light emitter to the target's surface and being reflected back to the detector—and with the known speed of light in a specific environment and the time cost—the distance can be calculated simply. Based on calculating the time cost (denoted as "T"), the depth sensors can be classified into two categories: 1) Pulsed modulation and 2) continuous-wave (CW) modulation [37]. In pulsed modulation, T is measured directly, which requires that the sending time and receiving time are detected with high accuracy. Thus, to achieve high accuracy in timing, the light pulse needs to be short with fast rise and fall times, and high optical power is required (Fig. 2.9).

In the continuous-wave modulation method, T is not measured directly, but the phase shifts between the light waves leaving the emitter and the light waves arriving at the detector [38] are determined, as shown in Fig. 2.10. As the phase shift is calculated for a distance measurement, the shape of the signals can be specified as sinusoidal, square, and other types of waves. The distance can be formulated as

$$\text{Distance} = \frac{1}{2}cT \tag{2.1}$$

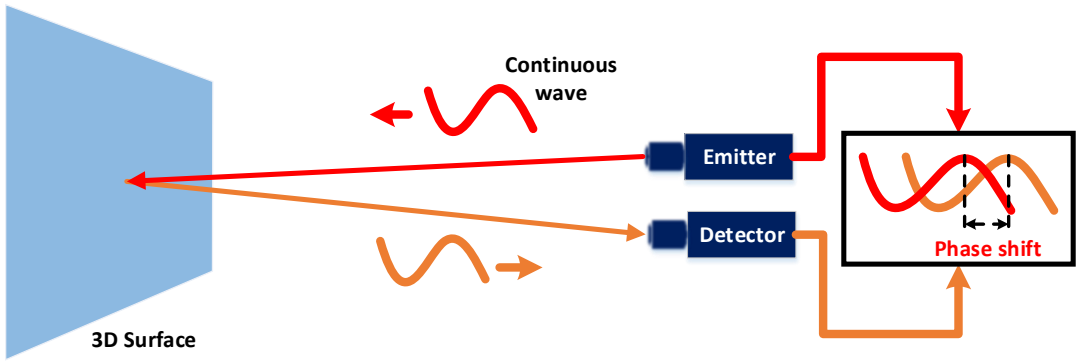


Fig. 2.10: CW (Continuous-wave modulation) measure the distance by calculating the phase shift between the sending light wave and the receiving wave.

where c is the actual speed of light (usually an infrared ray) in a specific environment, and T is the actual time cost of the light's round trip. As the continuous-wave method measures phase shift instead of the time cost directly, T is able to be represented by phase shift φ

$$T = \frac{\varphi}{w} \tag{2.2}$$

where w is the angular frequency specified by the modulation, φ is the phase shift between the light wave leaving emitter and the light wave arriving at the detector, and φ is actually $N + \triangle\varphi$ where N means the number of the signal's half-wavelength within the phase shift, and $\triangle\varphi$ indicates the fraction of a half-wavelength. Whereas N cannot be detected when the light wave arrives at the detector, the measurement range is limited by $\triangle\varphi$ (i.e., the maximum measurement range has to guarantee that the phase shift is no more than one half-wavelength while the light wave arrives at the detector). Usually, to expand the measurement range, multiple frequencies are applied (i.e., multiple modulations or a modulation generating multi-frequency light waves are applied). Though the CW modulation method can measure the distance accurately without requiring a short/strong pulse, it has more noise, and its frame rate is limited by the integration time, as it is not detecting the time cost directly.

2.2.3 Triangulation-based Laser Sensing Device

Triangulation-based laser sensors usually shine a laser on the subject and employ a camera to measure the location of the laser dot, and then, based on how far away the laser strikes

a surface, the laser dot appears at different places in the camera's field of view. This method is called triangulation because the laser dot, the camera, and the laser emitter form a triangle [39]. These kinds of sensors are usually able to acquire high-quality data for building precise 3D object models but are rather expensive compared with other sensors and require expert knowledge to operate. Examples of this kind of sensors are the Cyberware Whole Body Color 3D Scanner, NextEngine desktop 3D scanner, and Creaform's handheld HandyScan scanner [40]. Moreover, users are always required to stand still during the capturing process, which is difficult in some situations, such as sensing 3D models for infants [41].

2.2.4 X-ray Computed Tomography Sensing Devices

Though conventional computed tomography (CT) is a medical imaging technology to generate a 3D image of the inside of an object, industrial X-ray CT can reconstruct a 3D model of the scanned object's external and internal structure from a number of 2D images taken with X-ray radiation in many positions around an axis of rotation [42]. As a nondestructive sensing technology, X-ray CT is able to measure both the outer and inner geometry of a solid object without the need to cut through it or destroy it; X-ray CT can also have rather high resolution or density; for example, the X-View X5000 has the best resolution at 500 nm; another benefit is that it can scan various surfaces, shapes, colors, or materials with certain density and penetrable thickness [42].

X-ray CT comes with the following limitations: (1) it can only sense objects within its maximum penetrable thickness- otherwise, it may result in low-quality X-ray images as the object absorbs too much energy; (2) X-rays are inherently noisy, as are the detector and its amplification, which limits the X-ray CT's performance; (3) scanning multi-material objects may fail if the sensing device cannot detect changes in the material properties [42].

2.3 Kinect depth accuracy assessment and improvement

A variety of approaches have been developed to evaluate the accuracy of Kinect v1. Khoshelham et al. proposed a mathematical model for Kinect v1 depth sensor measurement, and presented a theoretical error analysis which provides an insight into the factors affecting the accuracy of the data [43, 44]. In their model, they used calibration parameters, such as focal length, principal point offsets, lens distortion coefficients, base length, and the distance of the reference pattern, to calculate the 3D coordinates. Their experimental results showed that the random error of depth measurement becomes larger with

the increase in distance between the sensor and the target object. Another contribution of their work is that they adopted a point cloud obtained from a calibrated laser sensor as the ground truth to evaluate the point cloud data reconstructed by Kinect v1 with careful registration.

As Microsoft Corporation has developed a real-time human skeletons recognition system by Kinect v1 [3], some researchers also focused on accuracy evaluation for the human body joints tracking by Kinect. Wheat et al. asked the participants to perform movements (such as reaching and throwing objects), and then compared the detected joints movements by Kinect v1 and by the gold-standard system (a 12 digital-camera capture system) [45]. From the results, they concluded that Kinect v1 can be a potentially valuable motion analysis tool, but certain improvements in accuracy are required. They also tested the feasibility of Kinect v1 as a 3D scanning sensor and whole body tracking device, and obtained good results. Galna et al. [46] measured the relevant movements of subjects with Parkinson's disease using Kinect v1. They found that Kinect v1 detects the timing of movement repetitions accurately, but had varied success in measuring spatial characteristics of movements based on the range of motions.

Improving the accuracy of Kinect also attracts a lot of attentions. In the following part, we summarize previous work from two aspects, including calibrating color-depth sensors and increasing the overall performance with multiple Kinects:

- *Calibration*: Kinect is able to capture both color and depth images. Thus, Herrera et al. and Raposo et al. separately proposed algorithms to calibrate the color-depth camera pair [47, 48]. Their disparity distortion correction model considerably improved the reconstruction accuracy by taking into account color and depth features simultaneously and considering the camera pair system as a whole. With their contribution, 3D models with better accuracy were reconstructed.

- *Multiple Kinects*: As a single sensor may have limitations of resolution, field of view, or accuracy, some researchers used multiple sensors to improve their experimental results. For example, Tong et al. [49] proposed a method using three Kinects simultaneously to scan 3D full human bodies. Their method overcame a single Kinect's comparably low X/Y resolution and depth accuracy drawbacks, which can result in low-quality acquired data. With careful global alignment, they solved the loop-closure problem efficiently.

In this thesis, we propose to apply the trilateration principle with multiple Kinects v2 to improve the overall depth accuracy, which has not been conducted to the best of our

knowlege. The trilateration principle has been widely used in many location-estimation-related research areas. For example, Awad et al. used Received Signal Strength Indicator (RSSI) in Wireless Sensor Networks (WSN) to realize adaptive distance estimation and location based on trilateration concept [50]; Thomas et al. presented an alternative closed-form formulation to improve robot localization results by trilateration [51]; and Bajaj et al. introduced several cost-effective Global Positioning Systems (GPS) which use three or more satellites to determine the target's latitude, longitude, and altitude based on the trilateration principle [52]. In the following, we list the advantages of using trilateration principle in multi-kinect's case:

First, the trilateration principle considers each sensor and its corresponding measured distance as the centre and radius of a sphere, and then calculates these spheres' common interaction point by solving the sphere equation set [53]. As the calculation is mainly based on optimal estimation, the trilateration principle obtains much better position estimates compared with simpliy averaging the measurements from all of the sensors [54].

Second, the solution framework of trilateration is quite flexible, and can be easily extended to the situation with numerous Kinects. Murphy et al. applied the trilateration princple to track trucks in a practical mining application with eight sensors [55].

Third, the redundancy views of multiple sensors can enhance the robustness and accuracy of the measurement results. Both experiments by Le et al. [56] and Beyer et al. [57] validated that additional sensors can reduce measurement errors.

Chapter 3

Kinect Accuracy Assessment

Table 3.1 lists all the hardware coefficients published on the Microsolft Kinect for Windows v2 official website[18]. Besides better performance of the RBG camera and the IR (Infrared) camera, Kinect v2 is different from the former version (v1) in following aspects: Time-of-Flight (TOF) [58], Active IR, 25 joints body tracking, etc.

Table 3.1: Kinect for Windows v2 Sensor Coefficients[18]

Specifications	• ***RGB camera***: 1920×1080 (AKA 1080p) 30 fps 16:9 camera (compared to 640×480 (AKA 480p) 30fps 4:3 for Kinect v1). • ***Field of view***: 70° horizontal and 60° vertical field of view (compared to 57° horizontal and 43 ° vertical field of view for Kinect v1). • ***IR technology***: Active IR for the video camera to see in the dark/low light while no Active IR for Kinect sensor v1's video camera. • ***Depth sensing***: TOF (Time-Of-Flight) depth sensor for 3D tracking (compared to IR structured light depth sensor for Kinect v1) with resolution 512×424 and recommended distance 0.5m-4.5m. • ***Latency***: 20 ms minimum latency (compared to 102 ms minimum latency for Kinect 1 and 50 ms minimum latency for joypads in 60 fps games). • ***Microphone***: 4 microphone array operating at 48 kHz (compared to 4 microphone array operating at 16 kHz for Kinect v1). • ***Adjustable tilt***: Non-motorised manually hand-adjustable-only tilt (compared to motorized manually adjustable via joypad & automatically adjusted tilt for Kinect v1).

[18]http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx

Features	• ***Improved skeletal tracking***: The enhanced fidelity of the depth camera, combined with improvements in the software, have led to several skeletal tracking improvements. In addition to tracking as many as six complete skeletons (compared to two with the original sensor), and 25 joints per person (as compared to 20 with the original sensor), the tracked positions are more accurate and stable. The range of tracking is broader. • ***Finger tracking***: Kinect v2 is able to track 1 finger & thumb on each hand using its default skeletal system and all the fingers using custom system, compared to no finger tracking using default skeletal system and very limited finger tracking using custom system for Kinect v1. • ***Facial expression tracking*** & ***facial recognition***: The facial expression system of Kinect v2 is improved, which overcame the shortcomings of Kinect v1 including limited facial expression tracking and limited facial recognition. • ***Heart rate monitoring***: Kinect v2 is able to monitor the player's heart rate by evaluating the skin color (not possible with Kinect v1). • ***Voice recognition***: Kinect v2 is more advanced than Kinect v1 since the microphone array is with higher quality. • ***Simultaneous multi-app support***: Improved multi-app support enables multiple applications to access a single sensor simultaneously.
Enviroment Requirements	• 64-bit (x64) processor Physical dual-core 3.1 GHz (2 logical cores per physical) or faster processor. • USB 3.0 port dedicated to the Kinect for Windows v2 sensor (Intel and Renesas controllers). • 2 GB of RAM Graphics card that supports DirectX 11. • Windows or Windows Embedded 8 or 8.1

Since 3D data fusion, body tracking, and many other applications are based on the measured depth information, our accuracy assessment in this thesis mainly focus on the depth camera embedded in Kinect v2. Five attributes including accuracy distribution, depth resolution, depth entropy, edge noise and structure noise are evaluated to assess the performance of the depth camera. To localize the cameras' position of Kinect v2, we disassembled the sensor as shown in Fig. 3.2.

To assess the performance of the depth camera embedded in Kinect v2, a 19-inch screen

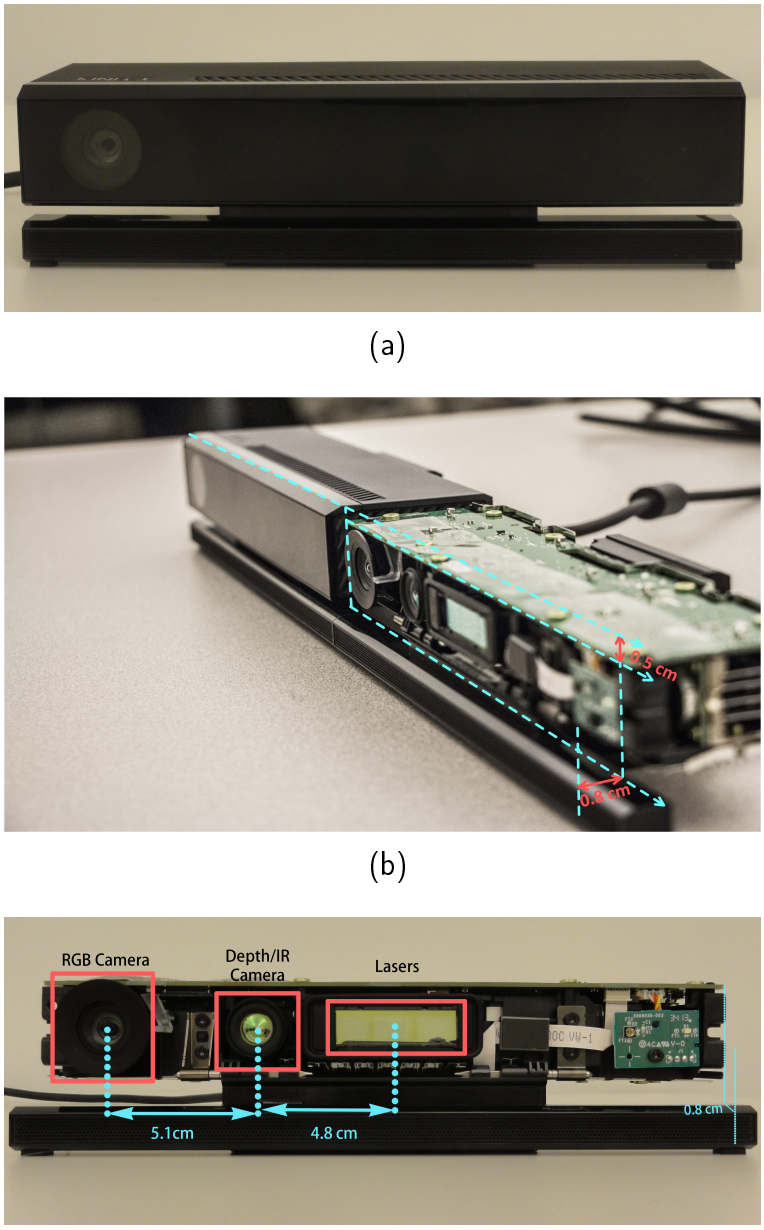


Fig. 3.1: Microsoft Kinect for Windows v2 sensor. (a) Appearance of Kinect v2 (front view). (b) Camera configuration (isometric view). (c) Camera configuration (front view).

is used as a planar surface in the experiments of the following sections for collecting depth data. The screen is positioned perpendicular towards Kinect v2 or pointed with a specific angle towards Kinect v2. To guarantee the aforementioned specific angle (perpendicular case corresponds with 90 degree's case), a AGPtek Handheld Digital Laser Point Distance Meter (measuring range: 40m, accuracy: ±2mm, laser class: class II, laser type: 635nm) was applied. Specifically, four distances between the Kinect front cover and the planar

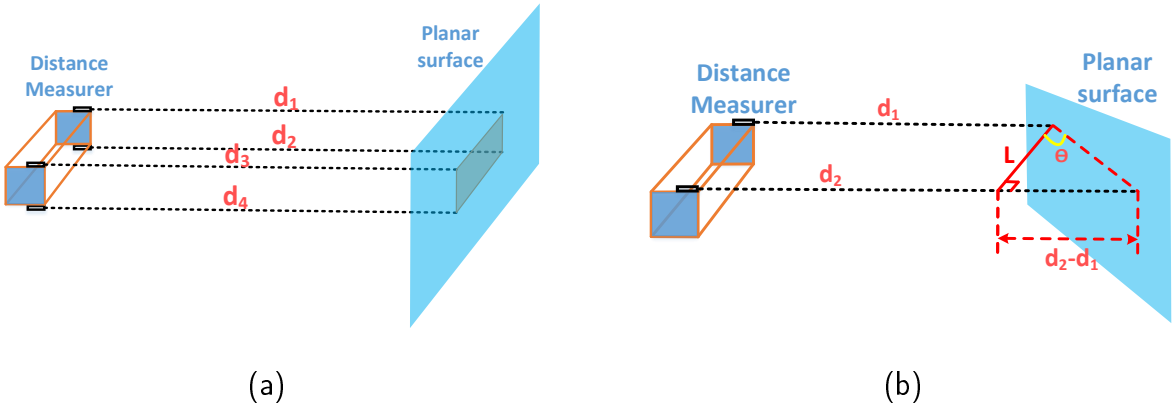


Fig. 3.2: Methods for positioning Kinect sensors accordingly. (a) To guarantee that the Kinect sensor is positioned perpendicular towards the planar surface, four distances between the Kinect front cover and the planar surface are measured with laser distance meter. (b) To guarantee that the Kinect sensor is positioned with specific angle (denoted as θ), two distance and the length (denoted as L) of the Kinect's front cover are measureed.

surface are taken with the mentioned laser distance meter. With these three distances and Kinect size data, the desired angle can be assured using geometrical relation: 1) For perpendicular case, four distances are measured (denoted as d_1, d_2, d_3 and d_4) with laser distance meter and we assume that the Kinect sensor is positioned perpendicular towards the planar surface when $d_1 = d_2 = d_3 = d_4$ is satisfied Fig. 3.2a; 2) For specific angle case, two distances (denoted as d_1 and d_2) and the length of the Kinect sensor's front cover (denoted as L) are measured and we assume that the Kinect sensor is positioned with specific angle (denoted as θ) towards the planar surface when $\tan\theta = \frac{d_2 - d_1}{L}$ is satisfied.

3.1 Accuracy Distribution

Depth accuracy means the difference between the true depth value and the average value of the depth values corresponding with a planar surface located in front of a Kinect v2. As shown in Fig.3.3(a), the depth values inside the light blue circle are captured by a Kinect v2 representing a planar surface and a average depth value is calculated as the center of the circle. The difference between the mentioned average depth value and the true depth value is defined as depth accuracy.

Specifically, by denoting the depth values measured by a Kinect v2 as a set M_d, and the true distance between the planar surface and Kinect as d, the depth accuracy can be computed by the following equation

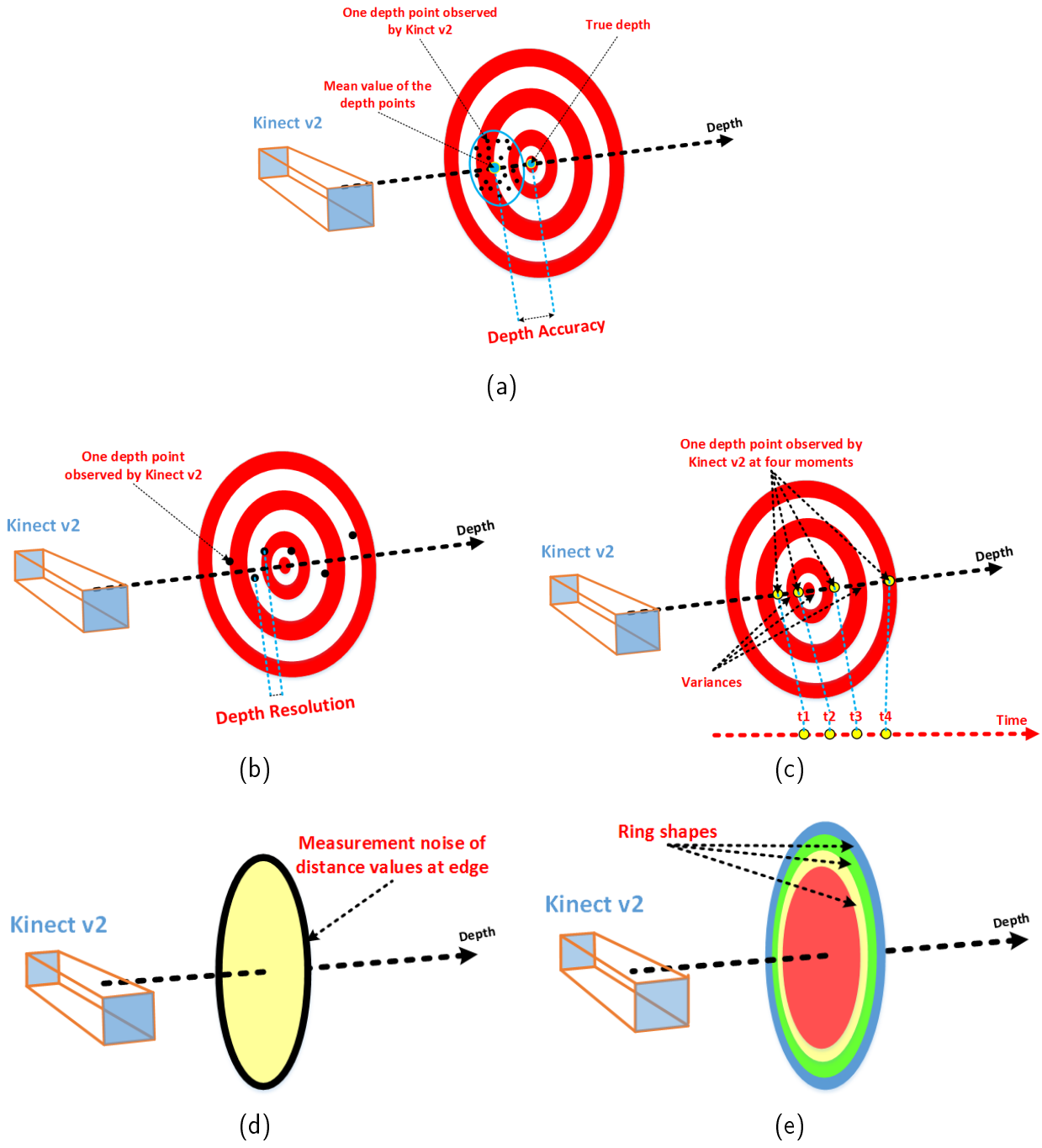


Fig. 3.3: Illustration of accuracy assessment of Kinect v2. (a) Depth accuracy. (b) Depth resolution. (c) Depth entropy. (d) Edge noise. (e) Structural noise. The target plates in (a-c) and (d-e) are parallel and perpendicular with the depth axis, respectively.

$$\text{Depth Accuracy} = d - \text{mean}(M_d) \tag{3.1}$$

where $mean(\cdot)$ computes the mean value of $(\cdot)$. In the experiment, to compute the depth accuracy, a planar surface is placed perpendicular towards Kinect v2 with a specific distance, and the depth value matrix representing the planar surface is obtained by Kinect SDK. Meanwhile, the distance between Kinect v2 and the planar surface is measured by a laser distance meter, which is assumed as true value here.

To illustrate how the depth accuracy differs in the space, a 3D cone is built to show the accuracy distribution. Here the accuracy distribution are evaluated from both horizontal and vertical direction. In the horizontal direction, the planar surface is pointed perpendicular towards the Kinect v2. The planar surface is placed at the distance of 1m, 2m, 3m and 4m away from the Kinect sensor. In the vertical direction, we use a flat wall as the planar surface to evaluate the verticcal accuracy and the Kinect sensor is positioned at the distance of 1m, 2m, 3m and 4m away from the wall. For both directions, at each location, a matrix of distance values in millimeters is exported by Kinect SDK (a Software Development Kit) and then loaded into Matlab for computation and visualization.

3.2 Depth Resolution

Depth resolution means the minimum detectable difference by the Kinect sensor in a certain continuous distance range. Thus, the smaller distance difference being detectable, the more precise depth values being measured, and the higher resolution being achieved. As shown in Fig.3.3(b), there are totally five depth values measured and the minimum difference among the differences between the adjacent depth values is the depth resolution. Specifically, the depth resolution is defined as follows

$$\text{Depth Resolution} = \min |M_d[i+1] - M_d[i]| \tag{3.2}$$

where $M_d[i+1]$ and $M_d[i]$ are two adjacent depth values in M_d. $min(\cdot)$ computes the minimum value of $(\cdot)$. In our set-up, to compute depth resolution, a planar surface is positioned with two specific angle (45 or 60 degrees) towards Kinect v2. The experiment is repeated when the planar surface is positioned at different distances (1m, 1.5m, 2m, 2.5m, 3m, 3.5m and 4m away from the Kinect sensor). At each position, depth values, in the recorded frame, are extracted and processed accordingly.

3.3 Depth Entropy

As limited to the performance of the depth camera, the infrared emitters, the surfaces' materials and the ambient light, the depth values keep changing over time. Even if the sensor's set-up is stationary, a given pixel may vary in several milimetres. As shown in Fig.3.3(c), the depth value at a specific position varies for times. By applying entropy concept, depth entropy is defined to illustrate the stability and reliability of the Kinect sensor with time. Specifically, the depth entropy is defined as follows

$$\text{Depth Entropy}(i) = -\sum_{j} p(i)_j \log_2 p(i)_j \tag{3.3}$$

where i is the index of the depth value in M_d. j is the possible difference values of $M_d(i)$ between two adjacent time frame. $p(i)_j$ is frequency of occurrence of j within the tested time frames. In our experiment set-up, a planar surface is pointed perpendicular towards the Kinect v2 with 1m, 2m, 3m, 4m distance away. The depth values representing the planar surface is exported from Kinect SDK each second and a total of 30 time frames are exported and post-processed to calculate the depth entropy.

3.4 Edge Noise

There exists measurement noise occuring at the edges of an object when the object has a sharp contour. As shown in Fig.3.3(d), a circular planar surface is positioned perpendicular towards the Kinect v2 where the colors represent different depth values measured by the Kinect v2. It is shown that the depth values corresponding with the contour have different values (i.e., measurement noise) compared with that corresponding with the central part. This measurement noise on the sharp contours of an object are defined as edge noise here. In our experiment set-up, a Kinect is pointed perpendicular towards a planar surface where a depth frame is recorded. The experiment is repeated with different locations of the planar surface (1m, 2m, 3m, and 4m away from the Kinect sensor, respectively). According to the depth values obtained from the frames, the edge noise's distributon is visualized and further analyzed on possible reasons.

3.5 Structural Noise

Ideally, the depth values should be constant when a planar surface is pointed perpendicular towards the depth camera, but in reality the camera cannot capture a perfect flat surface. The depth values are distributed in ring shapes with different radii as Fig.3.3(e). This noise phenomenon is defined as structural noise. To assess this noise character, we assume that a wall is even enough and consider it as a flat surface. A Kinect v2 is pointed perpendicular towards the wall for capturing depth data representing the wall. This experiment is repeated with different Kinect v2 locations (1m, 2m 3m and 4m away from the Kinect sensor to the planar surface) to analyze this noise phenomenon.

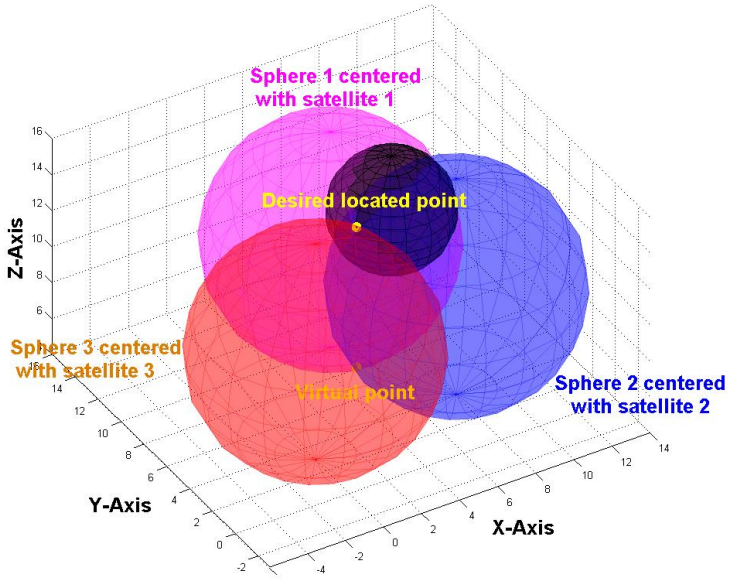
Chapter 4

Multi-Kinect Trilateration

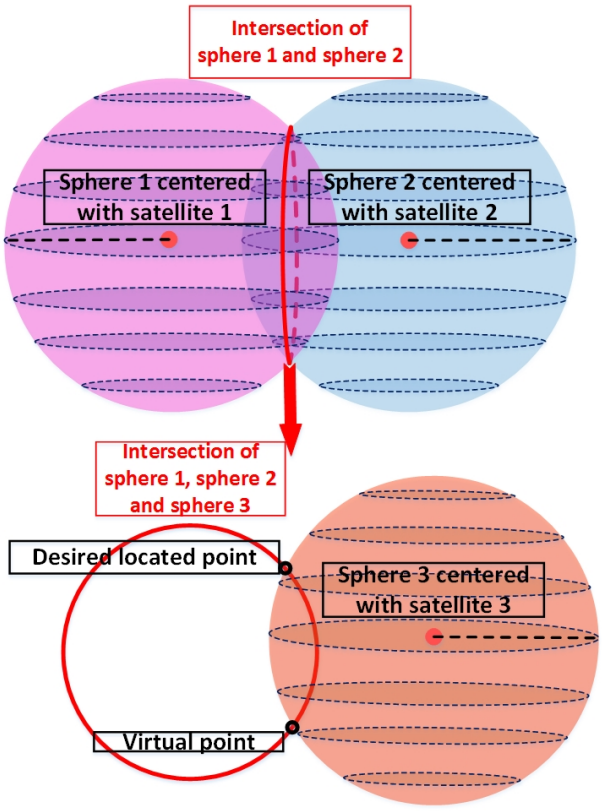
4.1 Trilateration Principle

The basic idea of trilateration method is utilizing multiple sensors to measure the distances between the target and each sensor, and building several spheres by taking each sensor as a center point and its corresponding distance as the radius. Thus, the target position is the common interaction point of these spheres, which can be determined by solving a set of equations representing all the spheres [55]. For example, in GPS, any position on earth is located by accurately measuring the distance from three satellites in space (this is why GPS require 24 stellites around the earth, ensuring at least 3 satellites can observe one position at the same time). In order to locate a position with triangulation, by considering the positions of the three satellites as the centers and the three distances from each satellite to the target position as the radiuses, three spheres can be generated in space (shown in Fig. 4.1(a)). Furthermore, as Fig. 4.1(b) shows, the intersection of the three sphere can be found as two points. One of the two points is the desired localized point, which should be able to be observed on the earth while the other is a virtual point generated by triangulation scheme. Usually the virtual point (such as Position 2 in Fig. 4.1(a)) is easy to be eliminated because of its long distance from earth or a fourth satellite is used to eliminate the wrong position.

In this thesis, different from the trilateration in GPS with satellites, two methods sensors based on trilateration principle for locating the target with multiple Kinect sensors are described: Geometrical method and Least Square method.



(a)



(b)

Fig. 4.1: Trilateration principle. (a) The isometric view of trialateration with three spheres corresponding with three satellites, where the smallest sphere represents the earth. (b) The intersection of two sphere (the red circle) and the intersection points of the three spheres, i.e., the desired localized point and virtual point.

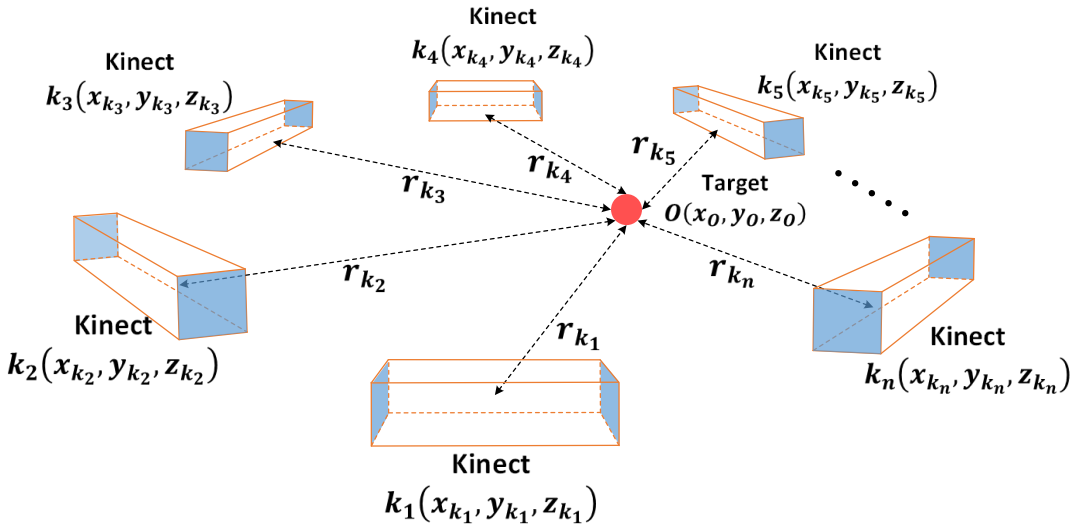


Fig. 4.2: Coordinate setting of the multi-Kinect trilateration.

4.2 Trilateration for Improving Multi-Kinect Accuracy

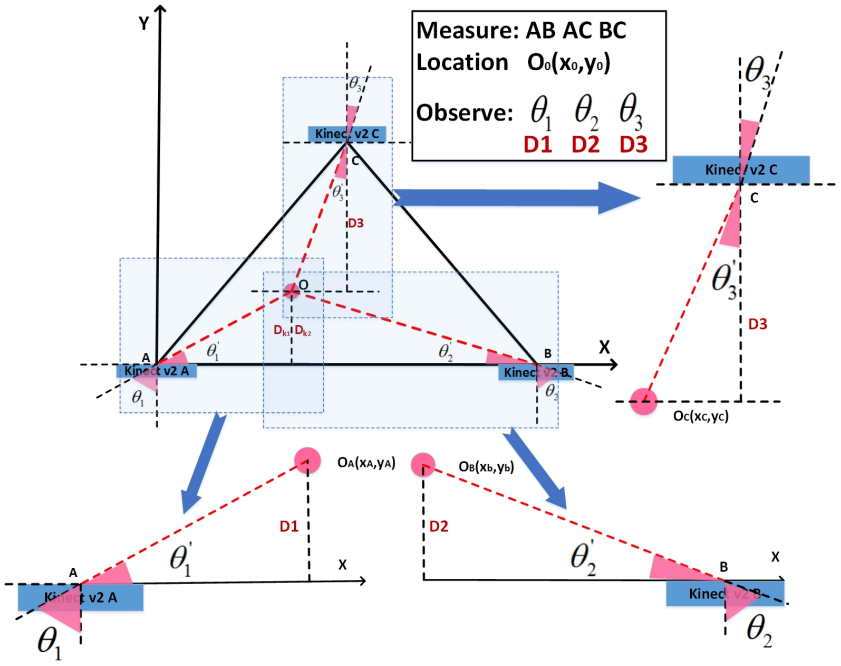
We apply trilateration principle to multiple-Kinect set-up to improve the localization accuracy of a target object when it is placed where the Kinect sensor cannot generate depth values accurately (e.g. distance larger than 4m from the observing sensor). Suppose there are n Kinect v2 sensors positioned accordingly in a specific coordinate system and the target is placed where each Kinect sensor is able to observe it (Fig.4.2). The target position can be calculated by solving the following sphere equations:

$$\begin{cases} (x_o - x_{k_1})^2 + (y_o - y_{k_1})^2 + (z_o - z_{k_1})^2 = r_{k_1}^2 \\ \vdots \\ (x_o - x_{k_i})^2 + (y_o - y_{k_i})^2 + (z_o - z_{k_i})^2 = r_{k_i}^2 \\ \vdots \\ (x_o - x_{k_n})^2 + (y_o - y_{k_n})^2 + (z_o - z_{k_n})^2 = r_{k_n}^2 \end{cases} \tag{4.1}$$

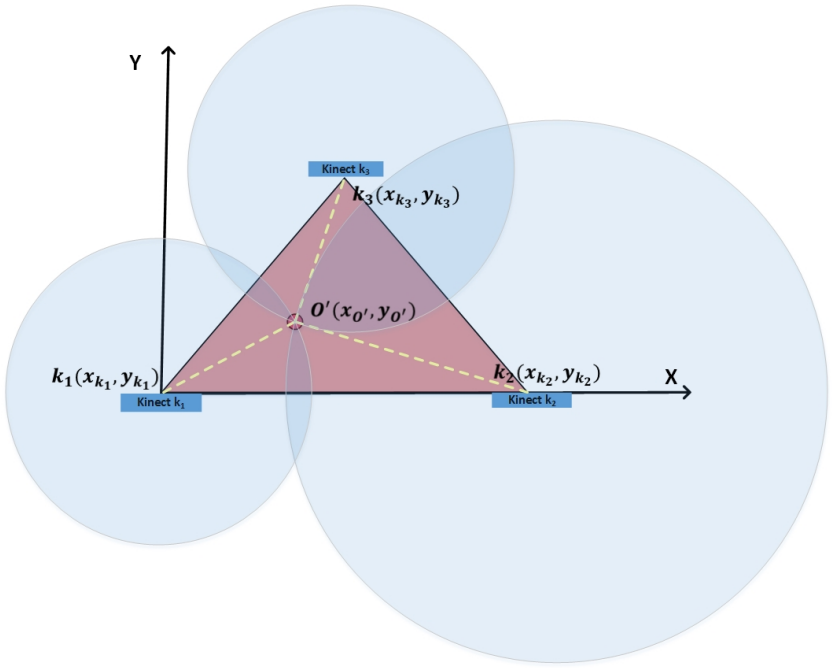
where $(x_{k_i},\ y_{k_i},\ z_{k_i})$ is the location of Kinect k_i, $(x_o,\ y_o,\ z_o)$ is the target position to be solved, r_{k_i} is the range between Kinect k_i and the target position measured by Kinect k_i.

4.2.1 Geometrical Method

As there are only three variables (x_o, y_o and z_o) within n equations, the most direct method to locate the target position is to pick three equations and solve them based on



(a)



(b)

Fig. 4.3: Multi-Kinect trilateration. (a) Trilateration set-up of multi-Kinect within a coordinate system. (b) The intersection point ($O^{'}$) of spheres centered with Kinect k_1, k_2 and k_3.

geometry rules. To simplify the problem, a three-Kinect set-up is given as Fig. 4.3a. In this configuration, the target object is a planar surface located more than 4m away from the three Kinect sensors. The three vertexes (k_1, k_2 and k_3) represent the locations of the three Kinects and O represents the location of the target object. $l_{k_1k_2}$, $l_{k_1k_3}$, $l_{k_2k_3}$ are measured with the aforementioned distance meter in real world, where $l_{k_ik_j} = ||k_i - k_j||_2$ is the distance between Kinect k_i and Kinect k_j. The depth-angle pair (θ_{k_1}, D_{k_1}), (θ_{k_1}, D_{k_2}) and (θ_{k_3}, D_{k_1}) are measured by Kinect k_1, k_2 and k_3, respectively. The depth values D_{k_1}, D_{k_2} and D_{k_3} are calculated by averaging 20×20 pixels' values at the center of the planar surface in the depth image from each Kinect.

The trilateration method for this case study is devided into three steps where the input is the measured parameters of $l_{k_1k_2}$, $l_{k_1k_3}$, $l_{k_2k_3}$ and the observed parameters (θ_{k_1}, D_{k_1}), (θ_{k_2}, D_{k_2}), (θ_{k_3}, D_{k_3}) from Kinect k_1, k_2 and k_3. The output is the optimized position of O (denoted as $O^{'}$). According to geometry rules, the detailed procedures are summarized as follows:

Step 1: We calculate the vertexes' positions ($k_1(x_{k_1}, y_{k_1})$, $k_1(x_{k_2}, y_{k_2})$, $k_1(x_{k_3}, y_{k_3})$) as the centers of three spheres with measured parameters ($l_{k_1k_2}$, $l_{k_1k_3}$, $l_{k_2k_3}$)

$$\begin{cases} x_{k_1} = 0 \\ y_{k_1} = 0 \\ x_{k_2} = l_{k_1k_2} \\ y_{k_2} = 0 \\ x_{k_3} = \sqrt{l_{k_1k_3}^2 - y_{k_3}^2}, \\ y_{k_3} = \frac{2\sqrt{S_1(S_1 - l_{k_1k_2})(S_1 - l_{k_1k_3})(S_1 - l_{k_2k_3})}}{l_{k_1k_2}} \end{cases} \tag{4.2}$$

where $S_1 = \frac{1}{2}(l_{k_1k_2} + l_{k_1k_3} + l_{k_1k_2})$.

Step 2: We calculate the angles and radiuses with parameters exported from each Kinect v2. The angles (from specific perspective of each Kinect v2) are calculated by Equation (4.3)

$$\begin{cases} \theta'_{k_1} = \frac{\pi}{2} - \theta_{k_1} \\ \theta'_{k_2} = \frac{\pi}{2} - \theta_{k_2} \\ \theta'_{k_3} = \theta_{k_3} \end{cases} \tag{4.3}$$

In addition, we compute the three distances from the target object (location of O) to each

Kinect sensor as three radiuses by Equation (4.6)

$$\begin{cases} r_{k_1} = \frac{D_{k_1}}{sin\theta'_{k_1}} \\ r_{k_2} = \frac{D_{k_2}}{sin\theta'_{k_1}} \\ r_{k_3} = \frac{D_{k_3}}{sin\theta'_{k_1}} \end{cases} \tag{4.4}$$

Step 3: The position of O is computed with trilateration method which is denoted as $O'(x_{O'}, y_{O'})$ in Fig. 4.3(b), by solving Equation (4.5)

$$\begin{cases} (x_{O'} - x_{k_1})^2 + (y_{O'} - y_{k_1})^2 + (z_{O'} - z_{k_1})^2 = r_{k_1}^2 \\ (x_{O'} - x_{k_2})^2 + (y_{O'} - y_{k_2})^2 + (z_{O'} - z_{k_2})^2 = r_{k_2}^2 \\ (x_{O'} - x_{k_3})^2 + (y_{O'} - y_{k_3})^2 + (z_{O'} - z_{k_3})^2 = r_{k_3}^2 \end{cases} \tag{4.5}$$

where $z_{k_1} = z_{k_2} = z_{k_3} = 0$, as Kinect v2 k_1, k_2 and k_3 are set at the same horizontal plane. Thus, the target's trilateration result ($x_{O'}$, $y_{O'}$ and $z_{O'}$) can be calculated by solving the three equations (4.5).

4.2.2 Least Square Method

As the sensor number is usually much more than the to-be-determined position variables (i.e., the sensor information is usually redundant), the solution of the above mentioned equation set can be achieved under the least squares sense by applying the pseudo-inverse. This section introduces how to apply least square principle in multi-Kinect trilateration for locating the target. In reality, there is always noise and disturbances in real applications (such as accuracy limit for hardware devices, the clock offsets, etc.). Here, we use $e_{k_i}^2$ to represent the noise coming from i-th Kinect and the Equation (4.1) can be written as

$$\begin{cases} (x_o - x_{k_1})^2 + (y_o - y_{k_1})^2 + (z_o - z_{k_1})^2 - r_{k_1}^2 = e_{k_1}^2 \\ \vdots \\ (x_o - x_{k_i})^2 + (y_o - y_{k_i})^2 + (z_o - z_{k_i})^2 - r_{k_i}^2 = e_{k_i}^2 \\ \vdots \\ (x_o - x_{k_n})^2 + (y_o - y_{k_n})^2 + (z_o - z_{k_n})^2 - r_{k_n}^2 = e_{k_n}^2 \end{cases} \tag{4.6}$$

Hence, we can obtain an optimal position of the target by minimizing the sum of the errors $e_{k_i}^2$ $(1 \leq i \leq n)$

$$(x_o,\ y_o,\ z_o) = \arg\min(\sum_{i=1}^{n} e_{k_i}^2) \tag{4.7}$$

From Equation (4.1), for the i-th Kinect, the sphere equation containing the range measurement r_{k_i} and its position coordinate $(x_{k_i}, y_{k_i}, z_{k_i})$ is

$$(x_o - x_{k_i})^2 + (y_o - y_{k_i})^2 + (z_o - z_{k_i})^2 = r_{k_i}^2 \tag{4.8}$$

where $i = 1, 2, \cdots, n$. By adding and subtracting x_{k_j}, y_{k_j} and z_{k_j} $(j = 1, 2, \cdots, n,\ j \neq i)$ to the above equation, we have

$$(x_o - x_{k_j} + x_{k_j} - x_{k_i})^2 + (y_o - y_{k_j} + y_{k_j} - y_{k_i})^2 + (z_o - z_{k_j} + z_{k_j} - z_{k_i})^2 = r_{k_i}^2 \tag{4.9}$$

After expanding and merging terms, we obtain

$$\begin{aligned} &(x_o - x_{k_j})(x_{k_i} - x_{k_j}) + (y_o - y_{k_j})(y_{k_i} - y_{k_j}) + (z_o - z_{k_j})(z_{k_i} - z_{k_j}) \\ &= \frac{1}{2}[(x_o - x_{k_j})^2 + (y_o - y_{k_j})^2 + (z_o - z_{k_j})^2 - r_{k_i}^2 + (x_{k_i} - x_{k_j})^2 + (y_{k_i} - y_{k_j})^2 + (\mathrm{z}_{k_i} - z_{k_j})^2] \\ &= \frac{1}{2}(r_{k_j}^2 - r_{k_i}^2 + l_{k_i k_j}^2) = b_{ij} \end{aligned} \tag{4.10}$$

where $l_{k_i k_j}$ is the distance between Kinect k_i and Kinect k_j, i.e.

$$l_{k_i k_j} = ((x_{k_i} - x_{k_j})^2 + (y_{k_i} - y_{k_j})^2 + (\mathrm{z}_{k_i} - z_{k_j})^2)^{\frac{1}{2}} \tag{4.11}$$

For Equation (4.10), let $i = 2, \cdots, n$, $j = 1$ and thus, Equation 4.1 is reformulated as $n-1$ linear equations with three unknown variables (x_o, y_o and z_o)

$$\begin{cases} (x_o - x_{k_1})(x_{k_2} - x_{k_1}) + (y_o - y_{k_1})(y_{k_2} - y_{k_1}) + (z_o - z_{k_1})(z_{k_2} - z_{k_1}) = \frac{1}{2}(r_{k_1}^2 - r_{k_2}^2 + l_{k_2 k_1}^2) = b_{21} \\ (x_o - x_{k_1})(x_{k_3} - x_{k_1}) + (y_o - y_{k_1})(y_{k_3} - y_{k_1}) + (z_o - z_{k_1})(z_{k_3} - z_{k_1}) = \frac{1}{2}(r_{k_1}^2 - r_{k_3}^2 + l_{k_3 k_1}^2) = b_{31} \\ \vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ (x_o - x_{k_1})(x_{k_n} - x_{k_1}) + (y_o - y_{k_1})(y_{k_n} - y_{k_1}) + (z_o - z_{k_1})(z_{k_n} - z_{k_1}) = \frac{1}{2}(r_{k_1}^2 - r_{k_n}^2 + l_{k_n k_1}^2) = b_{n1} \end{cases} \tag{4.12}$$

which can be written in the form of a general linear equation set

$$\mathbf{Ax} = \mathbf{b} \tag{4.13}$$

where

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} x_{k_2} - x_{k_1} & y_{k_2} - y_{k_1} & z_{k_2} - z_{k_1} \\ x_{k_3} - x_{k_1} & y_{k_3} - y_{k_1} & z_{k_3} - z_{k_1} \\ \vdots & \vdots & \vdots \\ x_{k_n} - x_{k_1} & y_{k_n} - y_{k_1} & z_{k_n} - z_{k_1} \end{bmatrix}, \\ \mathbf{x} &= \begin{bmatrix} x_o - x_{k_1} \\ y_o - y_{k_1} \\ z_o - z_{k_1} \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} b_{21} \\ b_{31} \\ \vdots \\ b_{n1} \end{bmatrix} \end{aligned} \tag{4.14}$$

Equation (4.13) can be solved by

$$\mathbf{x} = \mathbf{A}^{+}\mathbf{b} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \tag{4.15}$$

where $\boldsymbol{A}^{+}$ is the pseudoinverse of matrix $\boldsymbol{A}$. According to matrix theory, the above solution provides the optimial target position under the sense of least squres. In other words, this solution provides the best estimation of the target position which minimizes the sum of the residual errors coming from the sphere equations [59]. It is noted that, if the matrix $\boldsymbol{A}^T\boldsymbol{A}$ is singular or ill-conditioned, singular value decomposition (SVD) can be applied to calculated the peudo-inverse of $\boldsymbol{A}$.

The same three-Kinect set-up is given as Fig. 4.3 for showing details about how least square principle is applied for locating the target with trilateration principle. Conducting the similar steps (step 1-3 for calculating each Kinect sensor's location and obtaining the depth-angle pair from each Kinect sensor) mentioned in Section 4.2.1, we have the spheres' equations (4.16).

$$\begin{cases} (x_{O''} - x_{k_1})^2 + (y_{O''} - y_{k_1})^2 + (z_{O''} - z_{k_1})^2 = r_{k_1}^2 \\ (x_{O''} - x_{k_2})^2 + (y_{O''} - y_{k_2})^2 + (z_{O''} - z_{k_2})^2 = r_{k_2}^2 \\ (x_{O''} - x_{k_3})^2 + (y_{O''} - y_{k_3})^2 + (z_{O''} - z_{k_3})^2 = r_{k_3}^2 \end{cases} \tag{4.16}$$

where ($x_{O''}$, $y_{O''}$ and $z_{O''}$) is the target's position to be solved with least squares principle and $z_{k_1} = z_{k_2} = z_{k_3} = z_{O''} = 0$, as Kinect v2 k_1, k_2, k_3 and the target's position are set

at the same horizontal plane. As it is clarified in Section 4.2, the three equations can be reformulated in matrix form (Equation (4.13)) where

$$\mathbf{A} = \begin{bmatrix} l_{k_1k_2} & 0 \\ \sqrt{l_{k_1k_3}^2 - y_{k_3}^2} & \frac{2\sqrt{S_1(S_1-l_{k_1k_2})(S_1-l_{k_1k_3})(S_1-l_{k_2k_3})}}{l_{k_1k_2}} \end{bmatrix}, \\ \mathbf{x} = \begin{bmatrix} x_{O'} \\ y_{O'} \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} \frac{1}{2}(r_{k_1}^2 - r_{k_2}^2 + l_{k_1k_2}^2) \\ \frac{1}{2}(r_{k_1}^2 - r_{k_3}^2 + l_{k_1k_3}^2) \end{bmatrix} \tag{4.17}$$

4.2.3 Verification of Multi-Kinect Trilateration

To verify the trilateration methods indeed improve the overall Kinect measurements, three calculated positions of O (i.e., $O_{k_1}(x_{O_{k_1}}, y_{O_{k_1}})$, $O_{k_2}(x_{O_{k_2}}, y_{O_{k_2}})$, $O_{k_3}(x_{O_{k_3}}, y_{O_{k_3}})$) in the same configuration in Fig. 4.3a are considered

$$\begin{cases} O_{k_1} : x_{O_{k_1}} = \frac{D_{k_1}}{tan\theta'_{k_1}}, y_{O_{k_1}} = D_{k_1} \\ O_{k_2} : x_{O_{k_2}} = x_{k_2} - \frac{D_{k_2}}{tan\theta'_{k_2}}, y_{O_{k_2}} = D_{k_2} \\ O_{k_3} : x_{O_{k_3}} = x_{k_3} - D_{k_3} \cdot tan\theta'_{k_1}, y_{O_{k_3}} = y_{k_3} - D_{k_3} \end{cases} \tag{4.18}$$

where O_{k_1}, O_{k_2} and O_{k_3} are calculated solely based on k_1, k_2 and k_3, respectively. For example, $O_{k_1}(x_{O_{k_1}}, y_{O_{k_1}})$ is calculated with the Kinect k_1's position $k_1(x_{k_1}, y_{k_1})$ and its corresponding depth-angle pair (θ_{k_1}, D_{k_1}) by geometry rules. Thus, totally four calculated positions of O (i.e., O_{k_1}, O_{k_2}, O_{k_3} and O') are compared for verifying that the trilateration result (O') is better than the results based on one Kinect sensor (O_{k_1}, O_{k_2} or O_{k_3}).

Chapter 5

Results and Discussion

In this chapter, the results of the attributes defined in Chapter 3 for accessing depth accuracy of Kinect for Windows v2 sensor is given and discussed. The results of the experiment designed in Chapter 4 for verifying that the two multi-Kinect trilateration methods improve the accuracy of Kinect sensor is given and discussed.

5.1 Accuracy Evaluation

5.1.1 Accuracy Distribution

Each frame is recorded when the planar surface is positioned at one of the key positions (Fig. 5.2). Totally there are 21 key positions in the horizontal plane and 19 positions in the vertical plane. Based on the depth values representing the screen in each recorded depth frame, the mean depth value, standard deviation and depth range are calculated and visualized. For example, Fig. 5.1 shows the planar surface when being pointed perpendicular towards Kinect v2 (the central key position at 2m). The x and y axis represent horizontal and vertical axis of the frame (the depth frame contains a 512×424 pixel matrix where x range from 1 to 512 and y range from 1 to 424) and z axis represent the depth value of the corresponding pixel. It is shown that the depth values range from 1996mm to 2004mm here. The mean depth value and its standard deviation are 1999.8mm and 1.2mm, respectively. This result is obtained from one conduction of the aforementioned experiment in Section 3. The mean value and standard deviation might varies if more experiments are conducted under the same configuration. But the depth accuracy defined in Section 3 would be still within the range of our proposed 3D accuracy distribution map.

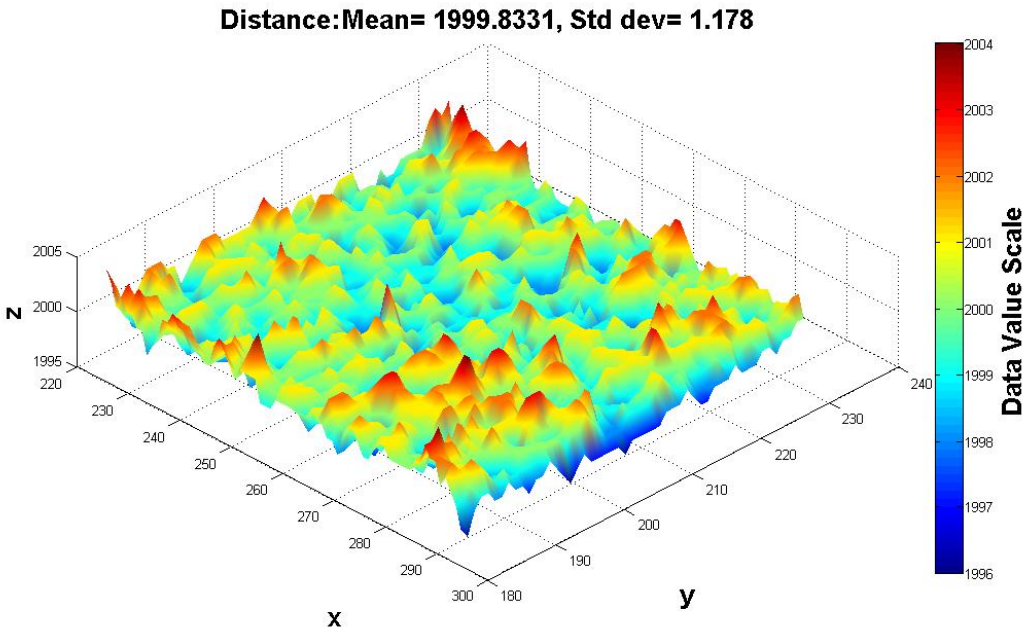


Fig. 5.1: Depth values' distribution representing the planar surface (70×50 pixels)

By summarizing the mentioned measurement accuracy of Kinect at all the key positions, we find that the accuracy error distribution of Kinect v2 satisfies an elliptical cone with 60 degrees' angle in vertical direction and 70 degrees' angle in the horizontal direction (Fig. 5.2). Moreover, we use three colors (i.e., green, yellow and red) to indicate the different accuracy areas by dividing the space of accuracy error distribution into three regions. Specifically, in the green, yellow and red areas of both the horizontal and vertical plane, the average accuracy is less than 2mm, between 2mm and 4mm, and more than 4mm, respectively.

5.1.2 Depth Resolution

As Chapter 3 mentioned, the planar surface is pointed with two specific angles (45 degrees and 60 degrees) towards Kinect v2 at different distances. For each recorded depth frame, a depth difference between every two adjacent pixels is calculated and the mean resolution is computed by averaging all the mentioned depth difference between each adjacent pixels in the whole depth image. Similarly, the standard deviation and max resolution are also calculated. The tendency of mean resolution, max resolution and standard deviation are shown in Fig. 5.3 when the distance is increasing, where the horizontal axis represents the distance between the planar surface and the Kinect v2. Here, the solid line and the dash line correspond to the tendency when the planar surface is inclined by 45 and 60 degrees

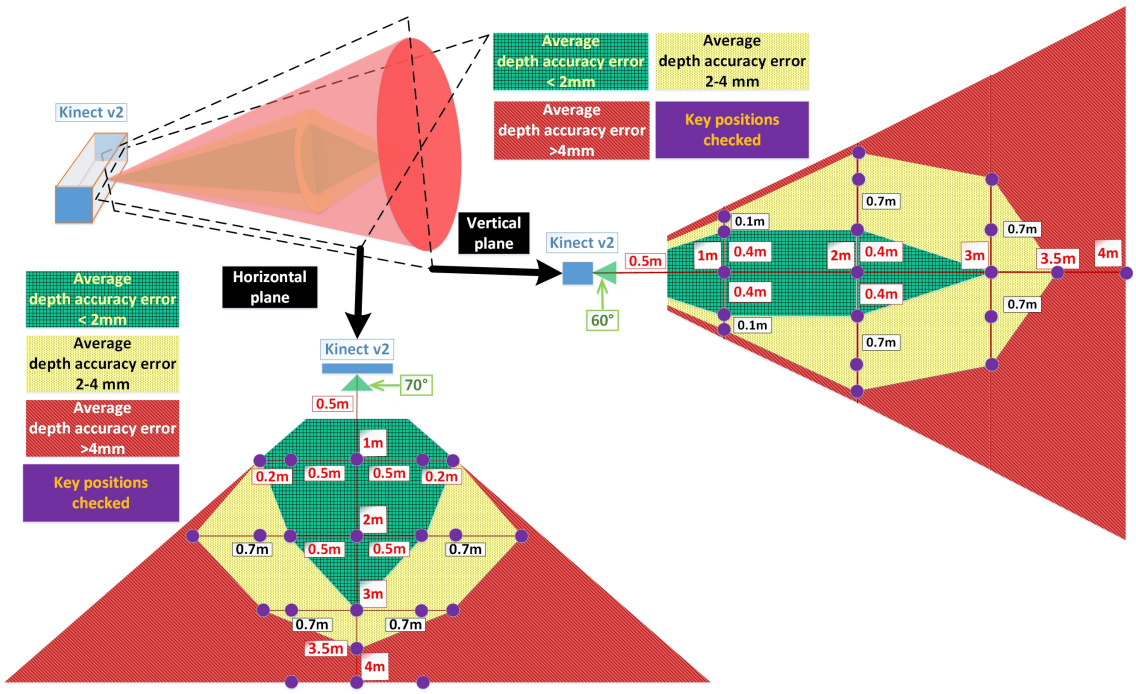
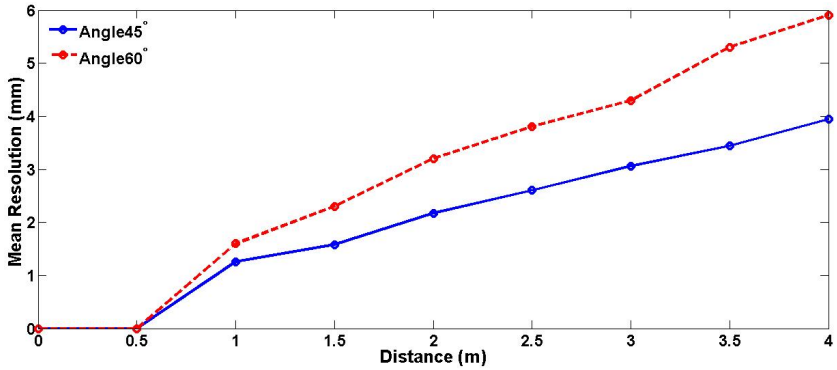


Fig. 5.2: Accuracy error distribution of Kinect for Windows v2.

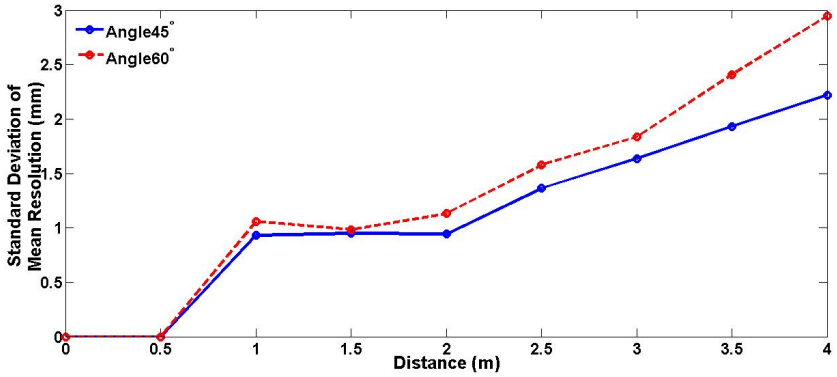
from Kinect v2, respectively. As Kinect v2 is not able to measure the depth within 0.5m, the resolution within 0.5m is set as 0. It is shown that: 1) the mean depth resolution and max resolution increase with distance, meaning that a coarser depth image is made with distance increase; 2) A larger tilt angle leads to a lower depth resolution and a larger standard deviation; 3) When the distance increases larger than 2m, the max resolution and standard deviation (for both 45 and 60 degrees tilt) increase faster.

5.1.3 Depth Entropy

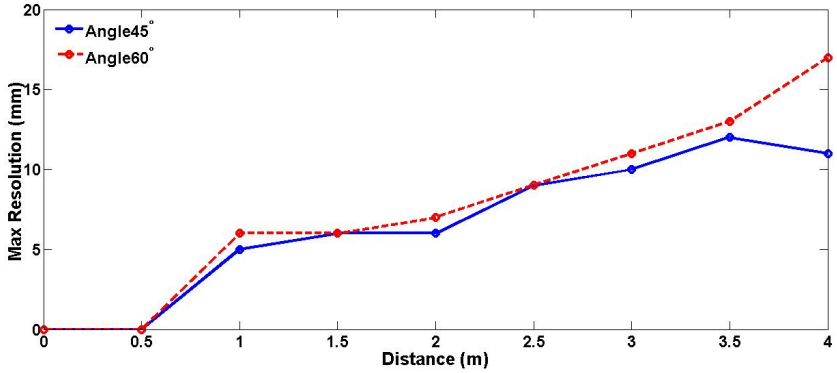
To illustrate the phenomenon that the depth value of each pixel keeps varying within 6mm over time, at each position of the planar plane (1m, 2m, 3m or 4m away from the Kinect v2), totally 30 depth frames are recorded. For these recorded 30 frames, the sizes of each frame's pixel matrix representing the planar surface are the same, but the distribution of the pixels' values is different (depth value keep varying over time). For each two adjacent frames of the 30 frames, every variance between the two pixels at the same position of the two pixel matrixes is calculated, which means totally 29 variances are calculated for one specific position of the 30 pixel matrixes. Based on the calculated variance, an entropy distribution figure can be built using Equation (3.3). We found there is no apparent singularity in these entropy distribution figures. For example, Fig. 5.4(a)



(a)



(b)



(c)

Fig. 5.3: Resolution tendency with the Kinect's location and attitude. (a) Mean resolution tendency. (b) Standard deviation of the resolution tendency. (c) Max resolution tendency.

shows the entropy distribution when the planar plane is 2m away from the Kinect v2 where each pixel's value represents a specific corresponding pixel's entropy on the depth frames. Furthermore, the general tendency of mean entropy tendency and its standard deviation are analyzed in Fig. 5.4 (b-c)) where the horizontal axis represents the distance between the Kinect v2 and the planar plane. It is shown that 1) the standard deviation of the entropy

decreases when the distance increases because the number of the pixels representing the screen decreases with the distance increases; 2) the mean entropy increases fast after the distance is beyond 2m.

5.1.4 Edge Noise

In the real applications, wrongly measured pixels by Kinect v2 can lead to abnormal contours. Here, we use this phenomenon to evaluate the edge noise of Kinect v2. Specifically, the planar plane is set in front of the Kinect v2 and the contours of the plane is analyzed. Fig. 5.5 shows a depth frame when the planar plane is located 1m away from the Kinect v2 where x and y axis represent the horizontal and vertical index of the frame, respectively.

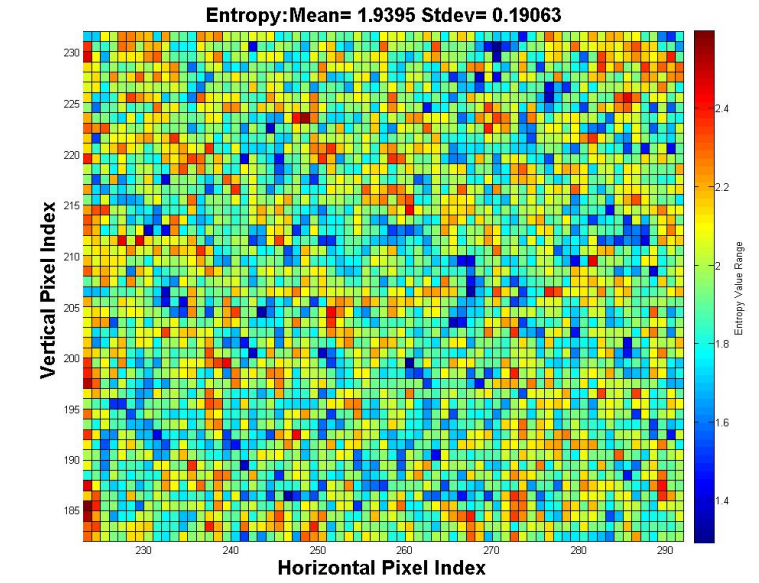
We observe that zero-value contours existing around the planar plane's edges (shown as deep blue contour). According to the manual of Kinect v2, a zero depth value means the position observed is out of Kinect sensor's observation range, but apparently the edges of this planar is within the measurable range (0.5m-4m). Here, the width of the zero pixels is 1~2 pixels (roughly 2-4mm) when the planar surface is 1m away from Kinect v2. According to our experiments, all the objects, with at least 0.5m distance from its background, have zero contours with a maximum width of 2 pixels. It is noted that there is no apparent increase of the width of the zero value contour when the distance from the Kinect v2 to object increases (from 0.5m to 4m).

5.1.5 Structural Noise

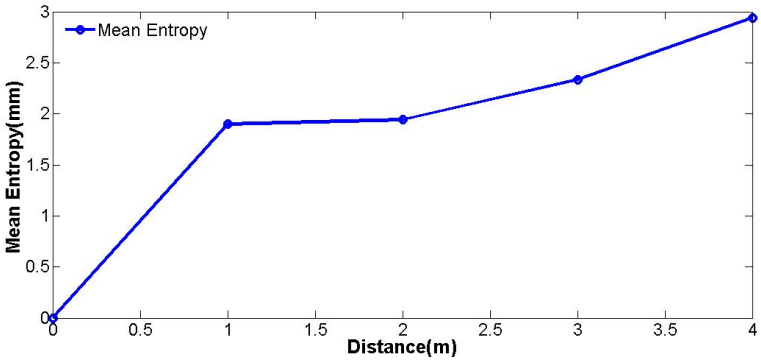
Kinect v2 has structural noise when capturing depth images. To analyze this issue, we test the measured flat plane in front of Kinect at 1m distance as shown in Fig. 5.6 where x and y axis represent the horizontal and vertical index of the frame respectively. It can be observed that the recorded depth values are distributed in the shape of rings, i.e., the depth values of pixels decrease when their distance to the central part increases. The reason of this ring phenomenon is hard to understand. However, we think it might be due to the diffraction coming from the random variance of the depth pixels mentioned in Subsection 3.3.

5.2 Trilateration for Multi-Kinect Result

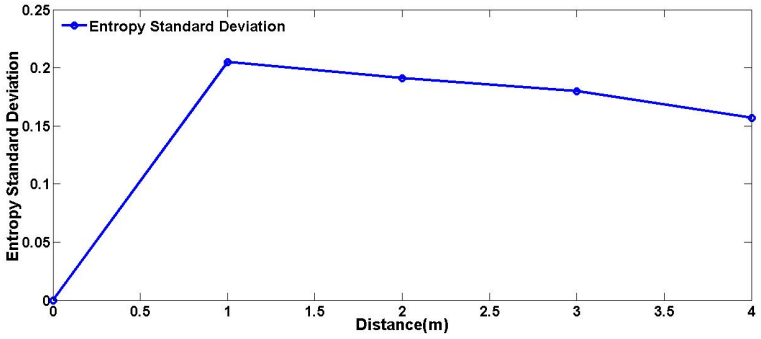
As the multi-Kinect trilateration setup described in Chapter 4 , three Kinect v2 are positioned at each vertex of a isosceles triangle. Without loss of generality, we set the location



(a)



(b)



(c)

Fig. 5.4: Depth entropy of depth pixels. (a) Entropy distribution. (b) Mean entropy. (c) Standard deviation of the mean entropy.

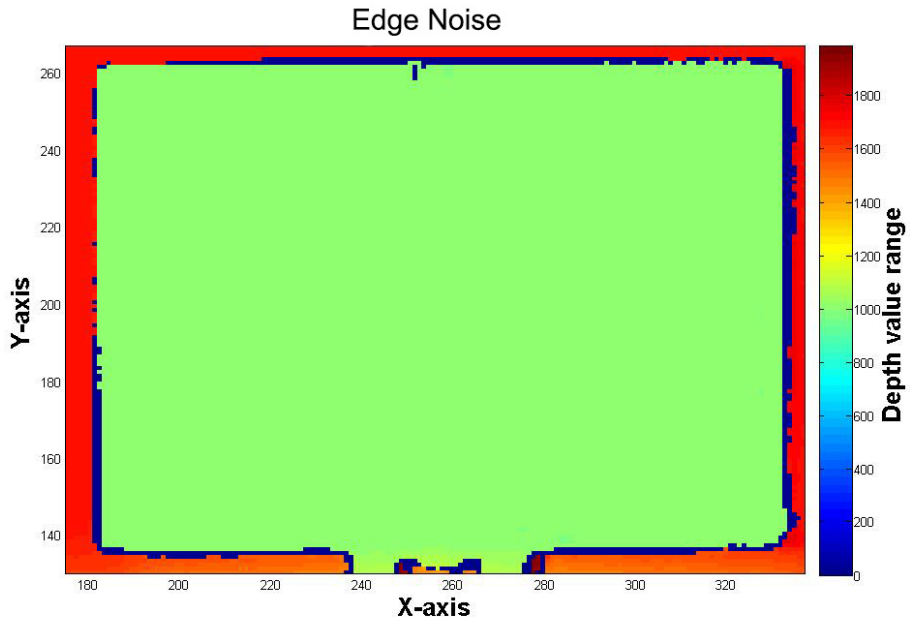


Fig. 5.5: Zero-value contour of the measured planar plane when being set 1m from the Kinect v2.

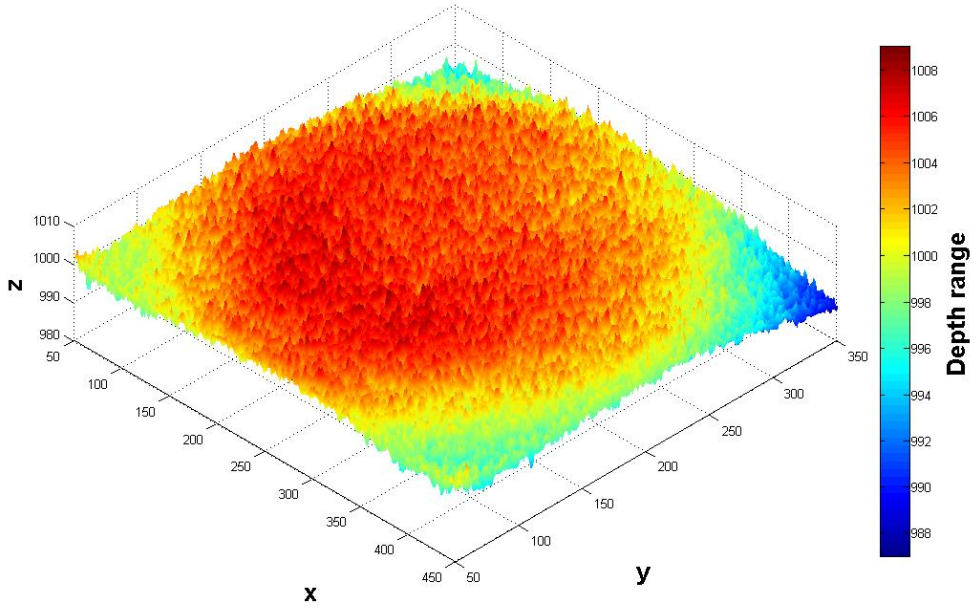


Fig. 5.6: Ring shape of the captured planar plane when being set 1m from the Kinect v2 (450×400 pixels).

of O in front of Kinect v2 k_3. The radiuses (r_1, r_2 and r_3) and the vertexes positions ($k_1(x_{k_1}, y_{k_1})$, $k_2(x_{k_2}, y_{k_2})$, $k_3(x_{k_3}, y_{k_3})$) are calculated first and then the "trilaterate" position ($O^{'}$) are computed by the steps described in Section 4.2 as shown in Table 5.1.

As described in Chaper 4, the geometrical method and the least square method are applied with the same three-Kinect setup. The trilateration results of the two methods are denoted as $O^{'}$ ($x_{O'}$, $y_{O'}$, $z_{O'}$) and $O^{''}$ ($x_{O''}$, $y_{O''}$, $z_{O''}$), respectively. In our experiment, the two results are exactly the same ($x_{O'} = x_{O''}$, $y_{O'} = y_{O''}$ and $z_{O'} = z_{O''} = 0$). This is because the three-Kinect setup gives three sphere equations, which means there is not a fourth equation for least squares method to minimize the error (there is no redundant data for the least squares to minimize the noise). Hence, we can see that the results of the geometrical method and the least square method are the same when a three-Kinect set-up is applied.

Table 5.1: Localization Comparison between by Single Kinect and by Multi-Kinect Trilateration

Method	Measured/ Calculated Position of O	Measurement/Calculation Error (Difference between O_o and the Measured/Calculated Position of O)
Measurement from Kinect k_1	O_{k_1} (2197.1, 4016.4)	$\|O_0 - O_{k_1}\| = 64.23\,mm$
Measurement from Kinect k_2	O_{k_2} (2057.4, 3985.0)	$\|O_0 - O_{k_2}\| = 78.99\,mm$
Measurement from Kinect k_3	O_{k_3} (2135.0, 3989.4)	$\|O_0 - O_{k_3}\| = 10.66\,mm$
Multi-Kinect Trilateration: Geometry method	$O^{'}$ (2156.4, 4013.9)	$\|O_0 - O^{'}\| = 25.61\,mm$
Multi-Kinect Trilateration: Least Square method	$O^{''}$ (2156.4, 4013.9)	$\|O_0 - O^{''}\| = 25.61\,mm$

* The measured position of O_o is (2135, 4000).

To verify the validity of the multi-Kinect trilateration, the position of O (denoted as O_o) is measured. $|O_{k_1} - O_o|$, $|O_{k_2} - O_o|$ and $|O_{k_3} - O_o|$ are compared with $|O^{'} - O_o|$ where O_{k_1}, O_{k_2} and O_{k_3} are obtained solely based on the measurements from Kinect k_1 to Kinect k_3, respectively. $O^{'}$ is computed based on the measurements from the three Kinects. The four mentioned measurement errors are compared, which shows that the position of O observed by Kinect k_3 has small measurement error (10.66 mm); the position of O observed by Kinect k_1 and k_2 has large measurement error (64.23 mm for Kinect k_1 and 78.99 mm for Kinect k_1); while the position of O observed by the proposed multi-Kinect trilateration method has the measurement error in between, respectively.

The measurement of O from Kinect k_3 has only 10 mm error because it is set straight

towards the planar surface while the other two Kinects are set with a 60-degree angle towards the plane. Although the measurement of Kinect k_3 is more accurate here, the observed object cannot be always at the central area of the Kinect sensor's view range. In this case, the trilateration method has an overall good performance. Besides, the measurement result computed by multi-Kinect trilateration is better than the mean measurement error (51.22 mm) of the three Kinect sensors.

Chapter 6

Human Gait Tracking with Multi-Kinect

In this chapter, the details about the implementation of a human gait-tracking system are given. This system applies the trilateration principle mentioned in Chapter 3 to track the joints of human legs. In total, six joints are tracked by three Kinect sensors that are positioned accordingly. The three Kinect sensors are connected to three computers, and the computers are connected to a server. According to the positions of each Kinect sensor and the data obtained, the trilateration procedure is executed in real time. The user needs to walk for a few steps inside the triangle with three Kinect sensors as the vertexes for collecting enough data required by trilateration. Then, six joints of the user's legs are virtualized in 3D.

6.1 Configuration

6.1.1 Hardware

Generally, according to the environment requirement by the Kinect v2 sensor (shown in Table 3.1), computers with high performance CPUs and graphic cards are required. To apply trilateration, at least three Kinect sensors are required as along with three computers. To connect three computer to a server, a router is necessary. The specific hardware configurations are shown in Table 6.1.

Table 6.1: Hardware Configuration

Hardware	Quantity	Specifications
Client/ Server	3	• ***Model***: Lenovo Y50 • ***CPU***: Intel(R) Core(TM) i7-4710HQ CPU @2.50 GHz • ***RAM***: 8.00 GB • ***GPU***: NVIDIA GeForce GTX 860M
Router	1	• ***Model***: Linksys E1200 • ***Technology***: Wireless-N • ***Ethernet ports***: 4 • ***Speed***: Up to 300 Mbps
Tracking sensor	3	• ***Model***: Kinect for Windows v2 sensor • ***RGB camera***: 1920×1080 (AKA 1080p) 30 fps 16:9 camera. • ***Field of view***: 70horizontal and 60vertical field of view. • ***IR technology***: Active IR for the video camera to see in the dark/low light. • ***Depth sensing***: TOF (Time-Of-Flight) depth sensor for 3D tracking with resolution 512×424 and recommended distance 0.5m-4.5m. • ***Latency***: 20 ms minimum latency. • ***Adjustable tilt***: Non-motorised manually hand-adjustable-only tilt.
Tripod	3	• ***Model***: DYNEX DX-TRP60 • ***Weight***: 4 pounds. • ***Size***: Extends to 60 inches. • ***Head type***: Pan head. • ***Material***: Aluminium.

6.1.2 Software

Several libraries and software programs are used to help build the gait-tracking system: 1) Virtual Studio 2013 is used as the C++ integrated development environment (IDE); 2) Boost c++ libraries are used to realize a local network environment, a time synchronization service, and data serialization service; 3) OpenGL library is used to visualize the human legs; and 4) Kinect SDK v2.0 is used to help obtain the source data from each Kinect v2 sensor (shown in Table 6.2).

Table 6.2: Software Configuration

Component	Software configuration
Computer/ Kinect Client	• Virtual Studio 2013 • Boost C++ Libraries v1.56.0 • Kinect SDK v2.0 (public version) • Windows 8.1
Computer/ Server	• Virtual Studio 2013 • Boost C++ Libraries v1.56.0 • OpenGL 4.2 • Windows 8.1

6.2 Deployment

Based on the multi-Kinect trilateration in Section 4.2 and the experiment result in Section 5.2, the deployment is designed as Fig.6.1. A coordinate system is built, and three Kinect v2 sensors are positioned at each vertex of a triangle. As one computer can only recognize one Kinect sensor, three Kinect sensors are connected to three computers. A router is applied to connect the computers and the server. Once the user starts walking, the joint positions captured by the three Kinect sensors are generated in a coordinate format by Kinect SDK and forwarded to the server through the router. Trilateration and virtualization procedures are done at the server using the data from the Kinect sensors.

6.3 Overall System Architecture

A human gait-tracking system's architecture is designed as in Fig.6.2. The system consists of several components: 1) A synchronization component guarantees that the clocks of Kinect clients and the servers are synchronized in milliseconds by network time protocol (NTP) [60]. 2) Based on the synchronization component, a time-scheduling component controls the joint data's transmission time sequence of the three Kinect clients such that, during one specific period, the server gets joint data one by one from each Kinect client. 3) At the Kinect client, the networking component serializes the joint data structure specified by Kinect SDK and forwards the data to the server; at the server, the network component deserializes the joint data from the three Kinect clients and sends the data to the trilateration component. 4) Trilateration component apply the trilateration method to

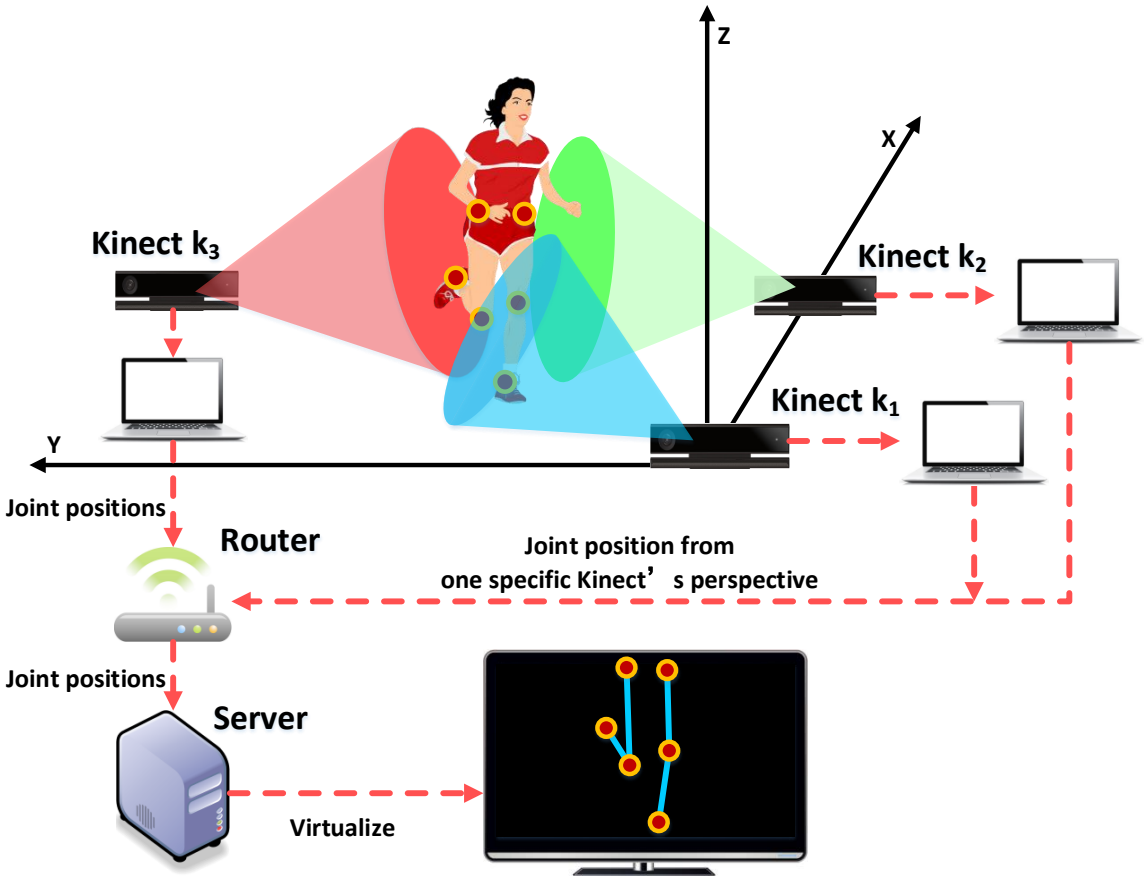


Fig. 6.1: The deployment of Multi-Kinect Human Gait Tracking System

compute the joint positions in 3D space. 5) Virtualization components apply OpenGL library to virtualize a 3D space and the human joints, and the positions are calculated by the trilateration component. The details of each component are given in Section 6.4.

6.4 Components Illustration

6.4.1 Synchronization Component

Many factors may affect the time when the server receives the joint data. These factors include the hardware differences (despite the same specifications), the network conditions, lighting conditions and so on. The speed of receiving joint data from the Kinect clients are different (i.e., during one specific period, joints data from one or two Kinect clients might be blocked on network), which can lead to an incorrect trilateration result. To solve

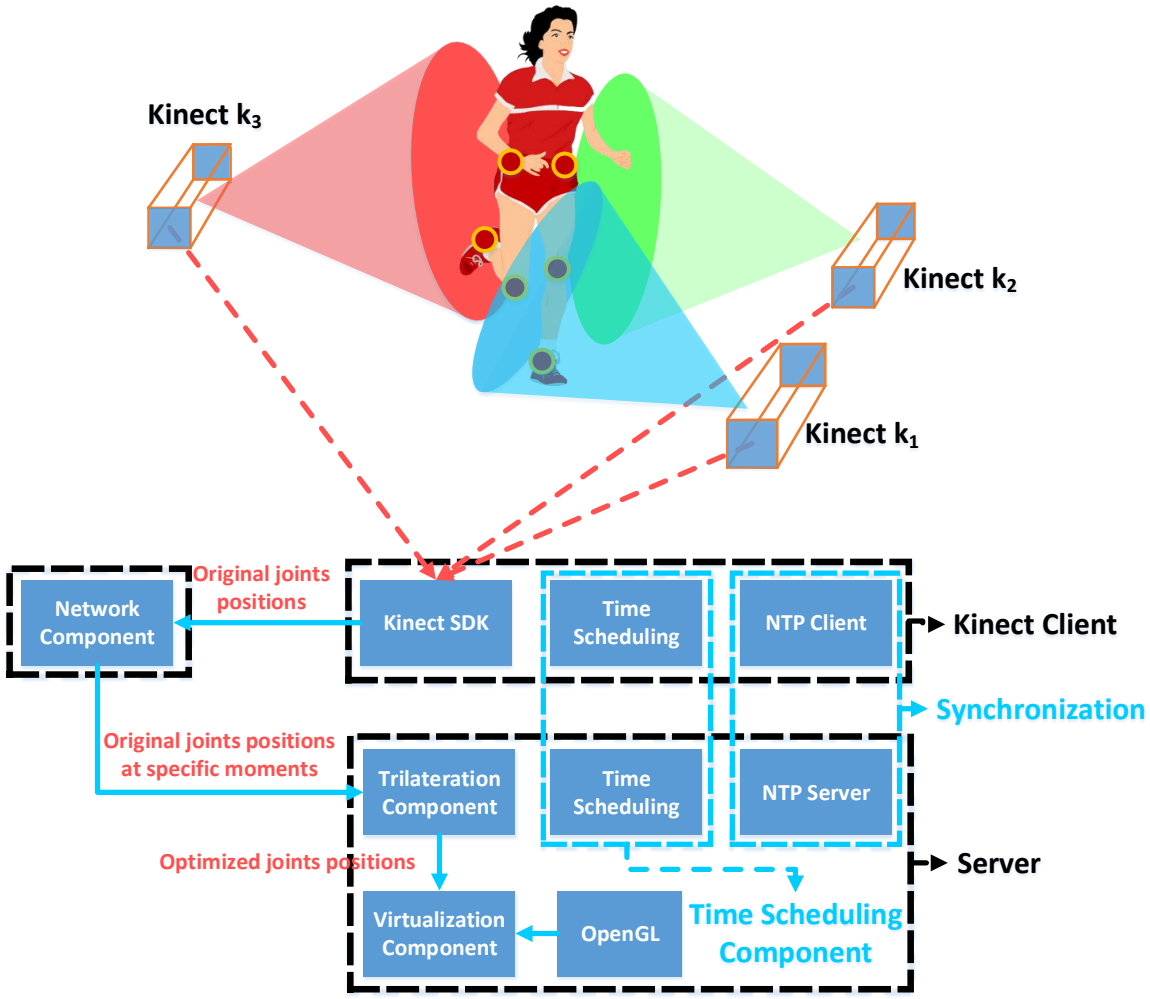


Fig. 6.2: Overall System Architecture of Multi-Kinect Human Gait System

this problem, a time-scheduling component is created to control the data-sending order of the Kinect clients. Further, to guarantee the time-scheduling component's correctness, the clock of each computer (the server and three Kinect clients) must be synchronized in milliseconds, since the general time cost from generating joint data by one Kinect sensor to receiving the joint data by the server is 3–5 ms (meseasured with Kinect SDK and a C++ Server).

Though the Windows 8.1 operating system has embedded a time-synchronization service based on NTP, the clock resolution of the NTP service is 1 second [61], which cannot satisfy this system's requirement of milliseconds of resolution with an accuracy of less than 1 ms. Thus, a synchronization component is created for getting a high resolution NTP service.

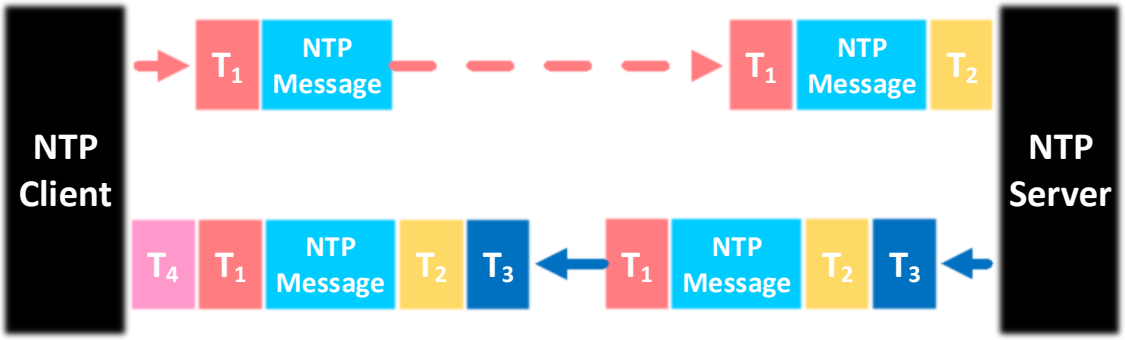


Fig. 6.3: The principle of NTP.

As shown in Fig.6.3, the NTP client sends a NTP message to the NTP server attached with a timestamp T_1 when the message leaves the NTP client. Secondly, the NTP server receives the NTP message and attaches a timestamp (T_2) when the message arrives at the NTP server. Thirdly, the NTP server sends back the NTP message to the NTP client attached with a timestamp T_3 when the message leaves the NTP server. Finally, the NTP client receives the message attached with a timestamp T_4 when the message arrives at the NTP client. With the four timestamps, we have

$$\begin{cases} T_2 = T_1 + t + d_1 \\ T_4 = T_3 - t + d_2 \\ d = d_1 + d_2 \end{cases} \tag{6.1}$$

where t is the time offset between the NTP client's clock and the NTP server's clock, d_1 is the transmission delay while the message is being sent from the NTP client to the NTP server, d_2 is the transmission delay while the message is being sent back to the NTP client from the NTP server, and d is the total transmission delay of the round trip. Assume $d_1 = d_2$, then we have

$$\begin{cases} t = \frac{(T_2 - T_1) - (T_4 - T_3)}{2} \\ d = (T_2 - T_1) + (T_4 - T_3) \end{cases} \tag{6.2}$$

According to Equation 6.1 and Equation 6.2, we have

$$t = (T_2 - T_1) + d_1 = (T_2 - T_1) + \frac{d}{2} \tag{6.3}$$

We can see that: 1) t or d is relevant to the differences of $T_2 - T_1$ and $T_4 - T_3$, which means t and d is not relevant to the process delay ($T_3 - T_2$) of the NTP server; 2) Equation 6.2 is obtained when $d_1 = d_2$ (d_1 or d_2 varies in the range of $0 \cdots d$). According to Equation 6.3, the maximum error of t is $\pm\frac{d}{2}$.

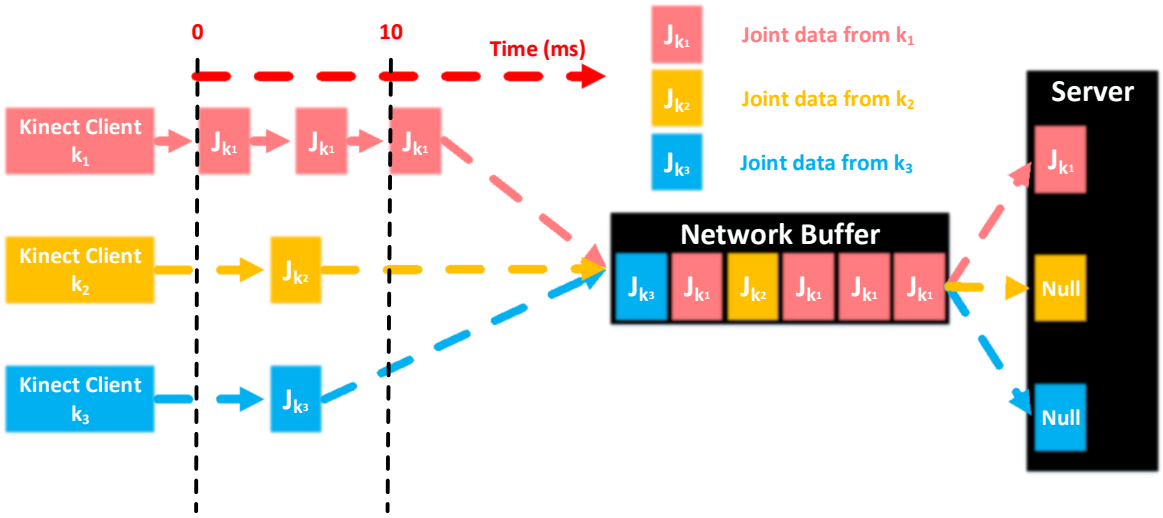


Fig. 6.4: Joint data from Kinect client k_1 and Kinect client k_2 is blocked on network.

Usually, in a local area network (LAN), the transmission delay (d) is less than 1ms (in this proposed system, the transmission delay is less than 1ms as tested by the ping service). Thus, the synchronization error of the synchronization component would theoretically be less than 0.5 ms (less than $\pm\frac{d}{2}$), which satisfy the requirement of the time scheduling component and is much better than the NTP service in Windows 8.1.

6.4.2 Time Scheduling

As the server receives joint data one by one, there is always a data receiving sequence for the three Kinect clients. Given conditions and hardware differences, the joint data cannot arrive in the right order (i.e., joint data from one or two Kinect clients might be blocked on the network while the previous one or two Kinect clients are sending more joint data to the server at the same time). As shown in Fig. 6.4, the speed of receiving joint data from Kinect client k_1 is much faster than those from Kinect client k_2 and Kinect client k_3. The joint data from k_2 and k_3 are blocked on network buffer and the server does not receive the joint data from k_2 and k_3 within a short time, which may lead to an incorrect trilateration result or a lack of enough data for the trilateration procedure.

Instead of continuing to send joint data to the server from Kinect clients, the time-scheduling component controls the Kinect clients' data-sending sequence by making Kinect clients obtain and send joint data one by one. Though the time cost from generating joint data by one specific Kinect client to receiving the data by the server is 3-5 ms, and the time cost of trilateration is less than 5 ms, this component cannot realize 5-ms resolution time

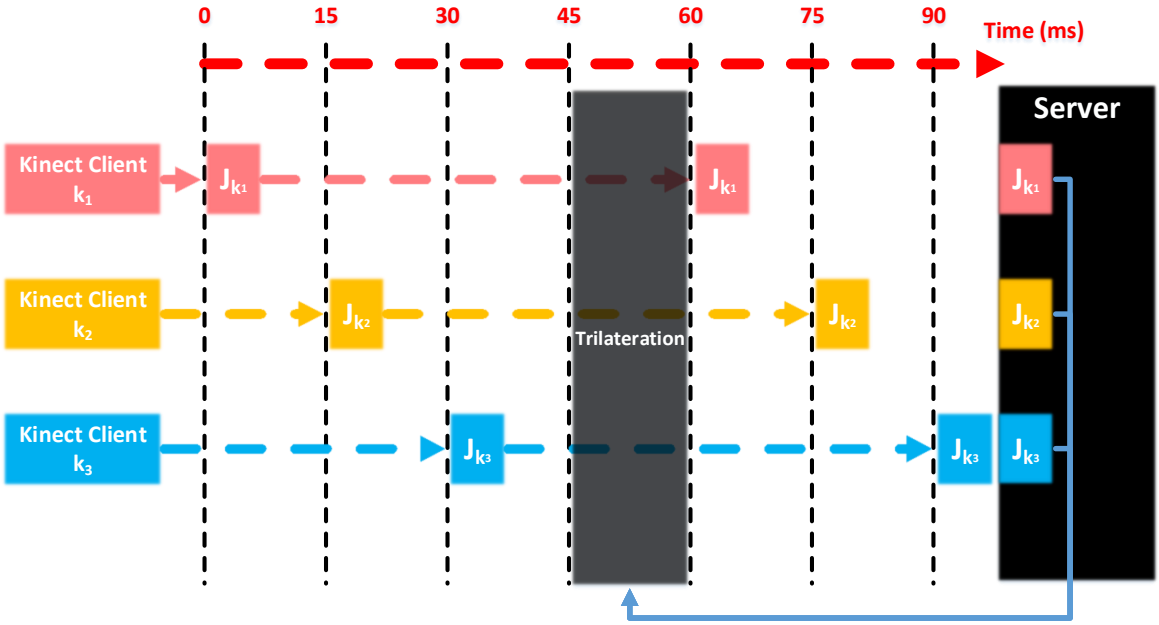


Fig. 6.5: Time schedule

scheduling, as the clock resolution given by Windows API is unstable (e.g., if Kinect client k_1 obtains joint data at time T_1 and sends it to the server, the time-scheduling component cannot guarantee that Kinect client k_2 would obtain the joint data at T_1+5 ms, as the time resolution is in the range of 1 to 15 ms). This is caused by the overclocking [62] feature of most modern CPUs. To guarantee the Kinect clients would obtain joint data one by one, this component schedule the periods (15-ms periods) for different Kinect clients to obtain and send joint data as shown in Fig.6.5: The time line is divided into 60-ms periods, and each 60-ms period is divided into four 15-ms periods. During each 60-ms period, the first 15-ms period is designed for Kinect client k_1 to obtain joint positions and send them to the server; the second and third 15-ms periods are designed for Kinect client k_2 and Kinect client k_3 to do the same procedures, respectively. The fourth 15-ms period is designed for the trilateration procedure, since three joint data sets are supposed to be received by the server during the last three 15-second periods. Thus, for each 60-ms period, the server would receive three joints position data sets from three Kinect sensors and do the trilateration during the last 15-ms period.

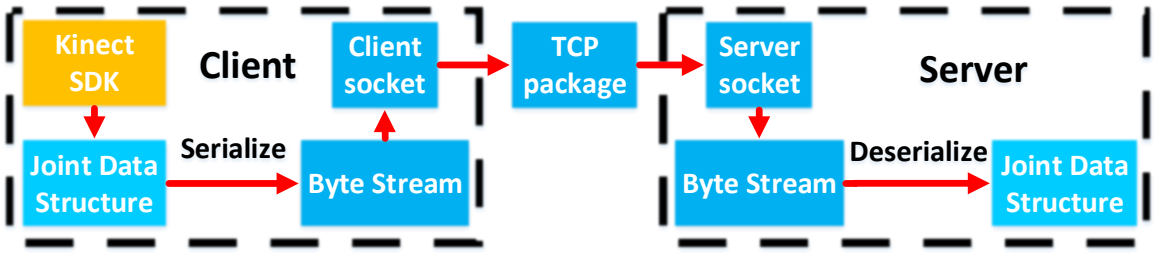


Fig. 6.6: Serialize procedure on the client and deserialize procedure on the server

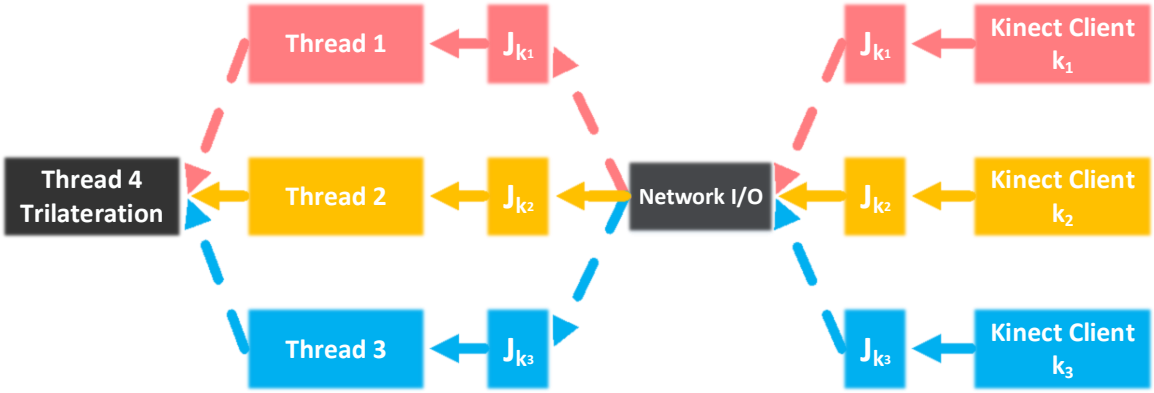


Fig. 6.7: Data transmission from Kinect clients to the multi-thread asynchronous server.

6.4.3 Network Component

As Fig.6.1 shows, there are three Kinect clients obtaining and transferring joint data. The network component keeps the connections between each client and the server. As the joint data structure is specified by Kinect SDK, and byte stream is the only data structure allowed in network data transmission, the joint data need to be serialized at the Kinect client such that the information can be reconstructed (deserialized) precisely with real-time values on the server. The Boost.Serialization library [63] is applied to realize serialization functionality, as shown in Fig. 6.6.

A multi-thread asynchronous server handles the joint data sets from the different Kinect clients. Each thread handles the joint data from one specific Kinect client. Thus, three threads are created for three Kinect clients, and the fourth thread is created to perform the trilateration procedure. The asynchronous feature of the server guarantees that the I/O is not waiting for joint data from one specific Kinect client (i.e., before receiving joint data from one specific Kinect client, I/O could be used for receiving joint data from another

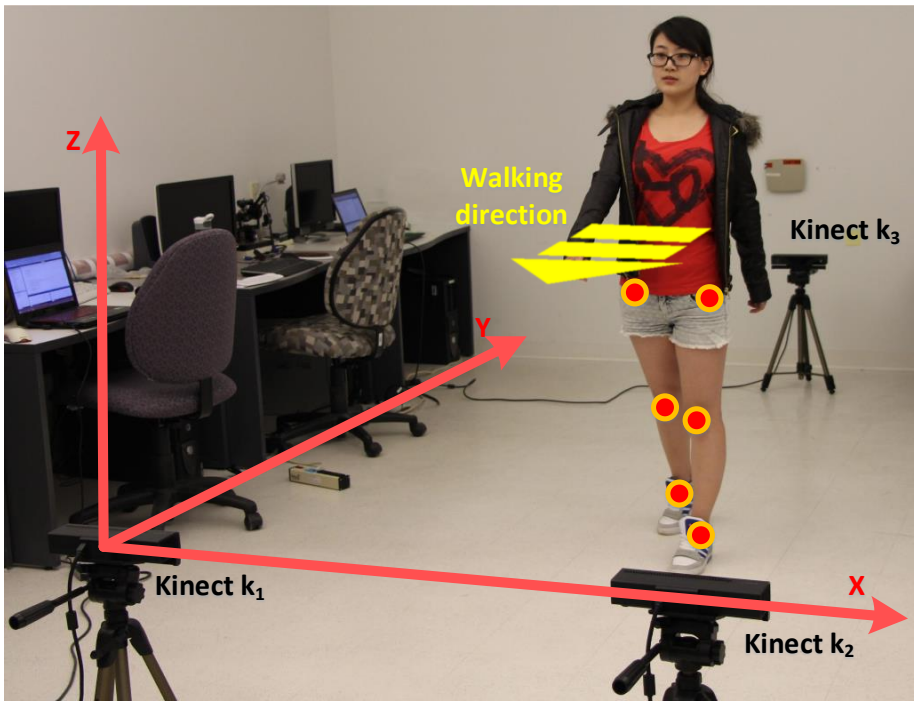
Kinect client by the other two threads). The Boost.Asio library [63] is applied to realize the multi-thread asynchronous server, as shown in Fig.6.7.

6.4.4 Trilateration Component

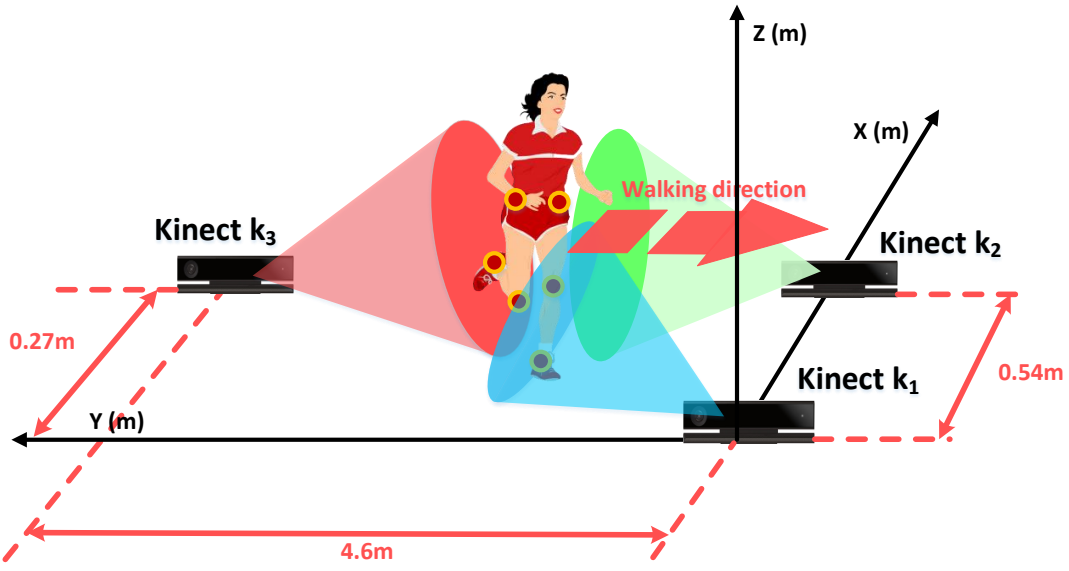
This system apply multi-Kinect trilateration principle mentioned in Section 4.1 to locate the joint position in space. The user is required to walk for several steps in a specific direction inside the triangle with three Kinect sensors as the vertexes. Totally six joints including right hip, left hip, right knee, left knee, right ankle and left ankle are tracked while the user is walking. The setup is shown in Fig. 6.8.

According to the published specifications, the Kinect v2 sensor cannot observe anything when the target is positioned less than 0.5 meters away from the sensor. Hence, the entire walking range is designed as a 4.6-meter height triangle and the recommend walking area is a 3-meter straight line at the central part. To be appropriate for other environments, the configuration (the three Kinect sensors' relative positions) needs to be customized manually, and the trilateration algorithm needs to be modified accordingly. The trilateration algorithm would be executed once three different joint data structures are received by the server in the right order, i.e., the trilateration thread would be executed after the other three threads get three joint data structures from the three Kinect sensors, respectively. The total time cost for getting one optimized joint position depends on the environmental condition. Here, under the system requirement mentioned in Section 6.1, the time cost for one optimized joint position would be 60 ms. The system would calculate 1,000 optimized positions for one specific joint (6,000 optimized joint positions for six joints for human gait tracking), so the total duration that the user needs to walk for is 60 seconds. To walk within a 3-meter range for 60 seconds, the user might need to walk 3 meters and walk back to the origin several times. The duration can also be customized manually if more joint data is required. As the Kinect sensor cannot track human body when the distance is larger than 10 meters, the recommended configuration for trilateration is that the largest distance between two Kinect sensors should be less than 10 meters.

Based on the multi-Kinect trilateration method mentioned in Chapter 4, the overall accuracy can be improved when more Kinect sensors are applied. Thus, to apply more than three Kinect sensors, similarly, the trilateration component needs to be modified (including modifying the parameter list, the matrix's size, etc.), several more threads need to be created for receiving the joint data from the additional Kinect sensors, and certainly the time schedule needs to be modified to be suitable for a larger joint data sending schedule. To give a general example, suppose there are n Kinect sensors obtaining and sending joint data; n threads need to be created to handle the joint data from each Kinect



(a)



(b)

Fig. 6.8: Triangulation configuration. (a) Trilateration configuration in reality. (b) The virtualized trilateration configuration map corresponding to (a), which shows more details about how the Kinect sensors are positioned.

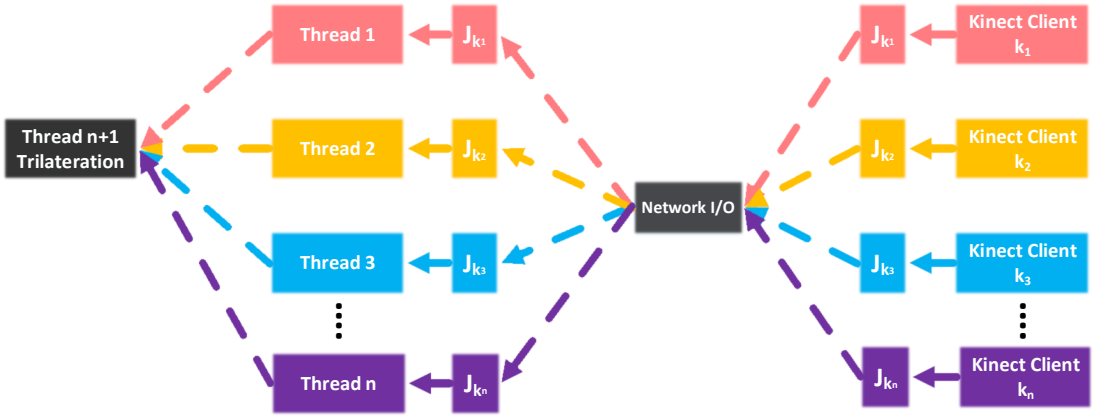


Fig. 6.9: N-Kinect multi-thread server

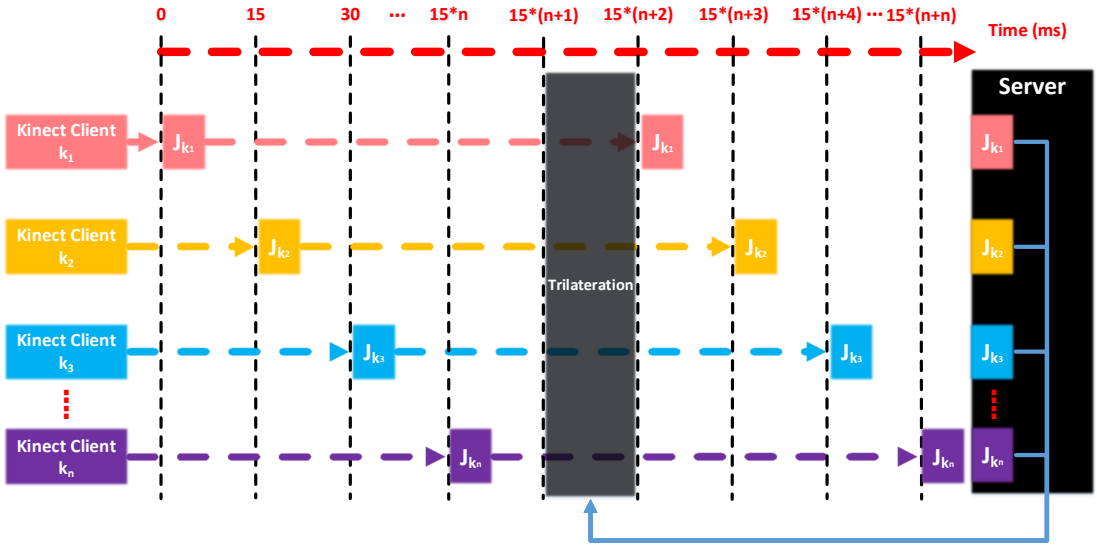


Fig. 6.10: N-Kinect time schedule

sensor, and the $(n+1)$-th thread is designed for executing the trilateration procedure (Fig. 6.9). Similarly, more specific periods need to be scheduled for additional Kinect sensors. In the configuration mentioned in Section 6.1, n 15-milliseconds periods are scheduled for n Kinect sensors to obtain and send joint data while the $(n+1)$-th 15-ms period is designed for executing trilateration procedure (Fig. 6.10).

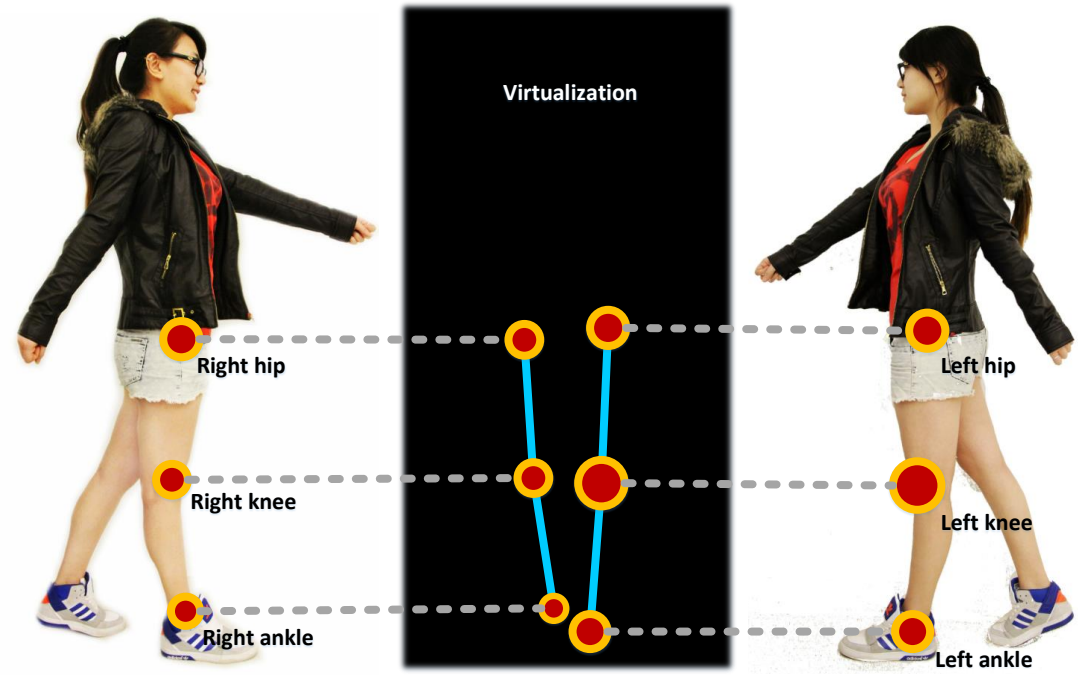


Fig. 6.11: Six joints are virtualized including left hip, left knee, left ankle, right hip, right knee and right ankle.

6.4.5 Virtualization Component

As described in Section 6.4.2, the total time cost for generating six-joint trilateration result is 60 ms. Actually, the time cost is larger when the cost of system operations is added (such as the switches between threads). If the virtualization is added, the total time cost for one frame (including 60 ms for trilateration, system operation and virtualization of the joints, texture, and bones in a 3D environment with OpenGL) would be around 100 ms. That is, the number of frames able to be generated for one second is around 10, which means the user can hardly see anything in real time. More powerful or professional hardware (with a stable high-resolution clock system, fast CPUs, and a powerful graphic adapter) would help a lot for creating a virtualization component in real time. As a limit to the time cost, in this thesis, the virtualization component is conceived as a non-real-time component. For each 60 ms, the six-joint trilateration results would be stored in a circular buffer with a sizable store of 6,000 joint positions (1,000 for each joint). Once the buffer is full, the joint positions (3D coordinates in float format) are exported and stored in a text file. Then the virtualization component reads the file and virtualizes the joints in a 3D coordinate system with OpenGL library. A total of six joints, including the left hip, left knee, left ankle, right hip, right knee, and right ankle are virtualized (Fig. 6.11).

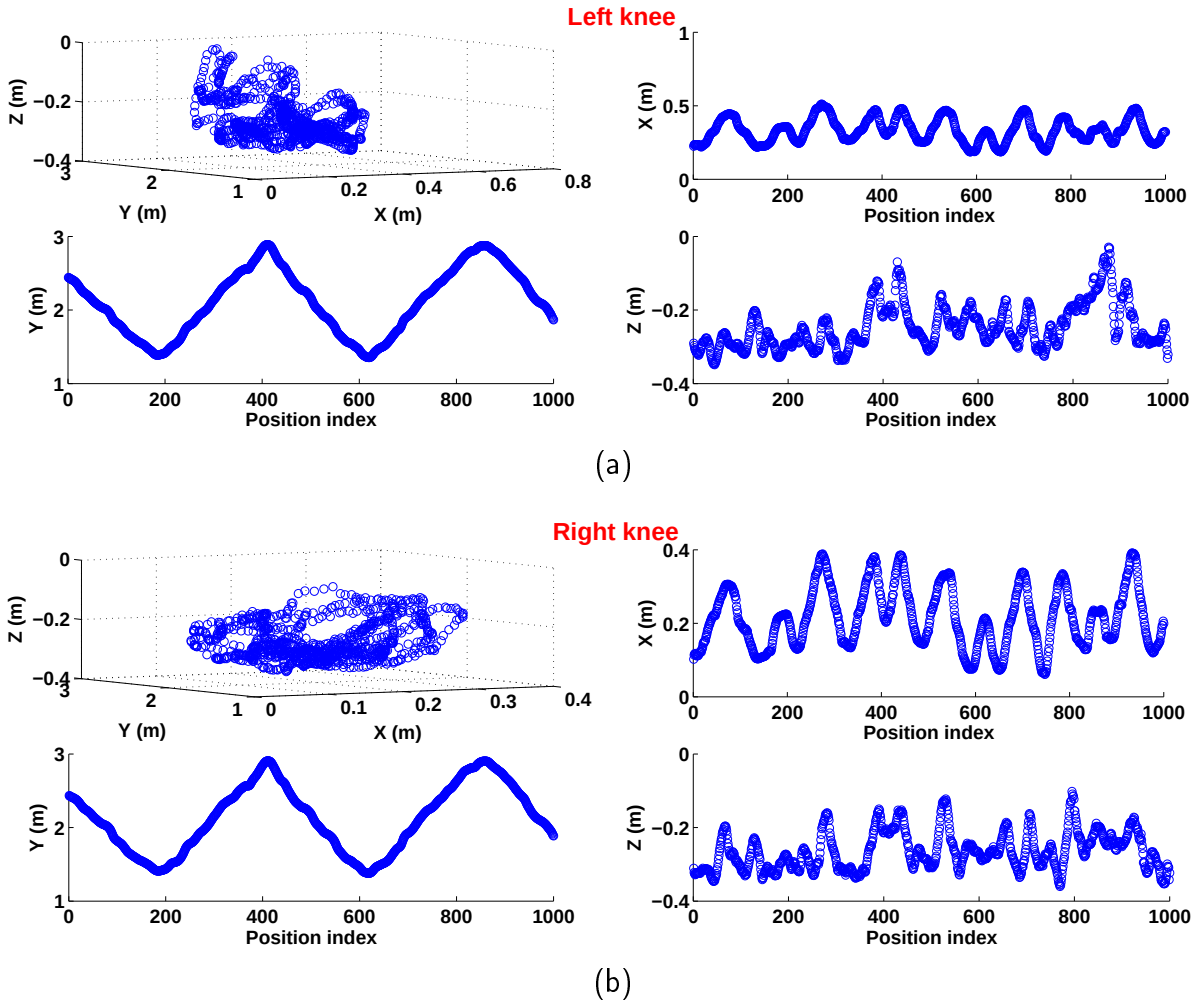


Fig. 6.12: The left knee and the right knee's trilateration result. (a) Trilateration result of left knee. The top-left sub-figure shows how left joint is moving in a coordinate system with 1000 recorded position variances while the user is walking. The last three sub-figure shows how the joint is moving in a specific direction corresponding to three axises (X, Y and Z axis) of the top-left sub-figure. (b) Trilateration result of right knee. Similarly, the top-left sub-figure shows how right joint is moving in a coordinate system with 1000 recorded position variances and the last three sub-figures show the joint's movement in specific directions.

6.5 System Result

In this human gait tracking system, to track joint with time, totally 1000 positions are recorded during 30 seconds (i.e., 6000 positions for six joints are recorded during 30 seconds while the user is walking within the measurement area). In our set-up (mentioned in Section 6.4.4), without loss of generality, the user is required to walk three steps forward,

four steps backward, four steps forward, five steps backward, and three steps forward. The trilateration result of the six joint movements is then recorded in a Text file, as described in Section 6.4.5.For giving more details about the human gait movement, the text file is loaded by Matlab and analyzed.

As shown in the figure Fig. 6.12, 1,000 position variances for each knee are recorded for 30 seconds and then virtualized by Matlab (Specifically, Fig. 6.12a shows left knee's movements and Fig. 6.12b shows right knee's movements). In this Figure, the X, Y, Z axes correspond to the three directions of the trilateration configuration mentioned in Fig. 6.8. As the user walks forward and backward several times, the movements in the 3D coordinate system (the upper left subfigure of Fig. 6.12a and Fig. 6.12b) might not be clear, while the last three subfigures show the details about how the joint is moving in specific directions. For example, in Fig. 6.12a, the bottom-left subfigure shows how the user's left knee is moving in the Y direction, which indicates that the user's left knee moves in the Y-axis's opposite direction for about 1.5 m during the first 200 position variances, 2.5 m in the Y-axis direction during the following 200 position variances, and similarly, the knee move forwards or backwards several times. These data from evaluating these joint movements can be applied for entertainment or medical purposes.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

Compared with Kinect v1, the newly released Kinect for Windows sensor v2 has improved the performance of the hardware, according to the published specifications on the official website. In this paper, we investigated several properties that are important for the practical usage of Kinect v2, including accuracy distribution, depth resolution, depth entropy, edge noise, and structural noise. Based on the results we obtained from the experiments, Kinect v2 has good accuracy if the object is positioned within the green regions (Fig. 5.2). Furthermore, we proposed a muti-Kinect trilateration approach to improve the accuracy when three Kinect sensors are used. The experiment shows that the multi-Kinect trilateration method works well even when the sensors are positioned more than four meters away from the target object.

There are also some other parameters affecting the Kinect v2's performance. For example, an object with reflective material (like mirror) can lead to a problem that the IR (Infrared) light emitted by Kinect sensor cannot be reflected back to the camera, making the depth values unreliable or unable to be determined. Similarly, an object covered with light-absorbing materials (like carbon black) can cause less IR light reflected back to the camera. Furthermore, if two Kinect v2 sensors are positioned towards each other, the area around the camera would disappear in the depth images captured by both two sensors. This phenomenon happens probably because the camera interference IR light from the other sensor. The unattractive power bricks, complex cables and the high requirement for the laptop also limit the use of the sensor.

Based on the experimental results, a human gait-movement tracking system was realized. With three Kinect for Windows v2 sensors positioned accordingly, six joints on the

user's legs were located with the trilateration method. A limit to the hardware performance is that the virtualization of the user's walking sense cannot be realized in real time. Instead, the joints' movement path was recorded as 6,000 continuous position variances for 30 seconds in text file and then loaded by a virtualization component for virtualization. Further, this system can be extended with a higher accuracy if more Kinect sensors are applied.

In summary, as a depth sensor with a relatively low price (much lower than professional depth cameras or tracking systems such as Vicon), Kinect for Windows sensor v2 shows acceptable performance and has a great potential to be applied to many fields, such as entertainment, education, and medicine. Based on the features of the Kinect for Windows v2 sensor, the human gait-tracking system realized in this thesis has a low cost and high accuracy and is an easy-to-use solution for gait tracking.

7.2 Future Work

In this thesis, a human gait-tracking system is developed with three Kinect sensors based on the trilateration location principle. Considering the realization of the gait-tracking system, there are several directions to work on to attain higher accuracy, easier setup, or real-time virtualization.

- ***Powerful hardware or a specific software environment.*** As mentioned in Chapter 6, the virtualization cannot be realized in real time because the clock resolution obtained from Windows API is limited, though the hardware's clock resolution is much higher. If better computers or a specific software environment providing higher clock resolution is applied, virtualization can be realized in real time.

- ***Different three-Kinect configuration.*** As described in Chapter 6, the configuration requires that three Kinect sensors are located at fixed positions for generating a triangle measurement area, which means that manual measurement is required (e.g., to guarantee three Kinect sensors are in the same horizontal plane, manual measurements need to be done). The sensors positioned at different horizontal plane may lead to the target is out of some sensors' viewing range while the other sensors is able to observe it. The same horiozontal configuration also gives more simple spheres' equations which reduce the calculations in trialateration procedure. Thus, one of the directions for future work would be to find a trilateration method that could ignore the height differences among the Kinect sensors' positions (i.e., try position the sensors at different heights and get a maximum measurement area). Also, as there are

many kinds of triangles for positioning three Kinect sensors as the vertexes, finding a three-Kinect setup that provides the best accuracy is also an interesting direction for future work.

- ***Expanding the system by applying more than three Kinect sensors.*** The trilateration location algorithm with multi-Kinect sensors is described in Chapter 4, and a least square method is proposed for minimizing measurement errors when more than three Kinect sensors are applied. Hence, to improve the accuracy, applying more Kinect sensors with well-designed configurations is an important direction for future work.

References

[1] M. Fleischer. Method of producing moving-picture cartoons, 1917.

[2] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception & Psychophysics*, 14(2):201–211, 1973.

[3] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[4] C. M. Ginsberg and D. Maxwell. Graphical marionette. In *Proc. Of the ACM SIGGRAPH/SIGART Interdisciplinary Workshop on Motion: Representation and Perception*, pages 303–310, New York, NY, USA, 1986. Elsevier North-Holland, Inc.

[5] B. Robertson. Mike, the talking head. *Computer graphics world*, 11(7):57, 1988.

[6] J.C. Sabel. Optical 3D motion measurement. In *Instrumentation and Measurement Technology Conference, 1996. IMTC-96. Conference Proceedings. Quality Measurements: The Indispensable Bridge between Theory and Reality, IEEE*, volume 1, pages 367–370, 1996.

[7] J.-C. Cheng and J.M.F. Moura. Capture and representation of human walking in live video sequences. *Multimedia, IEEE Transactions on*, 1(2):144–156, Jun 1999.

[8] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 8–15, Jun 1998.

[9] B. M. Thomas and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231 – 268, 2001.

[10] X. Tianwei, Y. Yao, Z. Yu, and D. Sidan. Markerless motion capture of human body using PSO with single depth camera. In *2012 Second International Conference on 3D*

Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), pages 192–197, Oct 2012.

[11] B. M. Thomas, H. Adrian, and K. Volker. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104:90 – 126, 2006.

[12] D. Reber. Nick strives to define motion capture. *Animation World Magzine*, 3:11, 1999.

[13] Sound & motion University of Oslo. http://www.qualisys.com/applications/customer-cases/sound-and-motion-at-fourms-lab/, Accessed April 1, 2015.

[14] Vicon. http://www.vicon.com/, Accessed April 2, 2015.

[15] Cyberware laser scanner. http://cyberware.com/products/scanners/px.html, Accessed August 28, 2014.

[16] S. Foix, G. Alenya, and C. Torras. Lock-in time-of-flight (ToF) cameras: A survey. *IEEE Sensors Journal*, 11(9):1917–1926, 2011.

[17] Primesense carmine sensors. http://www.primesense.com/wp-content/uploads/2-012/12/PrimeSenses_3DsensorsWeb, Accessed August 24, 2014.

[18] Kinect. http://www.microsoft.com/en-us/kinectforwindows/, Accessed August 28, 2014.

[19] J. Salvi, S. Fernandez, T. Pribanic, and X. Llado. A state of the art in structured light patterns for surface profilometry. *Pattern Recognition*, 43(8):2666–2680, 2010.

[20] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1297–1304, 2011.

[21] A. Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight sensors in computer graphics. In *Proceedings of Eurographics (State-of-the-Art Report)*, pages 119–134, 2009.

[22] P. Nobles, S. Ali, and H. Chivers. Improved estimation of trilateration distances for indoor wireless intrusion detection. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 2(1):93–102, 2011.

[23] Qualisys. http://www.qualisys.com/, Accessed April 2, 2015.

[24] Northern digital inc. http://www.ndigital.com/, Accessed April 2, 2015.

[25] S.L. Dockstader and A.M. Tekalp. Multiple camera tracking of interacting and occluded human motion. *Proceedings of the IEEE*, 89(10):1441–1455, 2001.

[26] T. Cloete and C. Scheffer. Benchmarking of a full-body inertial motion capture system for clinical gait analysis. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 4579–4582, Aug 2008.

[27] Xsens. https://www.xsens.com/, Accessed April 7, 2015.

[28] E. H. Hall. On a new action of the magnet on electric currents. *American Journal of Mathematics*, 2(3):pp. 287–292, 1879.

[29] Polhemus. http://polhemus.com/, Accessed April 7, 2015.

[30] A. Gmiterko and T. Lipták. Motion capture of human for interaction with service robot. *American Journal of Mechanical Engineering*, 1(7):212–216, 2013.

[31] Logitech. http://www.logitech.com/, Accessed April 7, 205.

[32] J. Vince. *Essential Computer Animation fast: How to Understand the Techniques and Potential of Computer Animation*. Springer-Verlag London, 2000.

[33] Leap motion. https://www.leapmotion.com/, Accessed April 7, 2015.

[34] L. Chen, H. Wei, and J. Ferryman. A survey of human motion analysis using depth imagery. *Pattern Recognition Letters*, 34(15):1995 – 2006, 2013. Smart Approaches for Human Action Recognition.

[35] J. Geng. Structured-light 3D surface imaging: a tutorial. *Advances in Optics and Photonics*, 3(2):128–160, Jun 2011.

[36] A. Shpunt and B. Pesach. Optical pattern projection, 2008.

[37] S. Foix, G. Alenya, and C. Torras. Lock-in time-of-flight (tof) cameras: A survey. *Sensors Journal, IEEE*, 11(9):1917–1926, Sept 2011.

[38] M. Hansard, S. Lee, O. Choi, and R.P. Horaud. *Time-of-Flight Cameras: Principles, Methods and Applications*. SpringerBriefs in Computer Science. Springer, 2013.

[39] K. Žbontar, M. Mihelj, B. Podobnik, F. Povše, and M. Munih. Dynamic symmetrical pattern projection based laser triangulation sensor for precise surface position measurement of various material types. *Applied Optics*, 52(12):2750–2760, 2013.

[40] Cyberwar. http://cyberware.com/, Accessed April 12, 2015.

[41] B. Allen, B. Curless, and Z. Popović. The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Transactions on Graphics*, 22(3):587–594, 2003.

[42] L. De Chiffre, S. Carmignato, J.-P. Kruth, R. Schmitt, and A. Weckenmann. Industrial applications of computed tomography. *CIRP Annals-Manufacturing Technology*, 63(2):655 – 677, 2014.

[43] K. Khoshelham and S. O. Elberink. Accuracy and resolution of Kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012.

[44] K. Khoshelham. Accuracy analysis of Kinect depth data. In *Proceedings of International Society for Photogrammetry and Remote Sensing Workshop on Laser Scanning*, pages 133–138, 2011.

[45] J. Wheat, R. Fleming, M. Burton, J. Penders, S. Choppin, and B. Heller. Establishing the accuracy and feasibility of Microsoft Kinect in various multi-disciplinary contexts. Technical report, Sheffield Hallam University, U.K., 2011.

[46] B. Galna, G. Barry, D. Jackson, D. Mhiripiri, P. Olivier, and L. Rochester. Accuracy of the Microsoft Kinect sensor for measuring movement in people with Parkinson's disease. *Gait and Posture*, 39(4):1062–1068, 2014.

[47] D. C. Herrera, J. Kannala, and J. Heikkilä. Joint depth and color camera calibration with distortion correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):2058–2064, 2012.

[48] C. Raposo, J. P. Barreto, and U. Nunes. Fast and accurate calibration of a Kinect sensor. In *Proceedings of International Conference on 3D Vision*, pages 342–349, 2013.

[49] J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan. Scanning 3D full human bodies using Kinects. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):643–650, 2012.

[50] A. Awad, T. Frunzke, and F. Dressler. Adaptive distance estimation and localization in WSN using RSSI measures. In *The 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools*, pages 471–478, 2007.

[51] F. Thomas and L. Ros. Revisiting trilateration for robot localization. *IEEE Transactions on Robotics*, 21:93–101, 2005.

[52] R. Bajaj, S. L. Ranaweera, and D. Agrawal. GPS: Location-tracking technology. *Computer*, 35:92–94, 2002.

[53] D. E. Manolakis. Efficient solution and performance analysis of 3-D position estimation by trilateration. *IEEE Transactions on Aerospace and Electronic Systems*, 32:1239–1248, 1996.

[54] T. Takatsuji, M. Goto, A. Kirita, T. Kurosawa, and Y. Tanimura. The relationship between the measurement error and the arrangement of laser trackers in laser trilateration. *Measurement Science and Technology*, 11(5):477, 2000.

[55] W. Hereman and S. W. Murphy. Determination of a position in three dimensions using trilateration and approximate distances. Technical report, Colorado School of Mines, USA, 1995.

[56] Q. V. Le and A. Y. Ng. Joint calibration of multiple sensors. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3651–3658, 2009.

[57] L. Beyer and J. Wulfsberg. Practical robot calibration with ROSY. *Robotica*, 22:505–512, 2004.

[58] D. Um, D. Ryu, and M. Kal. Multiple intensity differentiation for 3-D surface reconstruction with mono-vision infrared proximity array sensor. *IEEE Sensors Journal*, 11(12):3352–3358, 2011.

[59] R. A. Horn and C. R. Joohnson. *Matrix Analysis*. Cambridge University Press, 1990.

[60] L. D. Mills. Network time protocol (version 3) specification. Implementation and Analysis-RFC1305, 1992.

[61] B. Dawson. Windows timer resolution: Megawatts wasted. https://randomascii.wordpress.com/2013/07/08/windows-timer-resolution-megawatts-wasted/, Accessed March 9, 2015.

[62] S. Wainner and R. Richmond. *The Book of Overclocking: Tweak Your PC to Unleash Its Power*. No Starch Press, San Francisco, CA, USA, 2003.

[63] Boost C++ Library. http://www.boost.org/, Accessed May 26,2015.