HOLOSTREAM: HIGH-ACCURACY, HIGH-SPEED 3D RANGE VIDEO ENCODING AND STREAMING

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Tyler Bell

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

June 2018

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Prof. Song Zhang, Co-chair

School of Mechanical Engineering

Prof. Jan P. Allebach, Co-chair

School of Electrical and Computer Engineering

Prof. Fengqing M. Zhu

School of Electrical and Computer Engineering

Prof. Karthik Ramani

School of Mechanical Engineering

Approved by:

Prof. Venkataramanan Balakrishnan

The Michael and Katherine Birck Head of Electrical and Computer Engineering

Rejoice in the Lord always. I will say it again: Rejoice! Let your gentleness be evident to all. The Lord is near. Do not be anxious about anything, but in every situation, by prayer and petition, with thanksgiving, present your requests to God. And the peace of God, which transcends all understanding, will guard your hearts and your minds in Christ Jesus.

Philippians 4:4-7

ACKNOWLEDGMENTS

I would like to thank every family member, teacher, and friend who has helped guide and mentor me throughout my life and academic career. The following words will never be able to appropriately convey my most sincere gratitude.

First, I would like to express how grateful I am to my co-advisor, Prof. Song Zhang, with whom I have had the privilege of working with for five and a half years. Not only did joining his group provide the opportunity to conduct innovative and exciting research, but it also made me a better person. I will always consider myself lucky to have been able to learn under Prof. Zhang. Through his guidance as a research advisor, educator, and friend, I was able to push my limits, expand my horizons, and reach new potentials, both in the lab and in my life. As I begin a faculty position of my own upon graduation, I hope to model many of the traits that have helped Prof. Zhang become a successful mentor, educator, and researcher.

Next, I would like to express my gratitude to my other co-advisor, Prof. Jan P. Allebach. I am very grateful to have had the opportunity to work with and learn from him during my time at Purdue. Prof. Allebach's encouragement, helpfulness, and advice have impacted my academic career every step of the way. Prof. Allebach's character and the general manner in which he conducts himself have also influenced me greatly. For example, his ability to treat everyone fairly and with an attitude of helpfulness, while still managing a large variety of responsibilities, has served as a great example for how I should conduct my own relationships.

Moreover, I would like to thank my other Ph.D. advisory committee members, Prof. Karthik Ramani and Prof. Fengqing Maggie Zhu, for their tremendous help, guidance, and overall support. They have always been kindly willing to offer help and provide new insights into my work throughout my time at Purdue. I am so thankful

for their generosity, both in time and support, in helping me learn and grow during these early stages of my career.

Next, I would like to thank all of the outstanding past and present team members that I have had the privilege of working with, including: Yatong An, Michael Crawford, Michael Feller, Jae-Sang Hyun, Chufan Jiang, Ziping Liu, Bogdan Vlahov, and Peter Zibley. Working alongside each other has been an amazing experience. I will forever be thankful for our group's ability to feel like a family, always there to help and encourage one another when needed. In particular, I'd like to thank my colleague Prof. Beiwen Li, with whom I worked with for more than four years, for his friendship, encouragement, advice, and for his modeling of what a successful young academic looks like.

Lastly, none of this dissertation work would have been possible without the support of my family. To my wife, Melanie, thank you for your continued love, patience, and encouragement. Packing up and moving to Purdue was an adventure, and I'm so grateful for all that we have been able to experience during our time here, including the beginning of our marriage and welcoming our first child. You are an amazing wife, mother, and teammate, and I am beyond fortunate to be able to share my life with you. To my daughter, Natalie, thank you for your smile, your joy, and your resilience. You are only one year old, but in that year you have already provided us with so many new cherished memories and perspectives on life. To my friends, family, sisters, and grandparents, thank you for your continued love and positive words. To my mom and dad, thank you for an outstanding upbringing in which I was encouraged, supported, and most of all, loved wholly. I love you all.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

ABSTRACT

Bell, Tyler Ph.D., Purdue University, June 2018. Holostream: High-Accuracy, High-Speed 3D Range Video Encoding and Streaming. Major Professors: Song Zhang (School of Mechanical Engineering) and Jan P. Allebach (School of Electrical and Computer Engineering).

In recent decades, three-dimensional (3D) scanning techniques have improved in terms of speed, quality, and usability. These improvements have helped 3D scanning applications become increasingly popular within many different fields and industries, such as entertainment, security, manufacturing, and human computer interaction. However useful, real-time 3D scanning techniques generate large amounts of data, which may limit real-time storage or transmission of acquired 3D data. A platform which can enable high-quality, low-bandwidth 3D video communications could then be very useful within a broad range of applications, for example, telemedicine.

The goal of *telemedicine* is to use telecommunications technology to remotely diagnose, monitor, and treat patients, offering: reduced health care costs, increased access to distant specialists, reduced health care inequity, improved patient comfort, and assistance in the early detection of adverse symptoms. Each evolution of communications technology has offered new advantages to the remote physicians to aid in their work. For instance, modern two-dimensional (2D) video communications can be used to allow remote physicians to both converse with and visually assess their patient's affliction, even if they are from a rural or underprivileged area. Although 2D photographs and videos work well for visual assessments, they lack depth and perspective which may be required to form an accurate diagnosis or to monitor a patient's affliction over time. Three-dimensional (3D) scanning systems, however, offer the potential to capture this information with great accuracy. Thus, if telemedicine technologies adopted 3D scanning systems, accurate measurements of afflicted areas

could greatly influence the remote decision making of the physician. The goal of this dissertation research is to realize such a 3D communications platform that can transmit high-accuracy, high-resolution 3D data from one user to another over existing wireless network infrastructures.

The first challenge in realizing an effective 3D communications platform is obtaining the high-quality 3D data to be delivered. In the context of telemedicine, for example, the accuracy, resolution, and field-of-view (FOV) of the 3D data could have significant impact on a physician's ability to perform analysis and make sound decisions. Structured light 3D scanners have the potential to provide both high-resolution scans with great accuracy; however, due in part to their calibration procedures, they are most effective at close distances (i.e., small FOV). In order to extend the measurement range of a structured light system, we proposed a novel camera calibration procedure which can more feasibly calibrate long-range vision systems. The procedure was able to accurately calibrate a camera (in-focus at a long range) at a near range (where the camera may be substantially out-of-focus) with only 0.2% difference in focal length versus a traditional, in-focus calibration. Our calibration procedure was then used to aid in the calibration of a long-range structured light 3D scanning system, extending its effective FOV.

The second challenge in realizing an effective 3D communications platform is the transmission of 3D video: the information required to represent high-accuracy, high-resolution 3D data is quite large which makes it difficult to transmit quickly over standard wireless networks. Given this, we have proposed two techniques for encoding 3D data within a regular 2D image that are very resilient to lossy image compression; therefore, the encoded 2D image could then be substantially compressed using JPEG techniques. Our 3D data encoding procedures offered both very high compression ratios (i.e., small file sizes) and reconstruction accuracies (i.e., low error rates). For instance, one achieved a compression ratio of 935:1 when using JPEG 80, versus the OBJ file format, with a reconstruction error rate of 0.027%.

With these accomplishments, transmitting high-quality 3D video data wirelessly to remote devices became possible. This dissertation research then developed the novel *Holostream* platform for high-quality 3D video recording, encoding, compression, decompression, visualization, and interaction. A demonstration system successfully delivered video-rate photorealistic 3D video content over standard wireless networks to mobile (e.g., iPhone, iPad) devices. Taking advantage of this dissertation research's two proposed 3D geometry encoding methods and mature video compression techniques, Holostream was able to deliver high-quality 3D video and color texture data to mobile devices using only 4.8-14 Mbps. Even under extremely poor network conditions, structurally representative 3D geometry and reasonably high-quality texture information could be delivered using only 0.48 Mbps. Finally, to emphasize the efficiency of this platform, this dissertation research also developed a fully mobile Holostream platform implementation that enabled mobile iPad devices to both send and receive 3D video content, acquired via a commercially available structured light scanner, in real-time across wireless networks.

In summary, this dissertation research has (1) contributed methods for extending the effective FOV for high-accuracy, high-resolution 3D data capture; (2) contributed methods for the efficient and accurate encoding of high-resolution 3D range geometry data; and (3) has developed a novel and modular platform for high-quality, low-bandwidth 3D video communications. These contributions now enable individuals to transmit 3D video in real-time, using a 3D sensor of their choosing, over existing wireless networks, without losing significant quality due to the compressed transmissions. Such capabilities could not only transform how we communicate every day, but they could also enable many new applications, including those within telemedicine, the digital arts, homeland security, and remote surgery.

1. INTRODUCTION

As telecommunication technologies evolve, they become increasingly used within research, industry, and by the general public. For example, two-dimensional (2D) video calls (e.g., Skype, FaceTime) are widely used in today's society as a basic form of communication. Such technologies, enabled by mature video compression techniques, offer accessible, intuitive communications with reasonable visual fidelity. Recently, 3D video scanning has emerged as a useful technology within areas such as entertainment, security, and manufacturing; however, it remains difficult to transmit 3D video data in real-time across standard networks due to the inherent size of 3D data. Just as 2D compression techniques have enabled video communications, 3D data compression techniques have the potential to enable high-quality 3D video communications. One area which may significantly benefit from such advances is telemedicine.

By providing patients care within their own homes, *telemedicine* has the potential to reduce healthcare costs; reduce travel concerns for those patients who may be potentially ill; provide 24/7 access to medical advice; and increase access to specialized health care professionals, especially for those in underprivileged areas. A current state-of-the-art technology within telemedicine is 2D photography and video communications, however, they may not be a viable when physical measurements need to be taken and monitored over time, as they lack depth and a sense of perspective. The introduction of 3D video data to the telemedicine field has the potential to address these challenges, however, high-resolution 3D data requires significant information to represent. Given this, it is difficult to transmit 3D video data in real-time over the regular wireless networks in use today. The primary goal of this dissertation research is to overcome the constraints imposed by large 3D data sizes using novel 3D data encoding techniques. Enabled by these, this dissertation research then realized a real-

time, high-quality 3D video communications platform that could transmit compressed 3D video and color texture information over regular wireless networks.

This chapter provides an overview of this dissertation research. The driving motivations behind this research are introduced in Section 1.1. Section 1.2 will outline the objectives of this dissertation. Lastly, Section 1.3 will provide the overall organization of this dissertation.

1.1 Motivation

Two-dimensional (2D) video calls, enabled by mature compression techniques, are widely used today as a basic form of communication and collaboration. Three-dimensional (3D) video has recently emerged useful within many applications as it can provide accurate measurements and context about the real-world in which we live, work, play, and interact; however, it is currently very difficult to transmit in real-time across standard, medium-bandwidth wireless networks. If high-quality, low-bandwidth 3D video communications were achieved, one of the areas that could benefit significantly may be telemedicine.

Access to affordable, high-quality health care has been a topic of heated political debate since the early part of the 20th century. Despite discussions within political arenas, a real issue currently facing the health care industry is a shortage of physicians in the workforce [1], which by 2030 is expected to increase even more significantly [2]. Further, due to inequity in health care utilization, it will be the underserved[1] and elderly populations who are most likely to be affected by the shortages [1].

One promising solution to address the shortage of health care professionals, as well as their disproportionate distribution among the population, is to utilize communications technology. The term *telemedicine* has been defined in a variety of fashions, but generally speaking can be used to describe providing medical care over some distance

[1]In regard to health services, *underserved* refers to populations which are disadvantaged because of ability to pay, ability to access care, ability to access comprehensive healthcare, or other disparities for reasons of race, religion, language group or social status. [3]

via telecommunication technologies. An explanation for the variety of definitions may be due to the variety of health care areas in which telemedicine is employed. Given this, what is state-of-the-art in telemedicine for one medical area may differ from that of another, as they each could have differing technology requirements and constraints. For this reason, telemedicine can be thought of as a spectrum of technologies, which in practice current exist mainly between simple automated telephone systems and high-definition 2D teleconferencing. In general, these technologies have the potential to remotely diagnose, monitor, and treat patients [4]. Being able to accurately perform these tasks remotely may aid in reducing health care costs, in increasing access to physically distant specialists, in reducing health inequity, in improving patient comfort, and in the early detection of adverse symptoms or disease deterioration. Further, telemedicine technologies allow the patient to play a more active role in the management of their own health care [5]

The extent to which telemedicine is effective, however, is highly dependent upon the data transmitted when communicating. For example, a doctor may be able to diagnose, suggest treatments, and give much more accurate advice by via 2D video communication as opposed to a phone call. Two-dimensional photos and videos work well in the case of visual assessments (e.g., does the patient have a skin rash?), however, they lack depth and perspective information which may be required to form an accurate diagnosis or monitoring of a patient's ailment over time. For instance, if a patient shows their physician an open wound on their foot over a 2D video stream, it may not be possible for the doctor to adequately judge or assess how deep the wound is (i.e., if healing is progressing as it should be). If the patient were instead transmitting 3D video data to the specialist, both depth and color information could be used by the physician to measure the volume of the wound to determine its rate of healing. In general, telemedicine technologies utilizing 3D data communications have the potential to greatly influence the remote decision making of health care professionals by proving accurate measurements of adverse areas while enabling more realistic communications.

In practice, the accuracy, resolution, and field-of-view (FOV) of the 3D geometry received by the physician could have a significant impact on their ability to perform analysis and make sound medical decisions. In terms of the data acquisition itself, commercially available 3D sensors, such as the Microsoft Kinect (v1, v2), could be advantageous as they offer a large FOV, however, they are quite limited in terms of their resolution and accuracy. Alternatively, specialized structured light scanning devices can provide high-resolution, high-accuracy 3D data in real-time. However, due in part to their calibration procedures these systems are most effective at closer distances (i.e., they have a smaller FOV). In order to acquire accurate, high-resolution 3D measurements within a good FOV, new procedures to aid in the accurate calibration of a long-range structured light system are necessary.

Further, even if the captured 3D video data is of the ideal high-resolution and high-quality desired for assessment, it still needs to be efficiently transmitted from one user (i.e., the patient) to another (i.e., the physician). Since 3D data inherently requires a significant amount of information to represent, performing this in real-time, while maintaining the accuracy of the high-resolution 3D data, is a challenge. A straightforward solution would be to use little-to-no compression and transmit the data over specialized high-speed wired networks, however, this does not scale well; especially as mobile devices become increasingly used for every day tasks. A more viable and accessible solution to transmitting and receiving 3D video data would be to utilize wireless networks, such as one's home WiFi or cellular connection, however, these typically provide lower network speeds. To transmit 3D video with less bandwidth, one could simply reduce the precision of the 3D video data before transmission yet this will significantly impact the accuracy of received data. In order to realize the wireless transfer of high-quality 3D video data, methods for efficiently and accurately compressing 3D data are necessary.

1.2 Objectives

In pursuit of the above motivations, this dissertation research has the following areas of focus:

- **Develop a novel method for out-of-focus camera calibration with applications to large-rage structured light system calibration.** To obtain accurate measurements from an imaging system it must first be properly calibrated. The most extensively adopted camera calibration method involves capturing a planar 2D target, with predefined feature points, at arbitrary orientations. This approach can achieve good measurement accuracy, however, it was primarily developed for close range vision systems. To use the same calibration procedure for long range vision systems, the 2D planar target would have to be in the same order of size as the system's range. The scaling of the target thus becomes increasingly difficult in terms of fabrication accuracy, feasibility, and cost. This dissertation research aims to address these challenges with a new method for calibrating long range vision systems that uses a calibration target substantially smaller than the system's field of view that may be out-of-focus. This dissertation research further aims to utilize such a method to help extend the 3D measurement range of a structured light system. The details of this research will be introduced in Chapter 3.

- **Develop a method of encoding 3D range geometry, along with its respective texuture information, that is highly resilient to lossy compression artifacts.** Compression of 3D data is relatively new compared to the mature field of 2D image compression. Given this, one approach to 3D range data compression is to encode it within the three color channels of a standard 2D image. The mature 2D image compression techniques can then be leveraged to further compress the encoded 3D range data. As will be discussed in more detail in Chapter 2, there have been several approaches proposed for encoding 3D range data within the color channels of a 2D image, however, many

of them require extra space to store additional information, such as a texture image. In this dissertation research, we aim to develop a 3D range geometry encoding method that can accurately represent both 3D geometry and texture information within a single 24-bit color image while remaining resilient to lossy compression artifacts. The details of this research will be presented in Chapter 4.

- **Develop a method of encoding 3D range geometry that is highly resilient to lossy compression artifacts that has a computationally inexpensive decoding process.** As mentioned above, there have been several approaches proposed for encoding 3D range data within the color channels of a 2D image, however, they may require a computationally expensive decoding process, either due to (1) the inherent principles of the decoding process itself or (2) requiring substantial filtering or post-processing of the data due to a lack of resilience to lossy compression. In this dissertation research, we aim to develop a 3D range geometry encoding method that can accurately encode 3D geometry information within a single 24-bit color image in a manner that (1) is resilient to lossy compression and (2) provides for a computationally inexpensive decoding process (i.e., no matrix operations, little-to-no filtering). The details of this research will be introduced in Chapter 5.

- **Develop a novel platform for high-accuracy, high-speed 3D range video encoding and streaming over wireless networks.** The above objectives aim to help (1) capture high-resolution and high-quality 3D video data and (2) accurately and efficiently encode the 3D geometry information. To address the long-standing challenge of wireless, high-quality 3D video communications, this dissertation research also aims to develop a platform for high-quality 3D video recording, encoding, compression, decompression, visualization, and interaction. Such a platform will be able to successfully deliver photorealistic 3D video content over standard wireless networks to mobile devices. Further, this

dissertation research aims to emphasize the efficiency of the platform by developing a fully mobile implementation that allows mobile devices (e.g., iPad) to both send and receive 3D video content in real-time across wireless networks. The details of this research will be introduced in Chapter 6.

1.3 Dissertation organization

Chapter 2 of this dissertation will discuss the current state of telemedicine, previously proposed systems and methods for 3D communications, as well several different approaches to 3D data compression. Chapter 3 introduces a novel out-of-focus camera calibration method for use in long-range measurements, and how it can be used to extend a long-range structured light 3D scanning system. Chapter 4 will introduce a method for encoding 3D range geometry, along with its respective texture information, that is resilient to lossy compression. Chapter 5 will introduce another technique for 3D range geometry encoding, one that provides for a computationally inexpensive decoding process and is able to achieve high compression ratios with very low 3D reconstruction errors. Chapter 6 will introduce the novel Holostream platform developed for high-quality, low-bandwidth 3D video communications over wireless networks. Finally, Chapter 7 will summarize the contributions of this dissertation and will provide insight into several future research directions.

2. RELATED WORK

This chapter provides an overview of current technologies used within telemedicine and how the introduction of 3D video data may be able to address existing challenges. The aspects of 3D data communication have been explored, mainly in the context of 3D teleconferencing and telepresence technologies; and therefore, this chapter will also provide discussion on several proposed technologies of this type. Although great progress has been made, many of the telepresence and telecommunication technologies assume abundant network resources, and in doing so, do not focus on the transmission bottleneck imposed by the inherent size of 3D video data. To practically realize 3D communication systems, especially for applications that may require high-quality 3D video data (e.g., telemedicine), the 3D data need be efficiently and accurately represented before transmission. This chapter also will then discusses prior research in the area of 3D data compression.

2.1 Telemedicine

As introduced last chapter, telemedicine can briefly be defined as providing health care, over some distance, using telecommunication technologies. This can take place in many forms such as performing a diagnosis, remotely monitoring the progress of an ailment, or even having a general conversation. This section discusses the current state of telemedicine technologies, the challenges they face, and how 3D video data may be able to address them.

2.1.1 Current state of telemedicine

Currently, most telemedicine technologies exist on a spectrum between automated telephone systems and high-definition 2D video communications. In terms of how a specific telemedicine technology within this spectrum is used, it depends on the patient, the health care professional, and the nature of the medical issue being addressed. One approach to utilizing telemedicine is within an **asynchronous** (i.e., store-and-forward) fashion. In this scenario, a patient provides some data to their health care providers for their review at a later time. The benefit to the asynchronous method of telemedicine is that it is convenient for all parties involved: the patient and the physician do not have to schedule an appointment that fits into their respective schedules. This flexibility can have a negative impact, however, if the nature of the patient's conditions are time sensitive yet, are not reviewed in a timely manner. For instance, one study [6] evaluated the effectiveness of using static 2D images to diagnose hand injuries. An emergency room physician would take photographs of the hand injury in order to receiving guidance from an expert at a specialized center, however, it took on average 7 hours to receive their response.

Alternatively, telemedicine technologies can be used in a **synchronous** (i.e., real-time) manner to provide more natural communications, perform diagnoses, or monitoring; which is of course useful in the extreme cases when a patient's issue is time sensitive (e.g., hand injury in the emergency room). As it is more aligned with a regular face-to-face visit, synchronous communication is also useful in those scenarios where the interpersonal communication and human interaction is key to the patient's care. For example, interaction between a patient and their physician or their nurse can be used to detect symptoms of depression [5].

With the current network infrastructure and the ubiquitous nature of cameras and mobile devices, telemedicine technologies utilizing 2D photographs and video are being used in the management of acute trauma and burns, replantation candidacy of amputated digits, chronic wounds, and in free-flap monitoring [7]. Although these

technologies are useful when a patient's condition has strong visual components (e.g., being able to easily see a skin rash), they are difficult to use accurately when needing physical measurements, especially when measurements of the same area need to be monitored over time. In these scenarios, the addition of static or dynamic 3D data may useful.

2.1.2 The need for 3D within in telemedicine

Current 2D photograph and video based telemedicine methods are numerous and the technology enabling them are well adopted (e.g., iPhone cameras), however, they lack the depth and perspective information necessary for the physician to make certain evaluations. For example, Fig. 2.1 shows a photograph of a patient's arm that one

Fig. 2.1. A patient was bitten by their cat which caused their arm to become painful and swollen [7].

physician took and sent to another in search of advice [7]. The patient's cat had bitten them two days prior and their arm had become painful and swollen. Based on the photograph, the remote physician recommended an antibiotic; however, since the

remote physician had no context of the normal size of the patient's arm, the amount of swelling may be difficult to tell. Hypothetically, if the patient were to return home they could be instructed to send photograph updates to the physicians in order to monitor the effectiveness of the antibiotics. Given that each new photograph could have a different perspective, field of view, and lighting conditions, the registration between successive photographs could prove challenging, thus the remote monitoring of the healing progress would be difficult. The introduction of 3D data might then prove useful to addressing these concerns. In terms of diagnosis, if a patient's arms can be assumed as near-symmetric, the extent of swelling could be determined by comparing a 3D scan of the bitten arm with a scan of the arm that wasn't bitten. To monitor the influence of the antibiotic, the volume of iterative 3D scans could be assessed.

Telemedicine has also been explored for diagnosing and monitoring ulcers (chronic wounds). Chakraborty et al. [8] for example proposed a classification framework that could identify wound tissue and then predict the wound's status. Although successful predictions were achieved, this method relies on both the manual and automatic processing of 2D images, especially on the color information stored within. For this reason, when monitoring wound healing over time, it would be of critical importance that the images are taken with near identical perspectives, orientations, and lighting conditions. This is a feat which is difficult in practice, especially if the patient in their home and is not expertly trained to do so. Further, the wounds were initially assessed by a clinician who recorded their volume (length, breadth, and depth). This information is either ignored or difficult to use with accuracy when only using 2D images or video to remotely monitor the chronic wound, however, it could prove very useful in monitoring the healing progress. For example, the ulcer on the foot in Fig. 2.2(a) appears spatially small on the skin's surface, however, it is difficult to determine its depth. To address such issues, static 3D frames or 3D video could be used, allowing for the iterative monitoring of wound volume. This direction has been recently explored [9, 10], and the results of one study suggest that using 3D data to

(a) (b)

Fig. 2.2. Using telemedicine to assess foot ulcers. (a) Although the ulcer is spatially small on the patient's foot, it is difficult to determine the depth of the ulcer; (b) 3D reconstruction of a foot ulcer by [9].

assess an ulcer is more in line with an in-person clinical assessment (as opposed to monitoring using 2D photographs) [10]. Figure 2.2(b) shows a 3D captured foot ulcer with color texture mapping [9]. In this system, the 3D mesh can be interacted with (i.e., panned, rotated, and zoomed) its measurements can be taken.

The usage of 3D data for telemonitoring ulcers is very promising as the same fundamental techniques can be immediately useful within other medical areas. One issue that remains, however, is the size of 3D data. Even in the asynchronous scenario, one or a small set of 3D scans may take a significant time to upload to the online portals of healthcare professionals. Further, if 3D video data were desired to be used in a synchronous scenario, data sizes are of an even greater concern. The aspects of 3D data acquisition and transmission have indeed been explored, however mainly in the application area of 3D teleconferencing and telepresence technologies. The next section provides a discussion on several of these proposed technologies.

2.2 3D Telecommunication

To achieve 3D communications one has to achieve the ability for real-time 3D capture, transmission, and visualization. Each one of these by itself requires overcoming many technical challenges, both in hardware and software. Creating a unified technology which can simultaneously perform each one is of a greater challenge. Over the years, however, there have been quite a few state-of-the-art technologies proposed for addressing these challenges, often evolving with advancements made in 3D capture technologies. Several of the state-of-the-art 3D communication technologies will be discussed next, beginning with the seminal *office-of-the-future* concept. Following that, the technologies will be categorized by the capture techniques they employed to acquire 3D reconstructions.

2.2.1 The Office-of-the-future

One of the seminal works to address the challenges associated with 3D communications and telepresence was the *office of the future* [11]. This work was influential as it not only presented how 3D communication technologies could be composed, it also described many future directions and ideas for both improvements and exploration which many works followed, including several of those discussed later in this section. The authors envision an office environment where the lights could be replaced with camera-projector pairs, allowing for the office to simultaneously be 3D captured and used as a display surface. In the proposed vision for the office of the future, this 3D capture would be transmitted and re-displayed within another user's office, providing an immersive collaboration experience as illustrated in Fig. 2.3. Due to both hardware and software limitations at the time, the office of the future system could only achieve a capture rate of 3 Hz and did not transmit data from one instance of the system to another; however, it provided great ideas, insights, and prototypes that inspired the field in the years to follow.

(a) (b)

Fig. 2.3. A sketch of the proposed office of the future [11]. (a) Cameras and projectors on the ceiling and walls of the office 3D scan the user, their workplace, and the objects within; (b) the projectors turn the walls into a virtual collaborative working environment by projecting the 3D scans of other users, their offices, and objects into the user's physical space.

2.2.2 Reconstruction via Array of Cameras

Following the vision established in the office of the future, Towles et al. [12] proposed a system for remote 3D tele-collaboration. Their system used an array of seven cameras to compute five depth maps per frame via a trinocular stereo reconstruction algorithm. These depth maps, along with a registered color texture image, were then used to reconstruct a colored 3D point cloud. This system was able to achieve 15,000-20,000 3D points per depth map, which could be acquired at 2-3 Hz. The depth maps, color texture images, and calibration data were transmitted without compression using TCP/IP at a target bitrate of 75 Mbps. Once received at the remote end, the calibration data was used to reconstruct point clouds from the depth maps and the texture image was used to provide color information to the points. The reconstructed cloud was then visualized on a large window-like display to allow for realistic, life-sized communication between users, as shown in Fig. 2.4. Although the

Fig. 2.4. A user sits in front of the system developed by Towles et al. [12] for remote 3D tele-collaboration. The system uses an array of cameras to capture a colored 3D point cloud of the user which is then transmitted across a network to remote participants.

hardware costs were high and the rate of acquisition was low, the reconstructed point clouds were of reasonable density and captured a significant area around the user. Further, this system was one of the first to actually address the physical transmission of acquired 3D data, although the bitrates it assumed were still quite high, even by today's standards.

Shortly afterward, the blue-c telepresence portal was proposed by Gross et al. [13] as a set of 3D telepresence technologies which simultaneously achieves 3D capture and 3D display via an immersive CAVE^{TM}-like environment. The environment consists of three large rectangular projection screens which can be switched between an opaque state (for display via projection) and a transparent state (for capture). By switching states quickly enough, both display for and capture of the user within can be achieved. A virtual environment is displayed around the user using three pairs of projectors, each of which projecting onto the screens while they are in an opaque state. When the screens are in a transparent state, an array of 11 cameras on the outside looking inward captured the user within. As in system above, stereo reconstruction algorithms were then used to obtain 3D measurements of the user. Five additional cameras are used

Fig. 2.5. The blue-c telepresence portal [13]. Three large projection screens and projectors outside of the screen are used to display a virtual environment to the user. The cameras positioned on the metal scaffolding outside the system are used to capture the geometry of the user within which will be transmitted for display within a remote blue-c node.

to recover texture and to aid in calibration, brining the total number of cameras used by one blue-c node to 16. An entire blue-c system (screens, cameras, and projectors) can be seen in Fig. 2.5. The 3D video processing pipeline developed was able to reconstruct 25,000 points at 5 Hz or 15,000 points at 9 Hz. To reduce the amount of data transmitted, updates to a shared 3D environment were streamed, instead of each newly acquired 3D reconstruction. The operations streamed to change points in the shared reconstruction, such as *insert*, *update*, and *delete*, were transmitted over a 100 Mbps Ethernet connection. Although these changes required a bitrate of only 2.5 to 12.5 Mbps to stream these, this would scale drastically if either the number of points or frame rate increased. Remote users within another telepresence node (blue-c or other) could then see and collaborate with the captured user within their shared virtual environment. This technologies developed as part of the blue-c portal achieved many technical hurdles, however, the costs associated with the 16 cameras and the three large projection screens make this system very expensive to implement, especially at scale.

2.2.3 Reconstruction via Digital Fringe Projection

Although the above technologies began to realize the acquisition, transmission, and display infrastructures required for 3D video communication systems, none of them could not do so at natural video rates (i.e., 24 Hz or higher) or at high accuracies (e.g., mm) due to the limitations of the sensing technologies used. By the mid-2000's, however, real-time, high-resolution 3D scanning was made possible due to developments in structured light systems using digital fringe projection technologies [14].

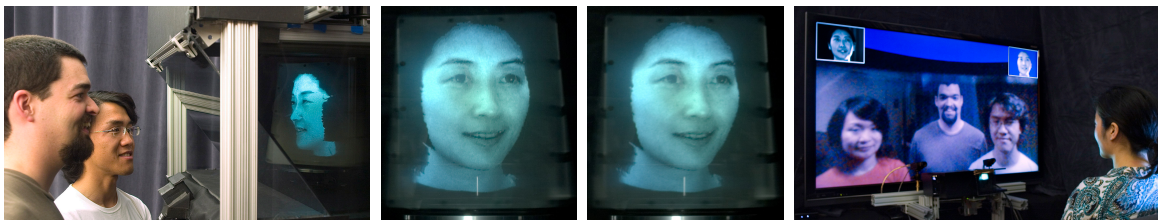


Fig. 2.6. The 3D video teleconferencing system proposed by Jones et al. [15].

Taking advantage of this, Jones et al. [15] used structured light and DFP techniques within a 3D teleconferencing system to provide high reconstruction accuracies at real-time speeds. The structured light scanner acquired accurate 3D representations of a user's face in one location which was then delivered to a remote system for visualization on an autostereoscopic 3D display. A 2D video feed of the remote users was provided to the 3D captured user to allow for two-way communication. Figure 2.6 provides several images of the entire platform. The structured light scanner was able to reconstruct very dense depth maps (640×480) with capture rate of 30 Hz. After a 3D scan was acquired, the downsampled depth map and its associated texture image transmitted to the 3D display projector for display over a high-speed wired network. The downsampling of the depth map is important as it needed to be rendered at

thousands of frames per second, requiring a very powerful computer, in order to be displayed on the autostereoscopic display.

2.2.4 Reconstruction via Dot Projection

Although DFP techniques could be used to acquire high-resolution 3D data of users in real-time with great accuracy, such systems may require a certain level of technical know-how in order to perform accurate calibration and may be relatively costly to construct. Hence, when commodity depth cameras made their introduction (e.g., Microsoft Kinect in 2010), many researchers shifted to them due to their low cost and ease of use.

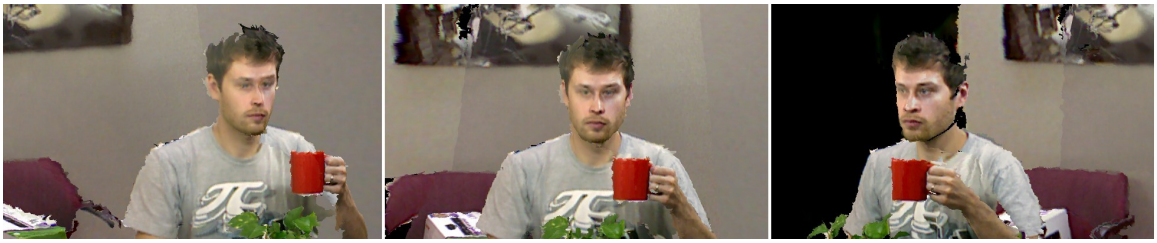


Fig. 2.7. Three different viewing perspectives of a single reconstructed scene, as captured by the telepresence system proposed by Maimone and Fuchs [16]. Depth maps and color images captured by five Microsoft Kinect devices are merged to generate these meshes at up to 30 Hz.

Maimone and Fuchs, for example, presented a telepresence system which offered real-time 3D capture using an array of commodity depth cameras [16]. The capture system uses five Microsoft Kinect devices which each provide a depth map and color image. These are then processed and merged together into a single colored mesh which is then shown to a remote participant on a large display. Figure 2.7 shows three different viewpoints of a mesh as reconstructed from the five Kinect devices. On the remote end, one single Kinect is used to perform head and eye tracking of the participant which enables the captured 3D mesh to be rendered in a realistic manner. For example, if the remote participant moved to their left, the rendering

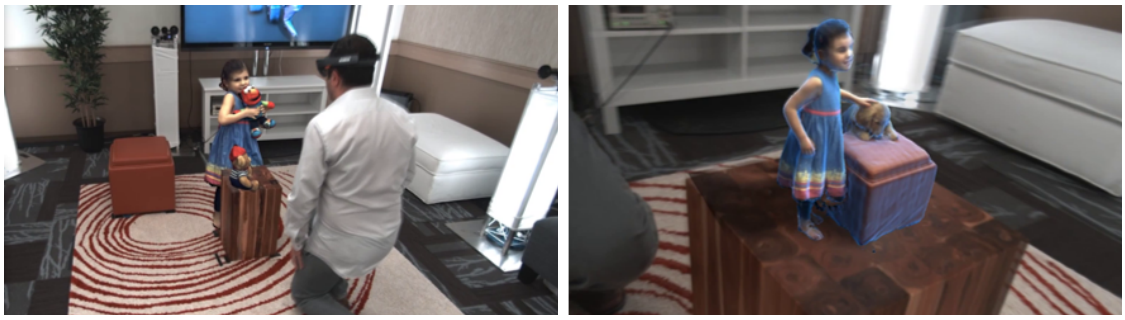
would adjust its perspective (e.g., rotate about the y-axis away from them). This is done to mimic the viewpoint that the remote participant might see as if they were sitting physically moving in front of the 3D captured user. The proposed system was able to impressively integrate depth and color data from five independent devices into a single colored mesh. However, in addition to the interference issues between Kinect devices causing holes in the captured depth maps, this system did not actually transmit any data. In fact, only a single computer was used to drive both the capture and display components.

Fig. 2.8. The telepresence system proposed by Beck et al. [17] enabled two groups of people interact and solve problems with one another, in 3D, within a virtual shared world. An array of Kinect devices captures and generates a single mesh of each group which is then transmitted to and visualized within the remote node.

Beck et al. [17] expanded on this work which allowed for two remote groups of people to interact together within one virtual, collaborative space as shown in Fig. 2.8. An array of Kinect devices captured the users of one location, unified all of the data into a single reconstruction (similar to the reconstruction method used in [16]), and then transmitted the resulting mesh and color images. This technique generated approximately 5.3 MB per frame, and thus required a high-speed (the authors used 10 Gbps wired Ethernet) connection to transmit. Once received by the remote participants, this data was rendered from within their perspective of the virtual shared

world. The end result allowed two groups of people to interact and solve tasks with one another, as though they were face to face.

(a) (b)



(c) (d)

Fig. 2.9. The Holoportation system proposed by Microsoft [18]. (a) An array of eight 3D sensors are used to capture the user and reconstruct a single mesh; (b)-(c) the local user is visualizing the transmitted 3D mesh of a remote user within an augmented reality (AR) headset in real-time; (d) a recorded 3D video sequence, from the interaction in (c), is played back in the local user's physical space within their AR headset, at a scale of their choosing.

Most recently, Microsoft proposed their Holoportation system [18] which captures a mesh of a user(s) or object(s) in one location and transmits it to remote users for visualization within augmented reality (AR) headsets. The capture setup of the Holoportation system includes eight 3D scanners, each consisting of two near infrared (NIR) cameras, one color camera, and one pseudo-random dot projector. Similar

to the Kinect, the projected patterns of each scanning system randomly encode the entire scene, allowing for each of the eight NIR camera pairs to reconstruct a depth map at 30 Hz. The eight depth maps are then merged together into a single 3D mesh as shown in Fig. 2.9(a). The 3D mesh (consisting of approximately 60,000 vertices) and color texture data require about 2 MB and 3 MB, respectively, to represent each frame. To transmit this data, TCP is used on a 10 Gbps line, requiring 1-2 Gbps per capture stream. A PC on the receiving end receives this data and renders it for display on the remote user's AR headset, allowing them to see and interact with the captured data within their own current, physical environment as shown in Fig. 2.9(b)-2.9(d).

2.3 3D Compression

Telepresence and 3D telecommunication technologies have progressed closer to the realizing immersive and natural 3D communications, however, one of the major issues that has not been addressed over the years is how data sizes can be drastically reduced in order to alleviate hardware dependencies (high-speed connections). Although technologies such as 3D reconstruction have made significant advances, it appears as though most new telepresence technologies have operated off of the same assumption that Towles et al. did in 2002 [12] when it comes to actually transmitting the reconstructed 3D data: "Our current assumption is one that foresees a future containing abundant network resources, and we are transmitting the 3D video data in an uncompressed format." A combined lack of focus regarding the efficient transmission of acquired data has made it difficult for these state-of-the-art technologies to realize 3D telecommunications over medium-bandwidth wireless networks at high qualities and in real-time. Thus in tangential communities, researchers have proposed methods to compress 3D data.

2.3.1 Mesh-based Compression

One conventional method to represent 3D data is within a mesh format. A 3D mesh is defined by a set of vertices (i.e., coordinate positions) and a set of edges (i.e., how the vertices are connected). On top of storing vertices and their connectivity, mesh formats often store additional properties about the 3D object, such as a texture coordinates, a normal map, or vertex color information. Standard mesh file formats (e.g., OBJ, STL, PLY) are based on a simple listing of the above data. Representing a mesh in an efficient manner, however, has been an active area of study for the past several decades.

The first mesh compression techniques focused on efficiently encoding how the vertices of the mesh are connected to one another. A well-designed connectivity encoding method aims to find optimal traversals of a mesh's vertices. This reduces redundancy in the connectivity information and thus reduces the overall file sizes. Many methods have been proposed to perform connectivity encoding such as triangle-strip [19–21], spanning tree [22], valence encoding [23, 24], and triangle traversal [25, 26]. These methods efficiently encode a mesh's connectivity information, but recall a mesh also stores vertex information. To encode the vertex positions, mesh connectivity methods typically follow a three-step procedure of coordinate quantization, prediction, and entropy encoding [27, 28]. The encoded vertex positions and connectivity information can then be saved and used to reconstruct the mesh at a later date.

In connectivity encoding methods, the aim is to optimize the storage of the mesh's connectivity information first; the vertex data is then encoded following the traversal as determined by the connectivity encoder. Although these connectivity-driven approaches do reduce the amount of data needed to represent a mesh, they may not be overall optimal as the size of a 3D mesh is generally more influenced by the vertex positions [27, 28] and not the connectivity information. In this case it then makes sense to optimally encode vertex positions first, and the connectivity information can be encoded following the order in which the coordinates are encoded. Kronrod and

Gotsman [29] have proposed such a geometry-driven mesh encoding method which optimizes the predictions between adjacent vertices. Connectivity information is then encoded following the path of optimal coordinate predictions. Since this connectivity encoding may not provide an optimal traversal of the mesh, a small penalty is imposed due to redundancy. It was found, however, that optimally encoding coordinate positions provided much more compact meshes, even with the small penalty [29].

Although mesh compression techniques have been and are still being widely investigated, they are still not efficient enough for real-time, high-resolution 3D data transmission. For example, the fast mesh compression method proposed by Mekuria et al. [30] was able to transmit dynamic 3D data, but even at low vertex densities (72K) and frame rates (12 Hz) the bit rates were still quite high (35 Mbps). The method achieved by Collet et al. [31] was able to achieve streamable bit rates less than 15.6 Mbps, however, it took on average over 25 seconds to encode each 3D frame.

2.3.2 Image-based Compression

In order to reduce encoding time, the complexity of mesh compressions method may be further reduced if exact restoration of the mesh's connectivity information is not required. This complexity can be reduced even further if the data has an underlying structure from which connectivity information can automatically be derived from. For instance, a grid or pixel structure is often assumed for 3D range data captured by a camera. The connectivity information in this case can be generated on-demand, by knowing this structure, without actually having to store it. The only data that needs to be encoded then is the 3D coordinate information. Efficiently encoding this 3D data is still a relatively new area of study, however, the field of 2D image compression is quite mature. Given this, if 3D coordinate information can be stored as regular 2D images, the advanced 2D image compression techniques could be leveraged.

Virtual Digital Holography Compression Methods

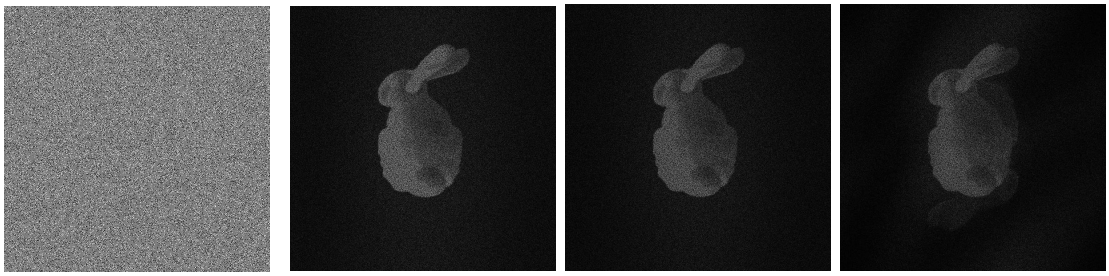
In the field of phase-shifting interferometry digital holography (PSIDH), 3D hologram information is encoded within a 2D complex wavefront. This wavefront can be computed from multiple phase-shifted interference, or *fringe*, patterns captured by a camera. A complex wavefront can reconstruct a 3D hologram even though they are 2D and nature. Since the wavefronts are composed of floating point complex numbers, compression methods are still necessary; however, the inherent grid structure of the wavefront allows the 3D geometry compression problem to be simplified to a 2D compression problem.

Darakis and Soraghan [32] proposed a hologram compression method which uses JPEG and JPEG2000 to directly compress camera captured fringe patterns. The compressed 2D fringe patterns can then be stored and used to reconstruct the complex wavefront when needed. This method is advantageous as it avoids directly compressing the wavefront with its complex values. This method also offers various levels of flexibility as the JPEG quality level in use can be user defined. To achieve the smallest file sizes, however, these JPEG quality levels need to be quite low which leads to error on the recovered complex wavefront. Further, the compressed data size is proportional to the number of fringe patterns captured. For example, if nine fringe patterns are used to reconstruct a wavefront instead of three, the number of images to store would be three times as many.

Darakis and Soraghan [33] proposed another hologram compression method which instead compresses the complex wavefront at the object's reconstruction plane. This method works by quantizing the complex wavefront data (a lossy procedure) and then losslessly encoding it using the Burrows-Wheeler transform [34]. Out performing JPEG compressing the fringe patterns directly [32], this method achieved approximately a 26:1 compression ratio with a normalized root-mean-square error of approximately 0.1. Further, this method kept intact the hologram's natural capability to be reconstructed at different depth planes and from different perspectives.

The above compression methods are applied to 3D hologram data coming from physical digital holography systems. Physical systems are naturally impacted by lighting conditions, surface textures, and speckle noise which in turn may affect the efficiency of hologram compression methods. In order to by pass these artifacts, a virtual digital holography system can be used to generate computer-generated holograms (CGH). A CGH is computed through numerically simulating how light reflects and propagates off of a virtual 3D object. These methods are advantageous as they can be generated from any arbitrary 3D object and are computed in a completely ideal and controlled manner. Recently, methods have been proposed for compression CGHs using JPEG [35] and HEVC [36], similar to the JPEG and JPEG2000 digital hologram compression method mentioned above. Figure 2.10 illustrates the HEVC method proposed by Xing et al. [36]. A virtual object, in this case the Stanford bunny, is encoded into a computer generated hologram via three phase shifted digital holograms, one of which can be seen in Fig. 2.10(a). A sequence of the digital holograms are then compressed into a HEVC video and stored. When reconstruction is desired, the video can be decoded to recover the digital holograms, generate the complex wavefront, and re-project the geometry. For example, Fig. 2.10(b) shows this reconstruction when the data was uncompressed. Figure 2.10(c) and Fig. 2.10(d) show the reconstructions when HEVC compression was used along with a quantization parameter (QP) of 40 and 43, respectively.

Although these compression methods are quite effective, generating a CGH is a costly operation, both in terms of computational complexity and memory [37]. These costs can be reduced by using a graphics processing unit (GPU) [38–40], however, these methods are still limited when generating high-resolution holograms. One could simply compute lower resolution holograms to avoid high computation costs, however, this would limit the number of possible viewing angles since the viewing angle of the reconstructed hologram is proportional to the spatial resolution of the CGH [40]. Therefore, in order to encode many viewing angles and a large number of 3D points, the spatial resolution of the CGH would have to be very large and is



(a) (b) (c) (d)

Fig. 2.10. Compression computer generated holograms using HEVC [36]. (a) One of the three computer generated phase-shifted digital holograms representing the Stanford bunny; (b) the bunny reconstructed from the digital holograms without compression; (c) the bunny reconstructed when the digital holograms were compressed with HEVC and a quantization parameter (QP) of 40; (d) reconstructed from HEVC and a QP of 43.

thus very costly to compute. Lastly, all hologram compression methods (physical and computer-generated) are greatly impacted by the noise caused by speckle (can be seen in Fig. 2.10(a), for example). This noise limits the encoding efficiency of the 2D lossy image compression methods making it difficult to achieve very high compression ratios (i.e., low file sizes) while also preserving the fidelity of encoded data.

Virtual Digital Fringe Projection (DFP) Compression Methods

Although they are similar in some ways to holography-based 3D compression methods, digital fringe projection (DFP) 3D range data compression methods are usually more advantageous as they (1) can establish a one-to-one correspondence between a 3D coordinate and a pixel on the 2D image to be compressed; (2) can eliminate the noise caused by speckle; and (3) can achieve much higher compression ratios utilizing standard 2D image compression techniques. In a manner similar to using a virtual digital hologram system to simulate a CGH, a virtual DFP system

can be used to convert 3D range data into a 2D image. Once here, the image can be further compressed using standard 2D compression techniques (e.g., JPEG, PNG).

Karpinsky and Zhang [41] used such a virtual DFP system to encode 3D geometry into the three 8-bit channels of a standard 2D color image. This approach was advantageous as the virtual system settings could all be precisely defined (e.g., zero ambient light, uniform reflectivity, ideal camera). Their method used two of the color channels to encode the 3D data via sine and cosine functions of the phase map. The third channel was then used to store fringe order information which allows for the proper decoding of the encoded phase map via pixel-by-pixel phase unwrapping. An example of the this method can be seen in Fig. 2.11.

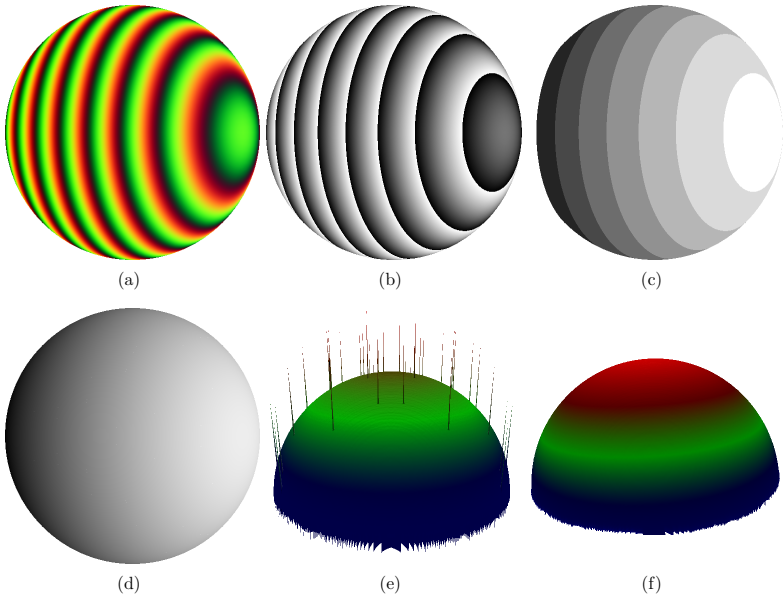


Fig. 2.11. The pipeline of the composite phase-shifting algorithm for 3D shape compression proposed by Karpinsky and Zhang [41]. (a) The 2D image containing encoded 3D information; (b) phase map decoded from the red and green channels of (a); (c) fringe order information (stair map) decoded from the blue channel; (d) the unwrapped phase map from (b) and (c); (e) recovered 3D shape; (f) recovered 3D shape after filtering.

To make this method more robust to lossy image compression and to enable lossy 3D video compression, Karpinsky and Zhang later improved this method by wrapping the fringe order information within a cosine function [42, 43]. Based on the same principles of a virtual DFP system, Wang et al. [44] proposed a two-channel method, using the first channel to represent the 3D range geometry and using the second to store fringe order information. This method works well when saving the encoded 2D image using a lossless method, however, a post-processing error compensation framework is needed to alleviate decoding errors. Further, the fringe order information in the second channel may contain sharp intensity changes which significantly impacts this method's ability to utilize lossy image compression. Hou et al. [45] also proposed a two-channel method which directly encodes the wrapped phase of a 3D range data frame into the first channel; the second channel is again used to store fringe order information. As with the method proposed by Wang et al., this method works well when storing the output 2D image losslessly. However, the sharp 2π discontinuities within the first channel's wrapped phase limits this method's ability to leverage lossy image compression.

The above virtual DFP methods are advantageous as they can encode 3D geometry information within the color channels of a regular 2D image, allowing existing 2D image compression techniques to further be leveraged in order to reduce file sizes. Even though the virtual DFP system can be precisely defined and controlled, it still faces drawbacks associated with triangulation and resampling. Further, these methods typically use a graphics rendering pipeline to create the encoded 2D image, and thus this image may be limited to the maximum resolution the computer's video card can support. This may then lead to the resolution of the encoded 2D image, and subsequently to the resolution of the reconstructed 3D geometry, being smaller than that of the original 3D data.

To bypass the issues caused by triangulation (e.g., holes on the data caused by regions not being visible to both the camera and projector) of the virtual DFP system, Zhang proposed the direct depth encoding method [46]. Leveraging the OpenGL

pipeline, this method samples the depth (z) map of 3D range geometry directly, encoding the sampled data into two color channels of a 2D image via the sine and cosine functions. The fringe order information, needed for phase unwrapping, is encoded into the third color channel as with the previous methods. Although this method still requires resampling the depth data, it allows for the use of an arbitrarily high resolution image to represent the 3D data. Ou and Zhang proposed a similar 3D range geometry compression method which encodes the scaling factor (s) map, instead of a depth map, of a physical structured light system [47]. Similar to the direct depth encoding method [46], the s map is encoded into two color channels of the output 2D image via the sine and cosine functions; the third channel is used to represent fringe order information. This method is capable of achieving the same spatial resolution as the camera without resampling, however, it may be more computationally expensive than similar methods. This is due to the method's need to compute 3D coordinates, pixel-by-pixel, during both the encoding and decoding processes.

The above methods reduce the dimensionality of the 3D data by encoding into the color channels of a standard 2D image. From here, mature image compression techniques can be leveraged to reduce file sizes further. To maintain high quality decodings, the above methods may use lossless image compression (e.g., PNG) to store the encoded 2D image, however, this limits achievable compression ratios and portability to video encoding (as most video codecs are lossy in nature). To achieve higher compression ratios and to be applicable to video codecs, lossy image compression may be used (e.g., JPEG), however, each of the above methods require large amounts of filtering to reduce compression artifacts imposed on the decoded data. This post-processing increases the computational complexity of the methods, thus increasing their overall decoding speeds.

2.4 Conclusion

Just as 2D image compression technologies unlocked the potential for ubiquitous image transfer and video streaming, 3D data compression technologies have the potential to realizing 3D data and 3D video transmission. Once 3D data compression can be achieved with high efficiency (small data sizes) and high quality (low error in the reconstructed geometry), high-quality. low-bandwidth 3D video communications can be realized, potentially enabling many new applications, including those within telemedicine, the digital arts, homeland security, and remote surgery.

3. METHOD FOR OUT-OF-FOCUS CAMERA CALIBRATION

As previously discussed, the accuracy, resolution, and FOV of acquired 3D geometry would greatly impact the effectiveness of a 3D communication system. In the context of telemedicine, these qualities of received 3D data may impact a physician's ability to perform analysis and to make sound medical decisions. Structured light systems would be advantageous to use as they can achieve both high-resolution and high-accuracy. However, these systems are more effective at closer distances, due in part to their calibration procedures. This chapter introduces our novel method for out-of-focus camera calibration which can be used to accurately and more feasibly calibrate large range vision systems. Additionally, how this method can aid in calibration of large range structured light systems will also be discussed. The major contents of this chapter were originally published in *Applied Optics* [48] (also listed as journal article [J3] in "LIST OF PUBLICATIONS").

3.1 Introduction

Two and three dimensional vision systems typically use at least one calibrated camera to capture images for information analytics. Measurement accuracy heavily hinges on the accuracy of camera calibration, thus accurate and flexible camera calibration has been extensively studied over the past few decades.

Accurate camera calibration can be carried out by using highly-accurate fabricated and measured 3D calibration targets [49,50]; since the 3D dimensions of these targets are known, the transformation from 3D world coordinates to the 2D imaging plane can be estimated through optimization methods. However, fabricating such highly-accurate 3D targets is sometimes difficult and usually very expensive. Since using a

3D target is equivalent to moving a 2D planar object perpendicular to its surface, Tsai [51] proposed a method to use 2D calibration targets with rigid out-of-plane movements to accurately calibrate a camera. By using a 2D flat surface, Tsai's method simplifies the target fabrication process since a 2D flat surface is easier to obtain. However, such a method requires the use of a high-precision translation stage which is often expensive. To further simply the camera calibration process, Zhang [52] proposed a flexible camera calibration method that allows the use of a planar 2D target with arbitrary poses and orientations, albeit it requires some known dimension feature points (e.g., checkerboard or circle patterns) on the target surface. Image processing algorithms are then used to detect those feature points for camera calibration. Zhang's method is, by far, the most extensively adopted method due to its flexibility and ease of use.

Recently, researchers have developed more flexible camera calibration approaches by using unknown feature points or even imperfect calibration targets [53–56]. Instead of fabricating a calibration target, researchers have demonstrated that active targets (e.g., digital displays) can also be used to accurately calibrate cameras [57] and can further improve calibration accuracy [58] since feature points can be more accurately defined and located. Similar active pattern approaches have also been used to calibrate a system using a fisheye lens [59].

To our knowledge, the state-of-art camera calibration methods were primarily developed for close-range vision systems (i.e., the sensing range is usually rather small such that the calibration target used to calibrate the camera can be accurately and feasibly fabricated). However, long-range vision systems are becoming increasingly used for applications such as navigation and large scale measurements. Given this, if high levels of accuracy are required for long-range vision systems, a camera calibration method for such systems is required. The challenge then becomes the fabrication of a calibration target that is in the same order of size as the system's range. Obviously, scaling the calibration target to the scale of a long-range system becomes increasingly difficult in terms of fabrication accuracy, feasibility, and cost.

This chapter presented a method that allows the use of a smaller calibration target for large-range vision system calibration. The aforementioned state-of-the-art camera calibration methods assume that the camera is at least nearly focused on the calibration target, and thus they fail if the camera is substantially defocused. The proposed method enables the calibration of an out-of-focus camera to conquer the challenges of calibrating large-range vision systems. By allowing for the placement of the calibration target closer to the camera than the sensing plane, the calibration target size can be substantially smaller, as illustrated in Fig. 3.1. Similar to the previously proposed calibration methods that use active targets [57–59], we also use an active digital display (e.g., a liquid crystal display monitor) to generate fringe patterns which encode feature points into the carrier phase which can be accurately recovered even if the fringe patterns are substantially blurred (i.e., camera is substantially defocused). Instead of using phase-shifted sinusoidal patterns, we use phase-shifted square binary patterns for phase generation to enhance fringe contrast when the patterns are substantially defocused and to eliminate the influence of digital display nonlinearity. Experimental results demonstrate that the proposed camera calibration method can accurately calibrate a camera regardless of the amounts of defocusing.

Section 3.2 explains the principles of the proposed out-of-focus camera calibration method. Section 3.3 shows some simulation results to validate the proposed method. Section 3.4 presents experimental results to further validate the proposed method. Lastly, Section 3.6 summarizes the research.

3.2 Principle

This section thoroughly explains the principle of the proposed method. Specifically, we will present the standard pinhole camera model followed by the feature point encoding and sub-pixel feature point extraction framework we developed to calibrate a camera regardless of its amount of defocusing.

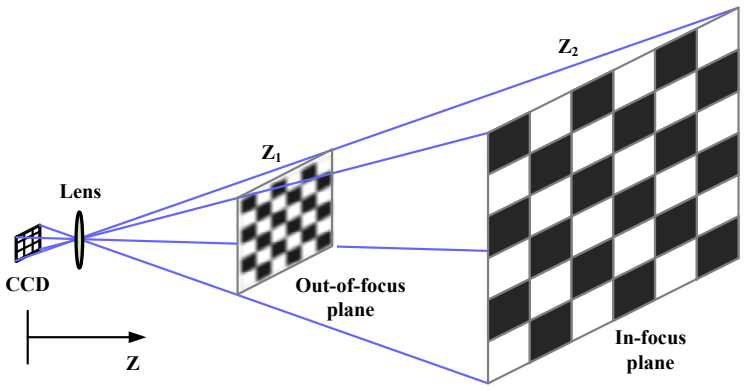


Fig. 3.1. Illustration of placing the calibration target at different distances from the camera. Compared to putting the calibration target at the focal plane (Z_2), the calibration target dimensions could be substantially smaller if the calibration target is placed at the out-of-focus plane (Z_1). However, if the target is placed at Z_1, the captured images are blurred, failing the current state-of-the-art calibration methods that assume the camera is at least nearly focused.

3.2.1 Camera lens model

In this research, we use a well-known pinhole model to describe a camera lens. This model essentially describes the relationship between 3D world coordinates (x^w, y^w, z^w) and its projection onto the 2D imaging coordinates (u, v). For a linear system, without considering lens distortion, the pinhole model can be mathematically described as,

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix}. \tag{3.1}$$

Here, s is the scaling factor; f_u and f_v are respectively the effective focal lengths of the camera along u and v directions; γ is the skew factor of u and v axes, for modern cameras $\gamma = 0$; and (u_0, v_0) is the principle point. In this equation,

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.2}$$

represents the rotation matrix from the world coordinate system and the camera lens coordinate system; and $\mathbf{t} = [t_1, t_2, t_3]^T$ describes the translation from the world coordinate system and the camera lens coordinate system.

If the camera lens is nonlinear, its distortion can be modeled as,

$$\mathbf{D} = \begin{bmatrix} k_1 & k_2 & p_1 & p_2 & k_3 \end{bmatrix}^T, \tag{3.3}$$

where k_1, k_2 and k_3 are the radial distortion coefficients, which can be rectified by

$$u' = u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \tag{3.4}$$

$$v' = v(1 + k_1 r^2 + k_2 r^4 + k_3 r^6). \tag{3.5}$$

Here, (u', v') are the camera coordinates after nonlinear distortion corrections, and $r = \sqrt{(u - u_0)^2 + (v - v_0)^2}$ represents the absolute distance between the camera point and the origin. Similarly, tangential distortion can be corrected using the following formula:

$$u' = x + [2p_1 uv + p_2(r^2 + 2u^2)], \tag{3.6}$$

$$y' = y + [p_1(r^2 + 2v^2) + 2p_2 uv]. \tag{3.7}$$

3.2.2 Feature point encoding

As introduced in Sec. 3.1, one of the most extensively adopted camera calibration approaches uses a planar object with a number of feature points with known dimensions on a flat plane [52]. Typically, a checkerboard or a circle pattern is printed

on a flat surface [60]. A method such as this first captures a sequence of images of the calibration object placed at different poses. Then, image processing is performed to detect the known feature points within the sets of images. Lastly, optimization algorithms are used to estimate camera calibration parameters.

Since only a number of known-dimension feature points are needed for calibration, these feature points can be generated digitally by a digital display device (e.g., an LCD monitor). If the monitor is flat, the same calibration approach can be adopted for camera calibration. Furthermore, since the feature points are discretely defined by LCD monitor pixel locations, as long as those pixels can be located, no real circle pattern or checkerboard is necessary for camera calibration. In this research, we use phase information to define such feature points.

Figure 3.2 illustrates the framework of using phase to encode desired feature points, and we propose to use such a framework to calibrate an out-of-focus camera. Briefly, the desired feature points for camera calibration are encoded by horizontal and vertical phase maps. These phase maps are then further carried along by phase-shifted fringe patterns which can be used to recover the original phase using phase-shifting algorithms.

Phase-shifting algorithms are extensively adopted in optical metrology mainly because of their accuracy and robustness to both noise and ambient lighting effects. In general, for N-equally phase-shifted fringe patterns, the fringe patterns can be mathematically described as,

$$I^i(x,y) = I'(x,y) + I''(x,y)\cos(\phi + 2i\pi/N), \tag{3.8}$$

where $I'(x,y)$ is the average intensity, $I''(x,y)$ is the intensity modulation, $i = 1, 2, \cdots, N$, and $\phi(x,y)$ is the phase to be solved by

$$\phi(x,y) = -\tan^{-1}\left[\frac{\sum_{i=1}^{N} I^i \sin(2i\pi/N)}{\sum_{i=1}^{N} I^i \cos(2i\pi/N)}\right]. \tag{3.9}$$

This equation produces the wrapped phase ranging from $-\pi$ to $+\pi$. To obtain the continuous phase without 2π discontinuities, one can use a spatial or temporal phase

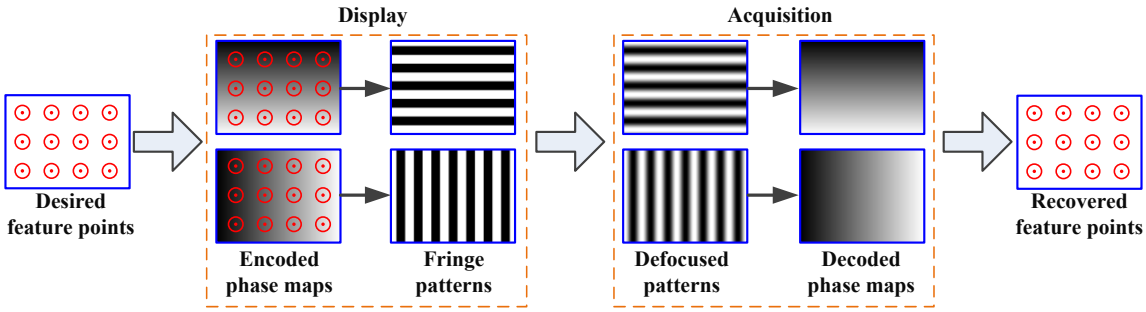


Fig. 3.2. Proposed framework for out-of-focus camera calibration. The target feature points are carried by the uniquely defined horizontal and vertical phase maps. Each phase map is the resultant of a set of phase-shifted binary structured patterns. These patterns are then displayed on an LCD monitor. The out-of-focus camera captures those structured patterns, which will be blurred according to the camera's level of defocus. These captured patterns are then used to recover horizontal and vertical phase maps from which the encoded feature points are decoded. The out-of-focus camera can then be calibrated using the recovered feature points.

unwrapping method. The essence of phase unwrapping is to find the fringe order $k(x, y)$ for each pixel, so that the phase can be unwrapped as

$$\Phi(x, y) = \phi(x, y) + k \times 2\pi. \tag{3.10}$$

The fundamental difference between spatial phase unwrapping and temporal phase unwrapping is that the spatial phase unwrapping algorithm finds $k(x, y)$ by analyzing the difference between the point to be processed and its neighboring pixels. In other words, the phase obtained using a spatial phase unwrapping algorithm is relative to one point, and thus the unwrapped phase is often called the *relative* phase. Temporal phase unwrapping algorithms, in contrast, uniquely find the phase values for each independent point without referring to the phase information of any neighboring pixel; thus, such a method can recover *absolute* phase. Given this, to uniquely carry phase information, using absolute phase is necessary.

In this research, we adopted a temporal phase unwrapping method which uses gray-coded binary patterns to uniquely determine fringe order, $k(x, y)$, and to unwrap the *absolute* phase values. Any subsequent unwrapping artifacts were eliminated by using the computational framework introduced in Ref. [61].

Once absolute phase maps are uniquely defined, they can be used to encode any arbitrary number of points at any location on the monitor since they are encoded in the continuous phase. This differs from the calibration method developed by Li et al. [60] where a physical calibration board, with printed circular patterns, was used as feature points. To be properly captured and identified by an in-focus camera, these circles have to be large enough for the camera to capture, so that these feature centers can be determined accurately from the camera images. Therefore, this requirement places a constraint on the total number of feature points that can fit on the board, unlike the proposed method where an arbitrary number of points can be used as feature points. For example, a point (u_0^d, v_0^d) on the monitor can be encoded as,

$$u_0^d \leftarrow \Phi_v(u_0^d, v_0^d), \tag{3.11}$$

$$v_0^d \leftarrow \Phi_h(u_0^d, v_0^d). \tag{3.12}$$

Here, Φ_v represents the vertical phase map that varies along u^d direction, and Φ_h represents the horizontal phase map that varies along v^d direction. These phase values can be further discretely represented as,

$$\Phi_v(u_0^d, v_0^d) = 2\pi u_0^d / P_v, \tag{3.13}$$

$$\Phi_h(u_0^d, v_0^d) = 2\pi v_0^d / P_h. \tag{3.14}$$

Here P_v and P_h respectively represent the vertical and horizontal number of pixels to represent 2π. And thus

$$u_0^d = \frac{\Phi_v(u_0^d, v_0^d)}{2\pi} \times P_v, \tag{3.15}$$

$$v_0^d = \frac{\Phi_h(u_0^d, v_0^d)}{2\pi} \times P_h. \tag{3.16}$$

As discussed earlier in this section, the phase can be carried along by N equally phase-shifted fringe patterns,

$$I_v^i(u,v) = 127.5[1+\cos(\Phi_v + 2i\pi/N], \tag{3.17}$$

$$I_h^i(u,v) = 127.5[1+\cos(\Phi_u + 2i\pi/N], \tag{3.18}$$

where $i = 1, 2, \cdots, N$. Once these phase-shifted fringe patterns and gray-coded binary patterns are captured by a camera, the absolute phase maps $\Phi_v(u^c, v^c) = \Phi_v(u^d, v^d)$ and $\Phi_h(u^c, v^c) = \Phi_h(u^d, v^d)$ can be computed for each camera pixel. These phase maps can be used to uniquely find the corresponding mapped points on the LCD pixel coordinates (u^d, v^d) for any point on the camera (u^c, v^c).

3.2.3 Subpixel feature point extraction

In theory, the phase can accurately carry encoded information (e.g., feature points). For example, to encode a feature point, (u, v), we can represent the same information in the phase domain, using horizontal and vertical phase, as (Φ_v, Φ_h), and if the phase is unique, the mapping from (u, v) to (Φ_v, Φ_h) is one to one. However, in practice, due to the nature of discrete fringe generation, and the sampling of the camera, unique one-to-one mappings cannot always be guaranteed. The defocusing of the lens further makes this one-to-one mapping a rarity, in practice. In general, without loss of generality, there are two scenarios: 1) the camera pixel is larger than the LCD monitor pixel; and 2) the camera pixel is smaller than the LCD pixel. The former corresponds to the case when the camera is far away from the LCD screen. For this case, some pixels on the LCD monitor may not have corresponding sampled pixels on the camera, as illustrated in Fig. 3.3(a), where the encoded pixel (u_0^c, v_0^c) is not resolved by the camera. The latter corresponds the the case when the camera is very close to the LCD screen, indicating that many camera pixels may correspond to one LCD pixel, as illustrated in Fig. 3.3(b). For either case, additional processing is required to accurately recover the feature point from the phase maps. For example, previous works have performed bilinear interpolation using pixels surrounding the fea-

ture points to establish correspondence between the two phase maps [62]. Although successful, the bilinear interpolation could introduce bias error if the phase value of one or more of these four points has large error.

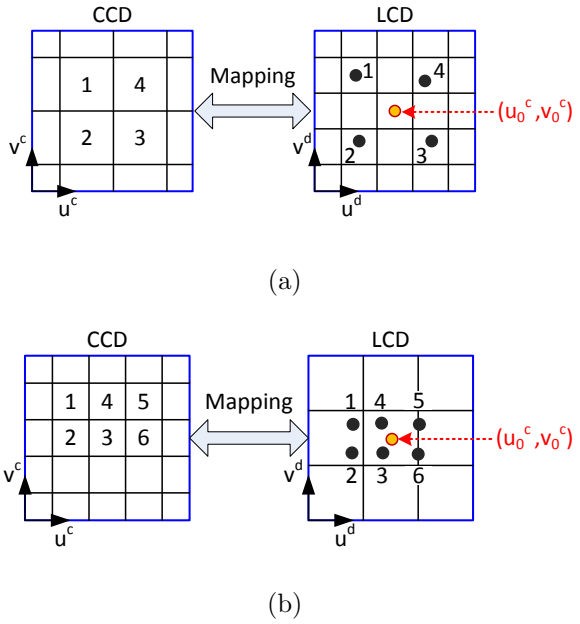


Fig. 3.3. Mapped feature point establishment through horizontal and vertical phase maps. (a) Camera pixel is larger than LCD pixel; (b) camera pixel is smaller than LCD pixel.

Instead of using the simple bilinear interpolation, in our research, we assume that any given point on the camera is locally planar, and thus the plane can be used to accurately determine any corresponding points. To determine sub-pixel accuracy feature points, we use the following steps

- *Step 1: Camera to LCD mapping creation.* This mapping is generated by the horizontal and vertical phase maps, e.g., $(\Phi_v, \Phi_h) \leftarrow (u^d, v^d)$. Since for each camera point (u^c, v^c), the horizontal and vertical phase values are unique, we can also establish the mapping $(u^c, v^c) \leftarrow (u^d, v^d)$.

- *Step 2: Local plane fitting.* For any feature point (u_0^d, v_0^d), locally find all of the camera mapped points (u_k^d, v_k^d). If we assume that those local mapped feature points are on the same plane, we will have,

$$u_k^c = a_1 u_k^d + b_1 v_k^d + c_1, \tag{3.19}$$

$$v_k^c = a_2 u_k^d + b_2 v_k^d + c_2. \tag{3.20}$$

  where a_1, b_1, c_1, a_2, b_2, and c_2 are plane coefficients. These coefficients can be determines by a least-square method using all of the local feature points.

- *Step 3: Subpixel feature point extraction.* Once the local plane functions are estimated, the coordinates for any given feature points (u_0^d, v_0^d) can be computed as

$$u_0^c = a_1 u_0^d + b_1 v_0^d + c_1, \tag{3.21}$$

$$v_0^c = a_2 u_0^d + b_2 v_0^d + c_2. \tag{3.22}$$

3.3 Simulation

Li et al. [60] thoroughly proved that phase information is preserved regardless of projector lens defocusing. Briefly put, if an optical imaging system is defocused, a point on the object no longer converges to a point on the image plane, but rather an blurred circular disk. However, considering the infinite light ray of the optical system, the center of a camera pixel, regardless of the amount of defocusing, corresponds to the peak intensity value of the circular disk. In practice, it is difficult to find the peak intensity value from an image due to surface reflectivity variation, however. In contrast, it is much easier to find the peak value through phase. Using phase, the center point still corresponds to the peak value of the circular disk regardless of the amount of defocusing.

We carried out some simulation to confirm that camera defocusing does not change the resultant phase of the phase-shifted fringe patterns. From diffraction theory, one

may know that lens defocusing can be simulated by convolving the image with a Gaussian function

$$G(x,y) = \frac{1}{2\pi\sigma^2} \exp\left[-\frac{(x-\mu_x)^2 + (y-\mu_y)}{2\sigma^2}\right]. \tag{3.23}$$

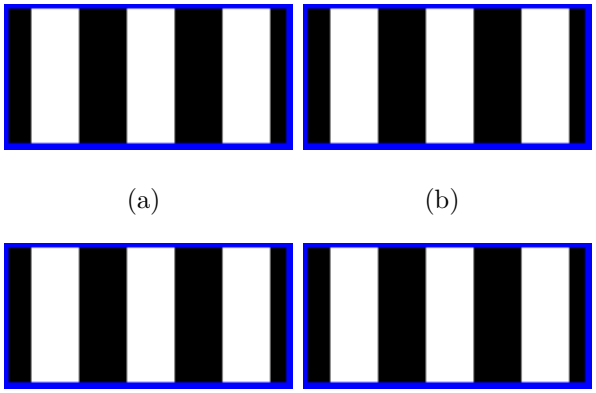
Here, (μ_x, μ_y) is the position of the focal center, and σ controls the width of the blurred image area.

Figure 3.4 shows one of the phase-shifted fringe patterns with different amounts of defocusing (e.g., different Gaussian filter sizes). Squared binary patterns, in lieu of sinusoidal patterns, were adopted in this research because: 1) squared binary patterns provide the highest possible contrast when the patterns are substantially defocused; 2) binary patterns are not affected by the nonlinear gamma of the LCD monitor; and 3) Esktrand and Zhang [63] has demonstrated that accurate phase can be recovered even if the patterns are ideally squared binary. In this simulation, the squared binary patterns had a period of 42 pixels and 21 equally phase-shifted patterns were used to compute the phase.

The absolute differences between the ideal phase without defocusing and the phase with different amounts of defocusing are shown in Fig. 3.5. The root-mean-square (rms) errors are all very small, demonstrating that lens defocusing indeed does not alter the phase carried by the fringe patterns. Therefore, it is theoretically possible to encode the feature point information into phase to avoid the problems caused by camera lens defocusing.

3.4 Experiment

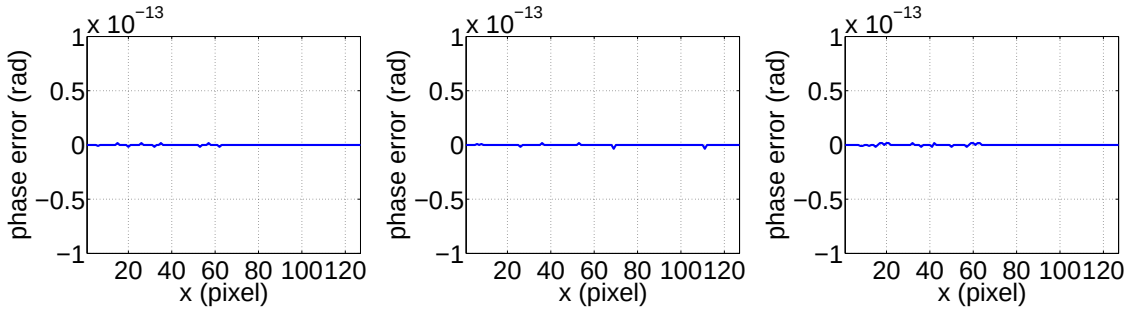
To verify the performance of the proposed method, we developed a camera calibration system that includes an LCD monitor (Model: HP EliteDisplay E241i 24-inch IPS LED Backlit Monitor) and a charge-coupled device (CCD) camera (Model: Jai Pulinx TM-6840CL) with a 12 mm focal length lens (Model: Computar M1214-MP2). The LCD monitor has a resolution of 1920 × 1200 and a pixel pitch of 0.270 mm. The camera resolution is 640 × 480 with a pixel size of $7.4\mu m \times 7.4\mu m$. The lens



(a) (b)

(c) (d)

Fig. 3.4. Example squared binary fringe patterns when the structured patterns are defocused at different degrees. (a) A focused binary pattern; (b) the pattern after applying a Gaussian filter with a size of 9×9 pixels; (c) the pattern after applying a Gaussian filter with a size of 17×17 pixels; and (d) the pattern after applying a Gaussian filter with a size of 33×33 pixels.



(a) (b) (c)

Fig. 3.5. Phase difference between blurred structured patterns and focused (without blur) patterns. (a) Gaussian filter size of 9×9 pixels (phase rms 3.9×10^{-16} rad); (b) Gaussian filter size of 17×17 pixels (phase rms 3.3×10^{-16} rad); (c) Gaussian filter size of 33×33 pixels (phase rms 4.6×10^{-16} rad).

is a 2/3-inch, 12 mm lens with an aperture of F/1.4-F/16C. The range of focus is approximately 150 mm to infinity.

For all of the following experiments, the camera focus remained constant and untouched to maintain the camera intrinsic parameters; differing amounts of defocus was realized by changing the distance, D, between the monitor and the camera. In this research, we tested four different amounts of defocusing: from the camera being focused to the camera being substantially defocused (i.e., we used four different distances). The active areas of the monitors used for calibration for each distance are summarized in Tab. 3.1. This table shows that the calibration target size needs to be proportionally scaled up when the distance between the camera the object increases, as illustrated in Fig. 3.1. Therefore, as discussed in Sec. 3.1, the conventional method of calibrating a camera lens is to use a larger calibration target when the sensing area is larger.

Table 3.1.
LCD monitor pixels used for camera lens calibration.

	Active monitor range (pixels)
$D_1 = 950$ mm	1260×960
$D_2 = 540$ mm	720×540
$D_3 = 250$ mm	330×250
$D_4 = 125$ mm	165×125

Our research is to prove that it is not necessary to increase target size for camera calibration, but rather one can place the calibration target closer to the camera. Obviously, if the focal plane of the camera does not change, the camera will be out of focus; making the image blurry when the calibration target is placed away from its camera focal plane. Figure 3.6 shows some example camera images for those four different distances. The same square binary pattern was displayed on the LCD monitor, but the structured pattern was blurred if the camera is not focused (e.g.,

the image shown in Fig. 3.6(d)); note that when the camera is away for the monitor, the patterns appear more dense.

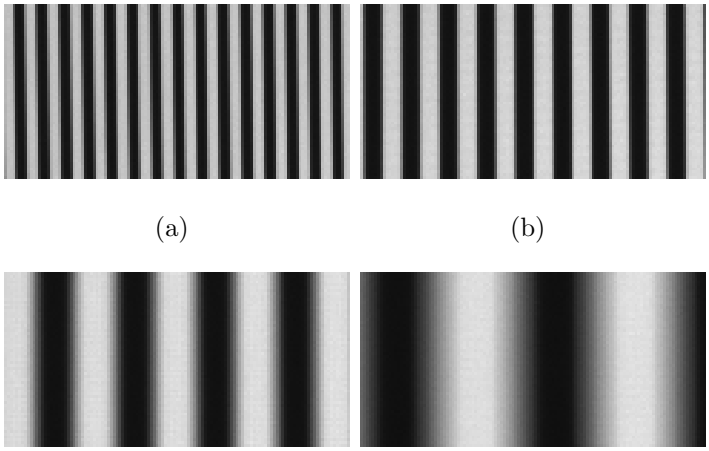


Fig. 3.6. Example images of a squared binary pattern when the camera is placed at different distance without changing its focus. (a) $D_1 = 950$ mm; (b) $D_2 = 540$ mm; (c) $D_3 = 250$ mm; (d) $D_4 = 125$ mm.

As discussed previously, since the feature points are carried along by phase values, and not intensity values, the appearance of the structured patterns should not alter the feature points if the phase itself does not change. The horizontal and vertical phase maps were encoded with 12 equally phase-shifted squared binary patterns with a fringe pitch of 24 pixels (i.e. $N = 12$ and $P_v = P_h = 24$ pixels for Eqs (3.15)-(3.18)). A temporal phase unwrapping algorithm was used to unwrap the phase pixel-by-pixel with unwrapping artifacts being removed using the computational framework discussed in Ref. [61].

Since the feature points are encoded in phase maps, they can be determined once the horizontal and vertical phase maps are computed regardless of the amount of camera defocusing by using the proposed method discussed in Subsec. 3.2.3. Figure 3.7 shows some example detected feature points at different amounts of camera

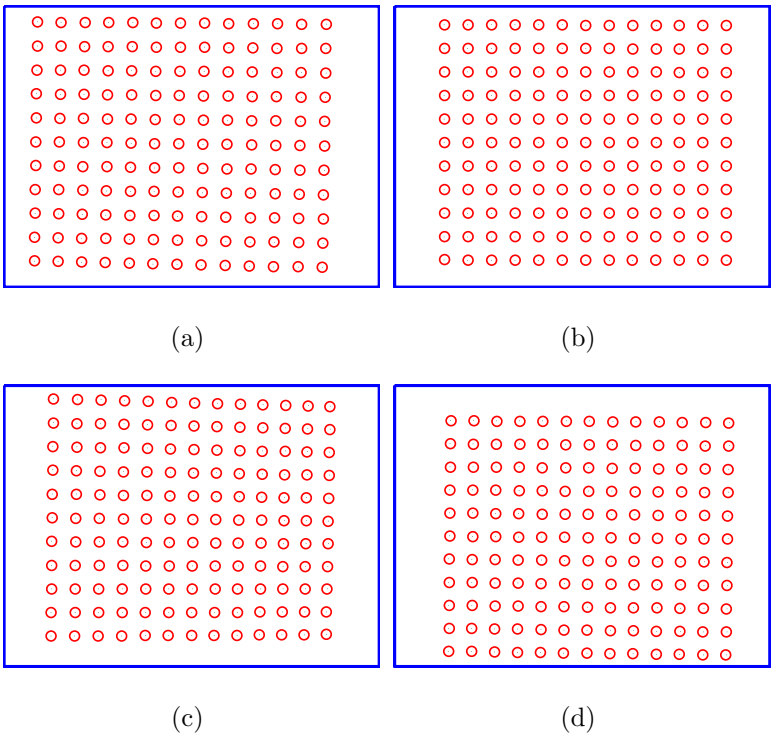


Fig. 3.7. Example detected feature points for one of the poses when the camera is at different amounts of defocusing (i.e., different distances from the LCD monitor). (a) $D_1 = 950$ mm; (b) $D_2 = 540$ mm; (c) $D_3 = 250$ mm; (d) $D_4 = 125$ mm.

defocusing. It at least visually appears that all of the feature points are properly recovered.

Once the feature points are detected, the camera intrinsic parameters can be estimated using the standard camera calibration approach. In this research, we used the OpenCV camera calibration software package to estimate the intrinsic parameters of the camera lens. Table 3.2 summarizes the intrinsic parameters estimated from four different levels of defocusing. For each amount of defocusing, we captured 15 different target poses and used 143 feature points for each calibration plane. These experimental results clearly demonstrate that the equivalent focal lengths estimated from different amounts of defocusing are extremely close to each other: less than 0.2%

difference. The principle points estimated from different amounts of defocusing are also very close to each other. One may notice the largest principle difference occurs when the camera is substantially defocused, which is still only approximately 1%.

Table 3.2.
Intrinsic parameters estimated when the camera is under different amounts of defocusing.

	$f_u(mm)$	$f_v(mm)$	$u_0(mm)$	$v_0(mm)$
$D_1 = 950$ mm	12.17	12.17	2.34	1.88
$D_2 = 540$ mm	12.16	12.16	2.34	1.88
$D_3 = 250$ mm	12.14	12.14	2.34	1.87
$D_4 = 125$ mm	12.16	12.16	2.32	1.89

For our camera lens, we found that keeping k_1 and k_2 for nonlinear distortion is sufficient. Table 3.3 presents the estimated distortion coefficients, and they are all very small. The reprojection errors are respectively 0.033 pixels, 0.022 pixels, 0.029 pixels, and 0.058 pixels for $D_1 = 950$ mm, $D_2 = 540$ mm, $D_3 = 250$ mm, and $D_4 = 125$ mm. As can been seen, all of these reprojection errors are very small.

Table 3.3.
Distortion coefficients estimated when the camera is under different amounts of defocusing.

	k_1	k_2
$D_1 = 950$ mm	-0.058	0.288
$D_2 = 540$ mm	-0.071	0.345
$D_3 = 250$ mm	-0.088	0.270
$D_4 = 125$ mm	-0.118	0.133

These experiments confirm that the calibration data are very close to each other when the camera is under different amounts of defocusing. One may notice that there are some slight differences between the different amounts of defocusing, which might be a result of the disparity between calibration poses used at each level of defocusing and/or the different number of pixels used to display the encoded fringe patterns.

3.5 Large range structured light system calibration

Structured light 3D scanners in their most basic form are systems comprised of a single camera and a single projector. Combined with digital fringe projection (DFP) techniques, they are highly advantageous due to their speed, resolution, and accuracy. Using such high-quality systems within applications where accuracy may be of importance (e.g., telemedicine) could be highly beneficial; however, such systems may be most effective at a close to medium distance from the device.

One factor that affects the accurate measurement range of a structured light scanner is the accuracy of its calibration. Typically, a calibration target is used to calibrate the respective intrinsic parameters of the camera and projector, as well as the extrinsic relationship between the two. Similar to the camera calibration method proposed above, the size of a calibration target used to calibrate a structured light system is usually within the same order of magnitude as the system's desired field-of-view (FOV). If one wishes to extend the accurate calibration range of a structured light scanner, one then has to scale the size of the calibration target used; however, accurately fabricating such large calibration targets is both difficult and expensive. Even if fabricated, they may be difficult to manipulate within the desired FOV to perform an unbiased calibration.

As in the camera calibration, we thus desire to perform this calibration at a close range, even though the system's desired working range may be at a longer distance. Given that both the camera and projector may be out of focus when calibrating at this close range, we have integrated the above out-of-focus camera calibration procedure

into a procedure for calibrating a large-range structured light system. The proposed camera calibration method can be used to calibrate the intrinsic parameters of an out-of-focus camera, and the method by Li et al. [60] is used to calibrate the intrinsic parameters out-of-focus projector. A low-quality Microsoft Kinect v2 is then used to assist in calculating the extrinsic relationship between the camera and projector.

Using this calibration procedure, a large range 3D structured light measurement system was developed with a FOV of $(1120 \times 1900 \times 1000)^3$, and experiments showed mean measurement accuracies as high as 0.07 mm when measuring a sphere with a 304.8 mm diameter. This accuracy is quite high, especially when compared to the mean 0.80 mm measurement accuracy achieved by the Kinect v2 which is popularly used for long range applications. This work was published in the Journal of *Applied Optics* where more details can be found [64].

3.6 Summary

This chapter has presented an out-of-focus camera calibration approach by encoding the calibration feature points into phase which are further carried along by phase-shifted fringe patterns displayed by a LCD monitor. Experiments demonstrated that the proposed method can accurately calibrate camera intrinsic parameters (e.g., focal length, principle points) regardless of the amounts of defocusing: the focal length difference is approximately 0.2% when the camera is focused versus substantially defocused. The proposed camera calibration method could significantly simplify the calibration of large-range vision systems, such as the large-range structured light system.

4. THREE-DIMENSIONAL RANGE GEOMETRY COMPRESSION VIA PHASE ENCODING

The previous chapter introduced our novel out-of-focus camera calibration procedure, and discussed how it could be used to extend the measurement range of a structured light 3D scanning system. In the context of achieving realistic 3D video communications, however, even if the acquired 3D data is of ideal resolution, accuracy, and FOV, one challenge that remains is transmitting it from one user to another efficiently, without losing significant data quality. This chapter thus introduces a novel method of encoding 3D range data, along with its respective texture information, within the color channels of a regular color 2D image. The proposed method is highly resilient to lossy compression artifacts, enabling extremely high compression ratios while maintaining very low reconstruction error percentages. The majority of content in this chapter was originally published in *Applied Optics* [65] (also listed as journal article [J7] in "LIST OF PUBLICATIONS").

4.1 Introduction

Three-dimensional (3D) scanning technologies have the ability to capture high-quality data at real-time speeds [66]. Such abilities have led to an increased adoption of these technologies within many various industries such as medicine, communication, entertainment, and manufacturing. Given the large amounts of data generated by 3D scanning technologies, the real-time storage and transmission of such data becomes important.

One way to represent 3D data is with a mesh format. A mesh is described by a set of *vertices* (3D coordinates) and a set of *edges* which specify the structure of the mesh (i.e., how the coordinates should be *connected* to one another). Some additional

attributes of the mesh may also be stored: such as a normal map, vertex colors, or a texture image along with texture image coordinates. Standard mesh file formats (e.g., OBJ, STL, PLY) are based on simple listings of the information required to reconstruct a mesh with its attributes. In the past several decades, much work has been done to try and represent this information as efficiently as possible.

Researchers have sought new ways to efficiently encode a mesh's connectivity information in order to reduce the amount of information needed overall to represent the mesh. Connectivity information can be efficiently encoded using intelligent methods of traversing the vertices or structures within the mesh. A well-designed encoding method reduces redundancy within the connectivity information, thus reducing overall file sizes, and many methods have been proposed (e.g., triangle strip [19–21], spanning tree [22], valence encoding [23, 24], triangle traversal [25, 26]). Once connectivity information has been encoded, the actual positions of the vertices are then encoded. This is typically done by following a three-step procedure of quantization, prediction, and entropy encoding [27, 28].

The above methods are connectivity-driven, meaning that the encoding of geometry information follows the order of the connectivity encoder. Given that the data size for a 3D mesh is generally more impacted by geometry (i.e., coordinate) data [27, 28], there have also been geometry-driven methods developed for the compression of 3D meshes. Such methods let the encoding be driven by what best encodes the coordinate positions, even if it does not result in an optimal encoding of the connectivity information. For example, Kronrod and Gotsman [29] proposed a method which optimizes the predictions between the positions of adjacent vertices. The connectivity information would then be encoded by following the optimized predictions. It was found that this optimization could provide much more compact meshes overall while paying only a small penalty for the non-optimal connectivity encoding [29]. Mesh compression problems become more simple if precise restoration of a mesh's connectivity is not required, or if the data has an underlying structure that automatically carries the connectivity information. For example, a regular grid or pixel structure

is often assumed with range data captured by a camera. In such cases, the encoding methods can primarily focus on how to precisely and efficiently encode 3D data itself.

In the field of phase-shifting interferometry digital holography (PSIDH), 3D hologram information is encoded within a 2D complex wavefront. Multiple phase-shifted interference, or *fringe*, patterns are captured by a camera and these patterns can recover both amplitude and phase information. Deformations of the phase from a reference plane are related to deviations of the object surface from the reference plane being captured. Typically, the information within the fringe patterns are used to derive a complex wavefront, or Fresnel field, which is used to reconstruct the captured object. As these wavefronts consist of complex, floating point values, methods for compressing the data are desired.

Since digital holograms have an inherent grid structure (the wavefront is computed from data digitally recorded by a camera), the 3D geometry compression problem is simplified to a 2D compression problem. Furthermore, instead of compressing the wavefront and its complex values, Darakis and Soraghan [32] proposed a digital hologram compression method which applies JPEG and JPEG 2000 compression directly to the camera captured interference patterns, from which a complex wavefront can later be computed. This method is flexible due to the ability to define and control the compression rates (i.e., the JPEG quality level in use). However, to achieve higher compression ratios, lower JPEG qualities are used which causes considerable error on the reconstructed wavefront. Further, the data size is proportional to the number of phase-shifted patterns captured by the camera. If this number is increased, compressing the wavefront and its complex values may be more efficient.

Darakis and Soraghan [33] proposed a method for compressing a complex wavefront—at the object's reconstruction plane—which first quantizes the complex data and then losslessly encodes it using the Burrows-Wheeler transform [34]. This method outperforms the method directly compressing the interference patterns using JPEG that the same team proposed earlier [32]. It achieved reasonably good compression ratios (e.g., approximately 26:1 for a normalized root-mean-square error of approximately

0.1), and retained the hologram's natural capability of being able to be reconstructed at different depths and perspectives. More thorough reviews of state-of-the-art compression methods using various digital holography approaches are given by Alfalou and Brosseau [67] and Dufaux et al. [68].

The physical properties of digital holography systems (e.g., lighting conditions, surface texture, speckle noise) could greatly affect the efficiency of hologram compression methods. To alleviate such potential problem, a *virtual* digital holography system can be used to create computer-generated holograms (CGH). These are generated by numerically simulating how light reflects and propagates off of a virtual 3D object. CGH methods are advantageous as they can both represent arbitrary 3D objects and are computed within a completely ideal environment. Recently, methods have been proposed for compressing CGHs using JPEG [35] or even HEVC [36]. Although these compression methods are quite effective, generating the CGHs themselves is both a computationally complex and memory expensive process [37]. Graphic processing units (GPU) can be used to effectively reduce the time of computing CGHs [38–40], yet such a method is still limited by the amount of on-board memory for high-resolution hologram generation. Moreover, since the viewing angle of the reconstructed image is proportional to the CGH size [40], the resulting CGH may have a very large spatial resolution compared to the number of points actually encoded. In general, all the compression approaches based on digital holograms suffer from the noise caused by speckle. The presence of speckle noise makes it difficult to fully leverage 2D lossy image compression methods, hindering the ability to achieve very high compression ratios while also preserving data quality after compression.

Compared to holography-based 3D geometry compression methods, digital fringe projection (DFP)-based 3D range data compression methods typically have the advantage of 1) one-to-one correspondence between a pixel on an image and one encoded 3D geometry point; 2) the elimination of speckle noise related problems; and 3) the ability to achieve much higher compression ratios with standard 2D image compression techniques (e.g., a magnitude higher for high-quality compression). Similar to

the concept of using a virtual digital holography system to calculate CGHs, a virtual DFP system can be used to precisely and quickly encode 3D coordinates within the three channels (RGB) of a regular 2D image using phase-shifting techniques. Once 3D range geometry is encoded into a 2D image, it can then be further compressed using well-established image compression techniques, such as PNG or JPEG, and saved to disk or transmitted over a network.

Researchers have proposed different approaches to encode 3D geometry into a 2D image using the concept of a virtual DFP system along with phase-shifting principles [41, 46, 69]. These methods use all three color channels of the output RGB image to encode 3D data. Typically two of the three color channels will be used to represent the 3D data. The third color channel is used to store important *fringe order* information which is needed for the proper recovery of the phase-shifted data within the first two channels. Although these methods are successful, using all three color channels of the output image limits the ability to save any additional information with the 3D data (e.g., a texture image). Given this, some methods have focused on encoding 3D geometry in such a way that it only uses two of the three color channels of the output 2D image.

Hou et al. [45] proposed a two channel method which was able to represent 3D geometry with one single channel of the output image, still using a second channel to store the fringe order information for decoding. Using one channel instead of two to represent 3D geometry information then leaves one channel free, either to be left empty or to store additional attributes of the data. Although efficient in this regard, this method uses only a single 8-bit color channel to represent 3D geometry limiting the precision of the encoding. Further, the data this method encodes to represent 3D geometry contains very sharp discontinuities which results in rapid intensity changes between pixels of the output image. This limits the method's potential extension to using lossy JPEG compression to further reduce file sizes, as sharp intensity changes can cause compression artifacts within the encoded data.

Wang et al. [44] also proposed a method which was able to encode 3D geometry using only two color channels of an RGB image, leaving one channel open for additional information. This method also uses one channel to represent 3D geometry information and another to store the fringe order information needed for decoding. While successful, the method has the same drawback that it only uses a single color channel to represent 3D geometry, limiting the precision of the encoding. Further, it requires a post-processing error compensation framework to alleviate decoding errors.

Karpinsky et al. [70] proposed a method which encoded 3D geometry into three color channels and then performed dithering on each one. Using this method, each 8-bit channel could be represented with a single bit per pixel, allowing all three color channels to be represented with only three bits per pixel. This method is quite advantageous in terms of its small file sizes and large amounts of remaining space within the output image for the storage of additional information (such as a texture image). In fact, the texture image itself can be dithered along with the geometry information in order to reduce data sizes even further. The main drawback of this method was that it required the usage of a lossless image compression technique (i.e., PNG) when storing the dithered channels. Ideally, lossy image compression techniques could be used to further decrease file sizes, however, if a lossy method (i.e., JPEG) was used to store the dithered channels, the resulting file sizes were larger than PNG. Since almost all of the widely used video codecs (e.g., H.264) employ some sort of lossy image compression, this would limit this encoding method's extension to 3D video applications.

One trait that's common in the methods which use phase-shifting concepts to represent 3D data within a 2D image is the need to encode the fringe order information. This is because the fringe order value for each 3D coordinate (or its associated 2D pixel) needs to be known in order to perform proper decoding of the coordinate. If there were another means to derive the fringe order information (instead of encoding it directly within the output image), up to an entire channel could be saved or used to store additional information.

This chapter proposes a novel method for 3D range geometry compression which utilizes the geometric constraints of the 3D capture system itself to derive this fringe order information, necessary for proper data decoding, in an on-demand fashion using the system's calibration parameters. The result of this is that fringe order information no longer must be stored along with the encoded 3D data within the output 2D image. This freedom allows our method the ability to precisely represent floating point 3D range geometry within two entire color channels while keeping the third color channel open for additional data storage. Further, the encoding within the two color channels is continuous in nature (i.e., no sharp intensity changes) which allows the proposed method to achieve extremely large compression ratios (i.e., smaller file sizes) via lossy JPEG encoding while maintaining very high reconstruction accuracies. For example, compression ratios of 3038:1 were achieved versus the STL format, with a root-mean-square (RMS) error of 0.47%, when the output image was compressed with JPEG 80%.

The proposed 3D range geometry encoding method can efficiently archive or transmit 3D range geometry data, which could be valuable for applications such as entertainment, security, and tele-communications. Further, given the method's ability to encode 3D range data within two color channels, a texture image can be stored in the third channel. This may be beneficial, for example, to the area of telemedicine: remote physicians could leverage both decoded 3D range geometry and 2D texture image to perform simultaneous physical measurements and visual assessments to make sound medical decisions.

Section 4.2 will describe the novel 3D range geometry encoding and decoding methods, specifically in how the geometric constraints of the capture system can be used to help decode geometry information stored within a 2D image. Section 4.3 will present various experimental results of the proposed encoding method, and Sec. 4.4 will summarize the chapter.

4.2 Principle

4.2.1 Phase encoding for 3D range geometry compression

A generic structured light scanner consists of one camera and one projector. The digital fringe projection (DFP) technique is one of the structured light methods which uses a projector to project phase shifted, sinusoidal fringe images onto a 3D scene. The camera will then capture the distorted fringe images projected upon the scene and can use these to compute distorted phase information. This phase information can then be used pixel-by-pixel to recover 3D coordinates if the DFP system is properly calibrated [60].

The concepts of phase shifting can also used to encode 3D geometry into a 2D RGB image. However, as discussed in Sec. 3.1, the state-of-the-art methods require one of the three output color channels to store the fringe order information needed to properly decode the phase shifted data. This chapter presents a novel method for encoding that can recover the geometry without needing to store fringe order information. Given this, the proposed method can use two data channels to precisely encode data while having one channel free to store additional data.

The proposed method directly encodes distorted phase information as captured by a DFP system, Φ, into two color channels (e.g., red and green) of the output 2D image:

$$I_r(i,j) = 0.5 + 0.5 \times \sin\left[\Phi(i,j)/SF\right], \tag{4.1}$$

$$I_g(i,j) = 0.5 + 0.5 \times \cos\left[\Phi(i,j)/SF\right] \tag{4.2}$$

where (i,j) are image pixel indices and where SF is a scaling factor. This encoding is advantageous as it retains the precision of the phase map while remaining very straightforward to implement. Once the phase has been encoded into the 2D image, it can be further compressed using conventional methods such as PNG or JPEG.

4.2.2 Phase decoding and unwrapping using geometric constraints

To recover phase back from the 2D image, ϕ is computed from the encoded data stored in the two channels:

$$\phi(i,j) = \tan^{-1}\left[\frac{I_r(i,j) - 0.5}{I_g(i,j) - 0.5}\right]. \tag{4.3}$$

This recovered phase ϕ is bounded within the range $[-\pi, \pi)$. The original unwrapped phase, Φ, can be recovered if the 2π discontinuities within ϕ can be identified, ordered, and corrected. It is this fringe order information, denoted by K, which existing encoding methods carry along within an additional color channel. To save data and to avoid using a color channel to carry along the fringe order information (either directly or within some other encoding), the proposed method uses the geometric constraints of the DFP system to generate an artificial phase map, Φ_{min}. Then, for each pixel, Φ_{min} can be referenced to determine the proper K value for that pixel. The following will describe the mathematical models governing the system and how they are used, as proposed by An et al. [71], to generate Φ_{min}.

The camera and projector within a structured light system can each be mathematically described using a pinhole model. Using this model, real world coordinates, (x^w, y^w, z^w), can be projected onto the 2D plane, at the coordinate (u, v), using the equation

$$s\begin{bmatrix} u & v & 1 \end{bmatrix}^t = \mathbf{P}\begin{bmatrix} x^w & y^w & z^w & 1 \end{bmatrix}^t, \tag{4.4}$$

where s is a scaling factor and $\mathbf{P}$ is the projection matrix. This matrix can be described as

$$\mathbf{P} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \tag{4.5}$$

where f_u and f_v are the focal lengths along u and v directions, respectively; γ is the skew factor of the two axes; r_{ij} and t_i are the rotation and translation parameters;

and (u_0, v_0) is the principle point. This projection matrix is often simplified into a single 3×4 matrix,

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}. \tag{4.6}$$

If the camera and projector of a DFP structured light system are properly calibrated, we know their respective projection matrices, $\mathbf{P^c}$ and $\mathbf{P^p}$. These matrices can be used to obtain two sets of equations, one for the camera (denoted superscript c) and one of the projector (denoted superscript p) describing the DFP system:

$$s^c \begin{bmatrix} u^c & v^c & 1 \end{bmatrix}^t = \mathbf{P}^c \begin{bmatrix} x^w & y^w & z^w & 1 \end{bmatrix}^t, \tag{4.7}$$

$$s^p \begin{bmatrix} u^p & v^p & 1 \end{bmatrix}^t = \mathbf{P}^p \begin{bmatrix} x^w & y^w & z^w & 1 \end{bmatrix}^t. \tag{4.8}$$

Equations (4.7) and (4.8) provide six equations yet there are seven unknowns: s^c, s^p, x^w, y^w, z^w, u^p, and v^p. To solve for the unknowns, one more equation is needed; typically, the linear relationship between some known absolute phase value, Φ, and a projector line is used to resolve this by providing an additional equation to solve for u^p or v^p (depending on the direction of Φ). At this point, the unknowns can be solved for, and a 3D coordinate for each camera pixel can be derived.

Similarly, consider if the absolute phase value is unknown yet the depth value z^w is known for a pixel. For a given z^w then, an artificial phase value can be determined. Further, if $z^w = z_{min}$, the artificial phase map is a minimum phase map, denoted Φ_{min}. This map can formally be defined by a function taking inputs z_{min}, the minimum z value; T, the fringe width on the projector used to capture the original data; and the respective projection matrices for the camera and projector, $\mathbf{P^c}$ and $\mathbf{P^p}$. Based on the fringe width T used by the DFP system, the minimum phase map may have a limited working depth range [71]. To ensure that Φ_{min} can be used to properly unwrap the decoded ϕ, T_s is used to derive Φ_{min}, and it is defined as $T \times SF$, thus,

$$\Phi_{min}(u^c, v^c) = f\left(z_{min}, T_s, \mathbf{P^c}, \mathbf{P^p}\right), \tag{4.9}$$

is of a function of $z_{min}, T_s, \mathbf{P^c}$ and $\mathbf{P^p}$.

To actually determine Φ_{min}, x^w and y^w are first computed for each camera pixel (u^c, v^c) via

$$\begin{bmatrix} x^w & y^w \end{bmatrix}^t = A^{-1}b, \tag{4.10}$$

where

$$A = \begin{bmatrix} p^c_{31}u^c - p^c_{11} & p^c_{32}u^c - p^c_{12} \\ p^c_{31}v^c - p^c_{21} & p^c_{32}v^c - p^c_{22} \end{bmatrix}, \tag{4.11}$$

$$b = \begin{bmatrix} p^c_{14} - p^c_{34}u^c - (p^c_{33}u^c - p^c_{13})\, z_{min} \\ p^c_{24} - p^c_{34}v^c - (p^c_{33}v^c - p^c_{23})\, z_{min} \end{bmatrix}. \tag{4.12}$$

Knowing x^w and y^w, (u^p, v^p) can be found for each camera pixel, similar to Eq. (4.8):

$$s^p \begin{bmatrix} u^p & v^p & 1 \end{bmatrix}^t = \mathbf{P}^p \begin{bmatrix} x^w & y^w & z_{min} & 1 \end{bmatrix}^t. \tag{4.13}$$

Finally, the artificial phase value, Φ_{min} can be determined via

$$\Phi_{min}(u^c, v^c) = u^p \times 2\pi/T_s. \tag{4.14}$$

This specific equation will provide phase assuming the fringe patterns are projected along the v^p direction; to obtain phase along the other direction, the u^p and v^p values can simply be swapped.

Once the artificial phase map has been derived, it can be used to determine the fringe order information, K, as

$$K(i,j) = Ceil\left[\frac{\Phi_{min}(i,j) - \phi(i,j)}{2\pi}\right]. \tag{4.15}$$

The fringe order information is then used to unwrap ϕ in order to recover the originally encoded phase information, Φ, via

$$\Phi(i,j) = [\phi(i,j) + 2\pi \times K(i,j)] \times SF. \tag{4.16}$$

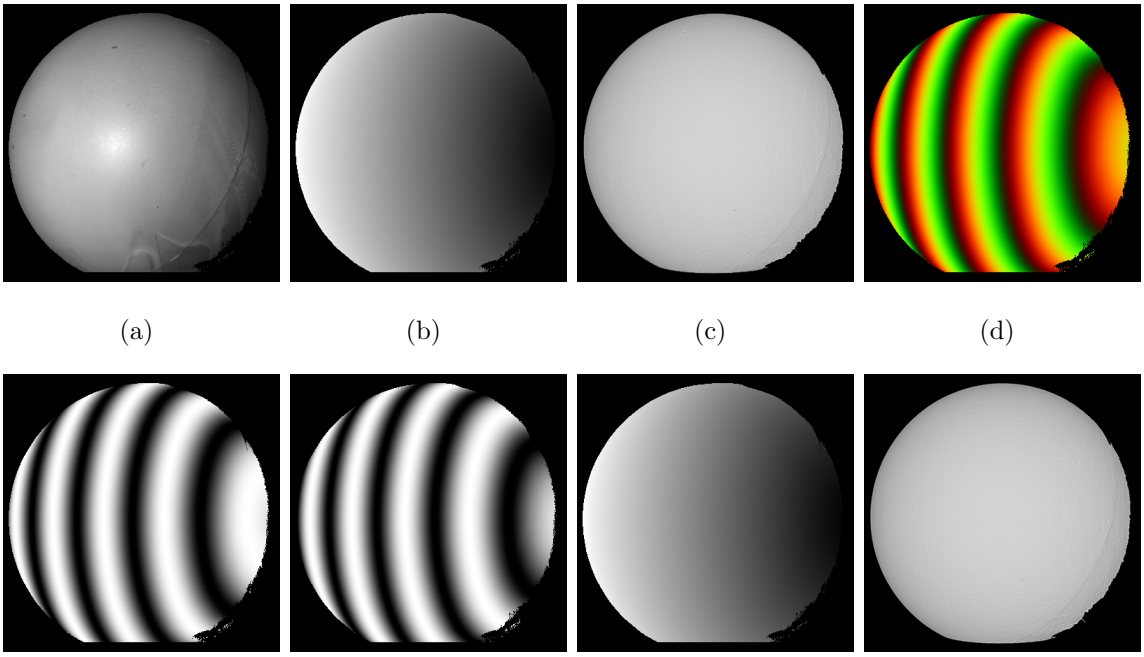
Now that the originally encoded phase, Φ, has been decoded and recovered, (x^w, y^w, z^w) coordinates can be reconstructed with Eq. (4.7) and Eq. (4.8) as described above.

4.3 Experiments

To test the proposed method, several different objects were captured with a DFP system. Comparisons made were between the 3D geometry reconstructed from the original unwrapped phase versus the 3D geometry reconstructed from the decoded, recovered unwrapped phase. The hardware system included a digital light processing (DLP) projector (Texas Instruments LightCrafter 4500) and a camera (PointGrey Flea3 FL3-U3-13Y3M-C) with an 8-mm lens (Computar M0814-MP2). The resolutions of the camera and projector were 480×640 and 912×1140, respectively. For all mentioned experiments, the fringe width used was $T = 36$ pixels. The system was calibrated following the method proposed by Li et al. [60], and only 553 bytes were required to store the resulting calibration parameters.

First, a matte white spherical object with a 4 inch (101.60 mm) diameter was captured by a DFP system and had its phase encoded into a 480×640 lossless PNG image using the proposed method. From this 2D image, the phase was decoded and used to reconstructed 3D coordinates. In this first experiment, no additional texture information was stored in the output 2D image: the phase was encoded into the red and green channels and the blue channel remained empty. Figure 4.1 illustrates this entire process: Fig. 4.1(a) shows a texture image of the sphere, Fig. 4.1(b) shows the original absolute phase, and Fig. 4.1(c) shows the sphere's 3D geometry. The phase from Fig. 4.1(b) was then encoded into a PNG image (cropped for visualization), shown in Fig. 4.1(d). Figure 4.1(e) and 4.1(f), respectively, show the red and green color channels of the PNG image which contain encoded phase information. These channels are then decoded to recover the absolute phase data, displayed in Fig. 4.1(g), which is used to recover the 3D sphere, shown in Fig. 4.1(h).

Figure 4.2 shows the reconstructed results when the output image shown in Fig. 4.1(d) was stored with different lossy image qualities (JPEG 100%, 80%, 60%, and 20%) using Matlab 2014b. One may notice that the reconstructed 3D geometry quality is fairly high if the JPEG 100% was used, as shown in Fig. 4.2(e); and the



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 4.1. Experimental results of capturing, encoding, and decoding a 4 inch (101.60 mm) diameter sphere. (a) A 2D texture image of the sphere; (b) original absolute phase of the sphere; (c) original 3D geometry reconstructed from (b); (d) encoded phase stored in a lossless PNG image via the proposed method, cropped for visualization from its original 480×640 resolution; (e) the red channel of (d); (f) the green channel of (d); (g) the decoded absolute phase from (d); (h) recovered 3D geometry reconstructed from the decoded (g).

associated the phase RMS error is small 0.17 mm or 0.35%. Even if JPG 20% was used the quality of the reconstructed 3D geometry is still reasonably good, and the error is still pretty small (0.85%).

It is important to note that these results were obtained without a kernel-based, post-processing filter or error compensation framework. The only post-processing performed was a simple threshold to remove significant boundary outliers.

The original 3D capture of the 4 inch (101.60 mm) diameter sphere required 65.0 MB, 9.0 MB, and 8.4 MB to store in the common mesh formats STL, OBJ, and PLY,

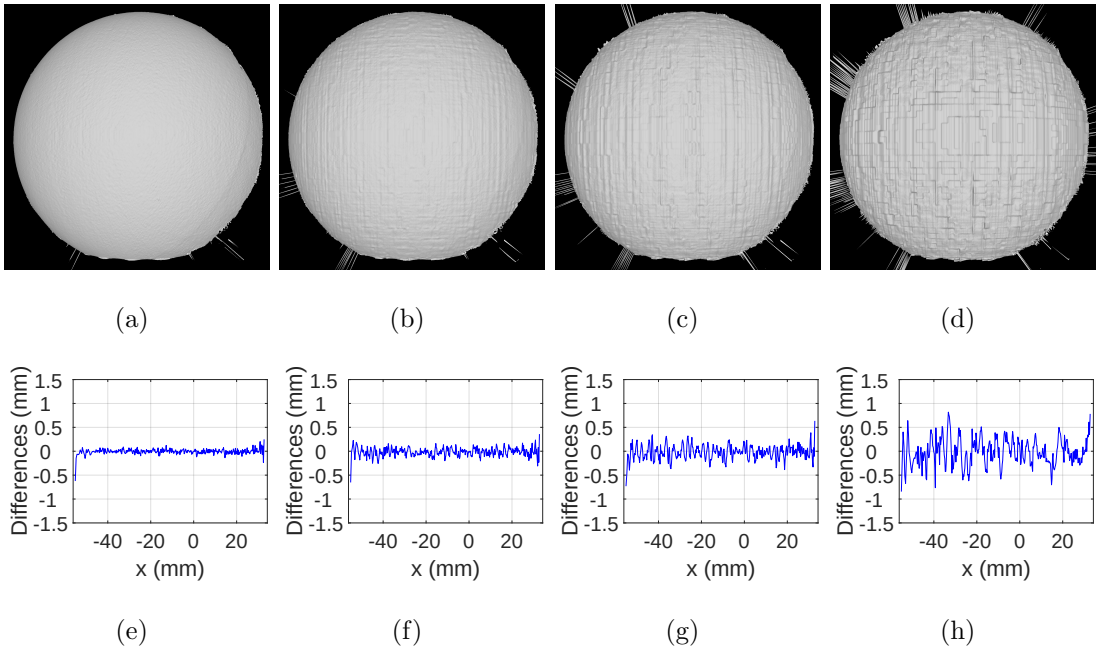


Fig. 4.2. Results of the sphere's phase being encoded into two 2D color channels and saved with different JPEG qualities (using Matlab 2014b). (a)-(d) 3D reconstructed results using the decoded phase from JPEG qualities 100%, 80%, 60%, and 20%, respectively; (e)-(h) Difference in z^w between the original sphere and the recovered sphere for a cross section. RMS errors for (e)-(h) are 0.17 mm (0.35%), 0.23 mm (0.47%), 0.31 mm (0.61%), and 0.43 mm (0.85%), respectively, after removing boundary outliers with a simple threshold.

respectively, in their ASCII formats. When storing the encoded sphere into a PNG image, the proposed method was capable of approximately a 688:1 compression ratio, with an RMS error of 0.02 mm (0.033%), versus the original geometry stored in the STL format. To obtain higher compression ratios, lossy JPEG was used to store the output image. For example, when JPEG 80% was used, a 3038.3:1 compression ratio was achieved with an RMS error of 0.23 mm (0.466%). Even when saving out the encoded sphere at the low image quality of JPEG 20%, the RMS error was only 0.42 mm (0.854%) and achieved a 6241.9:1 compression ratio versus STL. Table 4.1 shows

the overall compression ratios when Fig. 4.1(d) was encoded into different image qualities and compared against the common mesh formats.

	PNG	**JPG100**	**JPG80**	**JPG60**	**JPG40**	**JPG20**
STL	688.0 : 1	856.5 : 1	3038.3 : 1	3983.6 : 1	4795.6 : 1	6241.9 : 1
OBJ	99.9 : 1	124.3 : 1	441.0 : 1	578.2 : 1	696.0 : 1	905.9 : 1
PLY	88.6 : 1	110.2 : 1	391.1 : 1	512.7 : 1	617.3 : 1	803.4 : 1

Table 4.1.
Compression ratios of the encoded sphere using PNG and different JPEG levels versus common 3D mesh formats.

Another experiment was performed to evaluate the proposed method's ability to properly encode phase of multiple, more complex, geometries. In this experiment, a scene consisting of a cat sculpture and a dog sculpture was captured and had its phase encoded into a PNG image. This PNG image was then decoded to recover the phase from which 3D coordinates were reconstructed. Figure 4.3 demonstrates that the proposed method was indeed able to properly encode and decode phase containing multiple, complex, geometries. Figure 4.3(a) shows the original 3D geometry and 4.3(b) shows the 3D geometry reconstructed from a PNG image. Figure 4.3(c) is a rendered image that overlaps the two geometries together, showing that the difference between them is very small. Figure 4.3(d) provides an error map between the two reconstructions. The RMS error for this geometry reconstructed from a PNG image was 0.02 mm.

In the previous experiments, the encoded phase data was stored in the red and green channels; this would allow the blue channel to store texture information when desired. If the 2D RGB image is stored with a lossless compression method (e.g., PNG), it does not matter which respective channels the data and texture reside within. However, due to how most JPEG encoders typically perform their image encoding, different color channels end up being encoded at varying levels of fidelity.

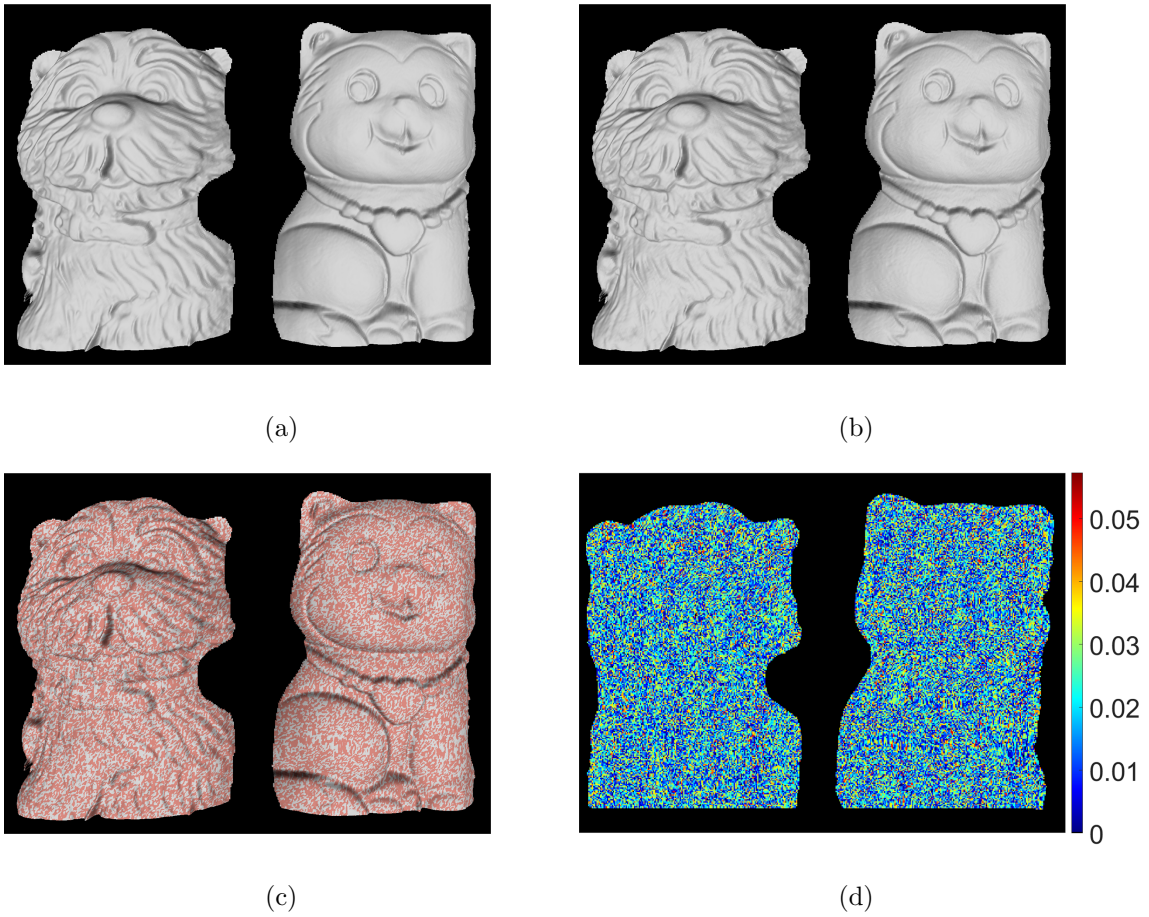


Fig. 4.3. Visual demonstration of reconstruction results when the scene contains multiple, complex geometries. (a) Original 3D geometry; (b) 3D geometry recovered from decoded phase in the red and green channels of a PNG image; (c) overlay of the original and reconstructed 3D geometries (gray color represents recovered geometry, red represents the original geometry); (d) error map, in mm, between the original and recovered geometry (RMS error of 0.02 mm).

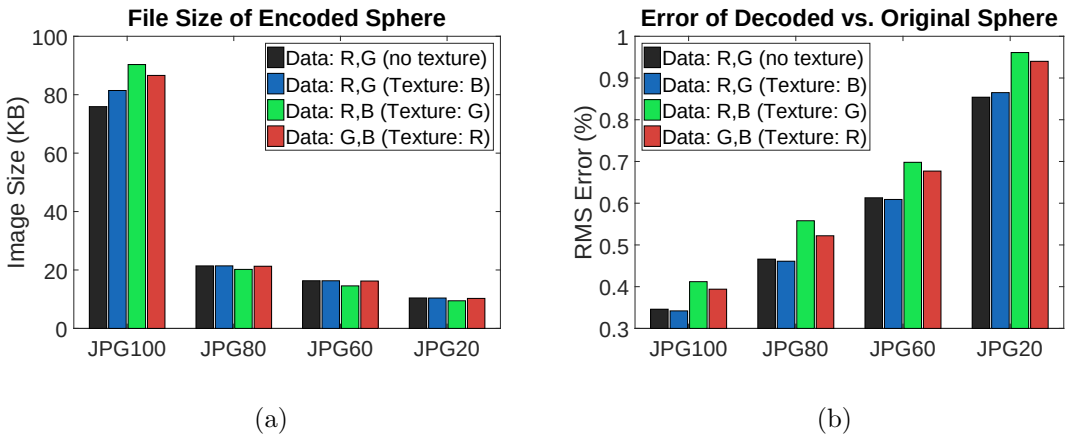
During JPEG encoding, an image's RGB values are transformed into $Y'C_BC_R$ color values, where Y' represents the luma component and where C_B and C_R represent the blue and red chroma components, respectively. The human visual system (HVS) typically is more sensitive to changes in luminance as opposed to changes in color [72]. Given this, JPEG encoders maintain high fidelity in the Y' component

and usually downsample the C_B and C_R components in aims to reduce the file size while minimizing impact on the perceptual quality of the reconstructed image. In the RGB to $Y'C_BC_R$ transformation used by JPEG, the highly-preserved Y' component is primarily influenced by the green channel, followed by the red and then blue channel values, [73]. This is done to further mimic the HVS as humans are typically more sensitive to green, red, and then blue light, respectively.

Another experiment was conducted to evaluate the impact this color JPEG encoding has on the proposed phase encoding method. In this experiment, the same 4 inch (101.60 mm) sphere from the previous experiments was used. There sphere's phase was encoded into the different channel arrangements (red and green, red and blue, green and blue) while storing the sphere's grayscale texture image in the remaining channel (blue, green, red, respectively). The RGB images were then stored using Matlab's JPEG encoder, which uses chroma subsampling, at various compression levels. From the compressed JPEG images, phase and geometries were reconstructed and compared against the original 3D capture of the sphere.

Figure 4.4(a) compares the file sizes across the various color channel arrangements. When JPEG 100% was used to store the 480×640 2D image, file sizes of 76 KB were obtained when encoding the phase into the red and green channels (leaving the blue channel empty); 81 KB encoding into the red and green channels (storing texture in the blue channel); 90 KB encoding into the red and blue channels (storing texture in the green channel); and 87 KB encoding into the green and blue channels (storing texture in the red channel). Overall, the selection of which color channels to store data and textures does not drastically affect the resulting file size of the 2D image, especially when using higher levels of JPEG compression, as the file sizes all become near equivalent.

Figure 4.4(b) compares reconstructed 3D geometry accuracies, versus the original sphere, when the different color channel arrangements are used. When JPEG 100% was used, the reconstruction errors were 0.346% when encoding phase data into the red and green channels (leaving the blue channel empty); 0.342% encoding into the



(a) (b)

Fig. 4.4. Storing encoded phase data and texture in different channels for various levels of JPEG compression. (a) Comparison of JPEG file sizes when the texture is not used or stored in the blue, green, or red channels; (b) comparison of reconstructed 3D geometry error (versus the original sphere) when phase data is encoded into various color channels, potentially along with a texture image in the remaining channel.

red and green channels (texture in the blue channel); 0.412% encoding into the red and blue channels (texture in the green channel); and 0.394% encoding into the green and blue channels (texture in the red channel). The overall trend was that the lowest errors could be achieved by encoding phase data into the red and green channels, storing texture in the blue channel, if desired.

A final experiment used the proposed phase encoding method to compress a dynamic sequence of 3D data frames along with their associated color textures. For this experiment, a color camera (PointGrey Grasshopper3 GS3-U3-23S6C) was used. Each captured frame's phase was encoded into the red and green channels of an output 2D image with the proposed method. Each frame's associated texture—before color demosaicing—was placed in the blue channel of its respective output image. Each output RGB image was then compressed using various levels of image compression: PNG, JPEG 100%, JPEG 95%, JPEG 90%, and JPEG 85%. It should be noted that chroma subsampling was not used for the JPEG encodings in this experiment.

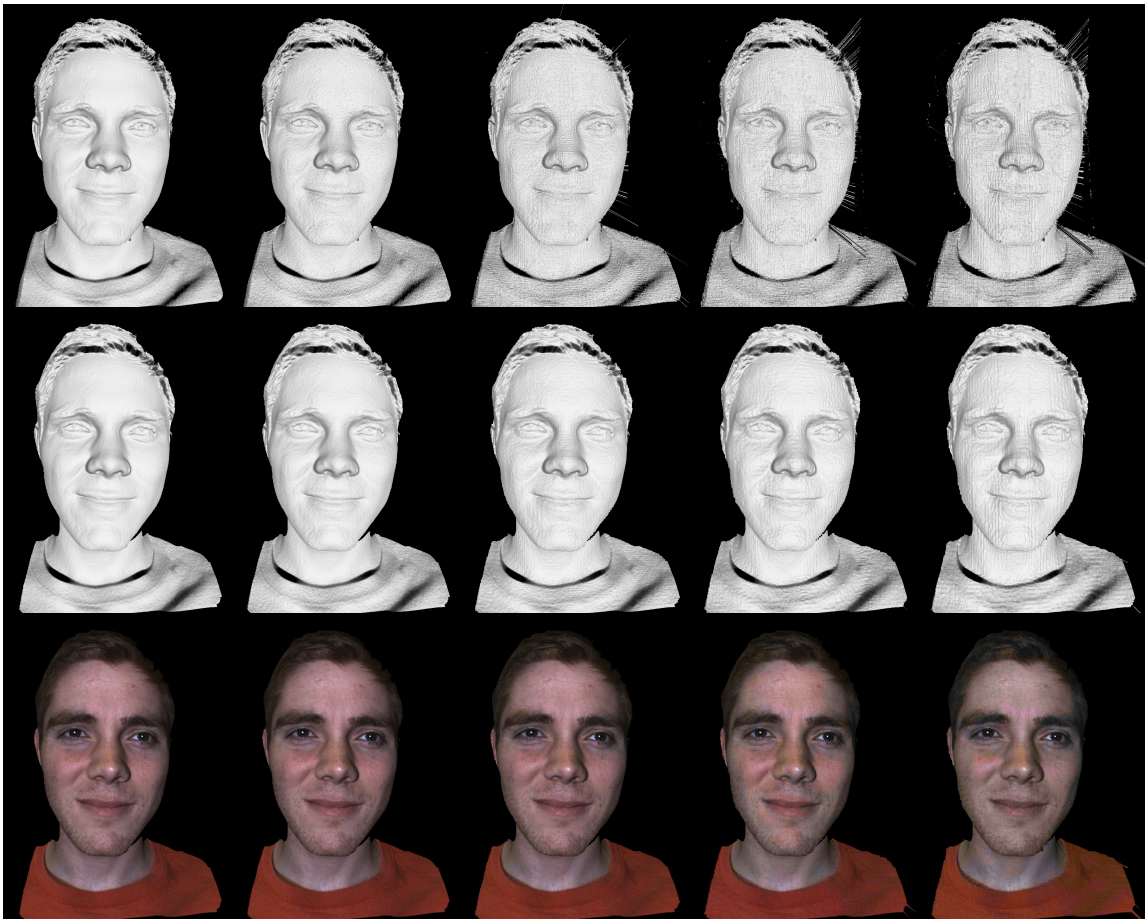


Fig. 4.5. Reconstructions of 3D data from a dynamic sequence (associated with Visualization 1). Each column, from left to right respectively, represents reconstructions from various levels of compression used to store the output 2D image: PNG, JPEG 100%, JPEG 95%, JPEG 90%, and JPEG 85%. First row: reconstructed 3D geometry from the compressed images. Second row: reconstructed 3D geometry with small median and gaussian filters applied. Third row: filtered reconstructed 3D geometry with color texture mapping applied.

Figure 4.5 shows reconstructions from the encoded images for one of the dynamic frames stored using various levels of 2D image compression: PNG, JPEG 100%, JPEG 95%, JPEG 90%, and JPEG 85%, from left to right. Visualization 1 shows several seconds of the decoded dynamic sequence. The first row shows the reconstructed 3D geometry without any post-processing or filtering. The second row shows the same

reconstructed 3D with small median and gaussian filters applied to remove noise around the edges and to reduce blocking artifacts imposed by JPEG. The third row shows the filtered reconstructed 3D geometry with color texture mapping applied. Color texture maps were obtained by demosaicing the texture stored within the blue channel of the encoded output images.

It is important to know that there is a tradeoff between accuracy and depth range. As previously mentioned in Sec. 3.2, the minimum phase unwrapping method has a limited working depth range [71] that was ensured by using a scaling factor, SF, in our proposed method. Increasing the SF extends the depth range but reduces its accuracy. Conversely, decreasing the scaling factor increases the accuracy but reduces the effective depth range of the encoding. Therefore, in practice, the selection of SF should be tailored for a given application where the depth range can be pre-defined.

4.4 Summary

This chapter presented a novel method for the compression of 3D range geometry into a regular 24-bit 2D RGB image which utilized geometric constraints of the 3D scanning device itself to reduce the amount of data which need be stored. The proposed method used two color channels to precisely represent 3D geometry information while leaving one channel free to store additional attributes about the data (such as a texture image). Our experiments demonstrated the overall efficiency and robustness of the proposed method. When PNG was used to store the encoded output image, compression ratios of approximately 688:1 were achieved versus the STL format with an RMS error of only 0.033%. Additional experiments highlighted the proposed method's resiliency to lossy JPEG image compression. For example, compression ratios of 3038:1 were achieved versus STL with an RMS error of 0.47% when the encoded image was compressed with JPEG 80%. Lastly, it was shown that the proposed method could reconstruct complex 3D geometry and color texture informa-

tion from a single, JPEG compressed 2D RGB image, which may be useful within applications such as communications and telemedicine.

5. MULTIWAVELENGTH DEPTH ENCODING METHOD FOR 3D RANGE GEOMETRY COMPRESSION

The previous chapter introduced a novel method, resilient to lossy compression artifacts, for encoding unwrapped phase data within the color channels of a regular 2D image. Although this method experimentally demonstrated its ability to accurately encode 3D range geometry, it relies on a decoding process that performs computationally expensive operations with and on matrices (e.g., matrix inverse). Similarly, several alternative methods have been presented for encoding 3D geometry into a color image (as discussed in Chapter 2), however, these often require substantial filtering or post-processing on the reconstructed geometry. In the context of mobile computing, where computational resources may be less adequate, such a methods may not be suitable for real-time scenarios. This chapter introduces a second novel method proposed by this dissertation research for accurately encoding 3D range data within the color channels of a regular 2D image. One highlight of this proposed method is that its decoding procedure is relatively computationally inexpensive, making it well-suited for, potentially mobile, real-time scenarios. The majority of content in this chapter was originally published in *Applied Optics* [69] (also listed as journal article [J2] in "LIST OF PUBLICATIONS").

5.1 Introduction

Recent advances in 3D scanning technologies have brought about the capabilities to capture high-quality data at very fast speeds [66]. Given such progress, one might consider these technologies to be on the brink of widespread dissemination. One inherent problem that must be further addressed, however, is establishing methods

for 3D range data compression that are robust and offer high compression ratios; such methods will ensure efficient storage and fast, high-quality data recovery.

Currently, one conventional storage standard for a single frame of 3D geometry is the mesh format. These formats (e.g., OBJ, PLY, STL) are generic in nature and perform their tasks well, storing the coordinates of each vertex often along with connectivity information. Additional information can also be stored with the mesh such as a surface normal map and a (u, v) map. Although these formats are able to perform their task of representing a frame of 3D geometry, they also require a large amount of storage to do so. For example, a single 640×480 frame of 3D geometry, with only vertex locations and connectivity information, needs about 13 MB of space [41]. For real-time or faster 3D capture systems that want to store or stream each single frame, this large file size becomes an issue.

Given this, other 3D range data compression techniques have been proposed. One such methodology is to encode the raw 3D data in some way such that it can be represented within a 2D image. The information stored within an image's color channels can then be used to recover and reconstruct the compressed geometry. Such approaches are able to take advantage of very well established image formats (e.g., PNG) and the infrastructure built around them.

To compress 3D geometry into a 2D image, one approach is to use the principles of virtual digital fringe projection (DFP) [41, 45, 74]. Using the conventions of DFP along with a virtual structured light scanner, this *HoloImage* approach projects fringe images upon the virtual 3D geometry and captures them virtually [74]. The resulting captured fringe images are then packed into the image (e.g., into its color channels) along with any information that may be required to unwrap the phase map between the phase images (e.g., stair image). An additional advantage to using a virtual fringe projection system which converts raw 3D geometry into a 2D image frame is its portability to video storage and streaming [42, 43].

Although able to achieve high compression ratios, virtual fringe projection approaches are limited by the spatial resolution of the image. To circumvent this,

Zhang proposed a method to directly encode the depth, Z, of 3D geometry directly into the fringe images to be stored within a 2D image's color channels [46]. This method works very well when storing its 2D images in lossless formats yet compression artifacts become difficult to avoid at high levels of lossy storage.

This chapter presents a new method of direct depth encoding, called multi-wavelength depth (MWD) encoding which uses a different technique to encode within, and recover geometry information from, a 2D image. When stored in a lossless image, the accuracy is identical to the direct depth method [46] yet boasts a smaller file size. Also, the proposed method improves upon the direct depth's encoding such that it is not affected as greatly by high levels of lossy compression. Further, direct depth's encoding technique is limited in its potential depth resolution. MWD's encoding method, however, lifts these restrictions to allow for high-resolution depth encoding. In our experiments, the new MWD method out-performed the current state-of-the-art direct depth method in terms of file size, recovered data accuracy, amount of post-processing required, and maximum depth resolution. For example, when the proposed MWD method was used to encode 3D geometry into a 2D image which was then stored using JPEG 80% from Matlab 2014a, the resulting file size for one frame was 30.3 KB. This is approximately a 935:1 compression ratio versus the same data in the OBJ format. In addition, the reconstruction error percent achieved was 0.027% without any post-processing filtering applied. Further experimental results will be presented to verify the capability of the proposed 3D range compression method.

Section 5.2 will explain the principles behind the structured light techniques employed, the encoding of the 3D geometry, and the decoding for the recovery of the geometry using the proposed method. Section 5.3 will present experimental results of the proposed compression method. Section 5.4 will discuss the results and merits of this compression method. Lastly, Section 5.5 will summarize the contributions of this research.

5.2 Principle

5.2.1 Phase-Shifting Technique for 3D Shape Measurement

The general principles behind the aforementioned 3D range data compression approaches are derived from structured light scanning techniques. Similar to a two camera stereo-vision method, a structured light method instead replaces one camera with a projector; thus, modern structured light systems are typically comprised of one camera and one projection unit. There are many structured light techniques [75], yet one that has stood out due to its simplicity, speed, and accuracy is digital fringe projection (DFP). In systems using DFP, patterns are generated which vary in intensity sinusoidally; to generate different patterns, the sinusoid structure can be shifted by some amount. These patterns are then projected sequentially onto some scene to be captured. Figure 5.1 displays an example of a 3D structured light scanner using a DFP approach. One example merit of phase-shifting techniques is their ability to achieve pixel-by-pixel spatial resolution. Many phase-shifting algorithms have been developed, and a summary of them can be found in this book chapter [76].

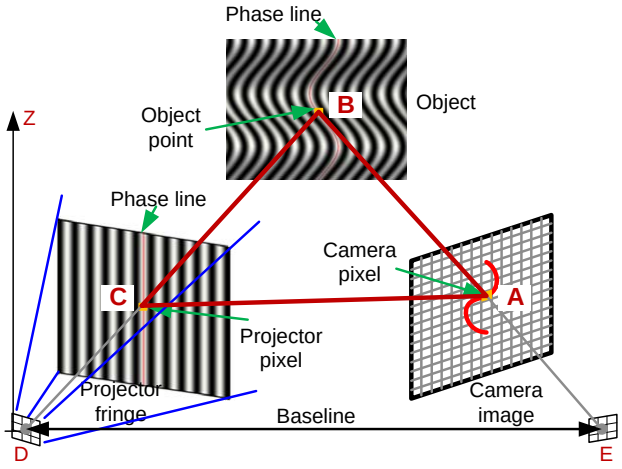


Fig. 5.1. Example structured light system setup using a digital fringe projection (DFP) technique.

With phase-shifting algorithms, the more steps used results in a lowered sensitivity to linear phase shifting errors [77]; however, with more steps brings a longer projection cycle. Given this, real-time 3D structured light scanning applications will traditionally use a three-step phase-shifting algorithm. The three-step algorithm requires a shorter projection cycle and can still render a high-quality phase. The three fringe images with equal phase shifts to be used within a three-step phase-shifting algorithm can be derived via

$$I_1(x,y) = I'(x,y) + I''(x,y)\cos(\phi - 2\pi/3), \tag{5.1}$$

$$I_2(x,y) = I'(x,y) + I''(x,y)\cos(\phi), \tag{5.2}$$

$$I_3(x,y) = I'(x,y) + I''(x,y)\cos(\phi + 2\pi/3), \tag{5.3}$$

where I' is the average intensity; where I'' is the intensity modulation; and where $\phi(x,y)$ is the phase to be solved for. This can be done via

$$\phi(x,y) = -\tan^{-1}\left[\frac{\sqrt{3}(I_1 - I_3)}{2I_2 - I_1 - I_3}\right]. \tag{5.4}$$

The resulting phase will contain values ranging from $-\pi$ to π with 2π discontinuities. This *wrapped phase*, ϕ, can then be *unwrapped* using a spatial phase-unwrapping algorithm, such as one described in [78], to recover a continuous, unwrapped phase map, Φ. Finally, if the structured light system is calibrated correctly [79], real-world coordinates (X, Y, Z) can be extracted from $\Phi(x,y)$ pixel-by-pixel.

5.2.2 HoloImage Encoding

The aforementioned *HoloImage* compression approach uses a *virtual* structured light system (i.e., virtual camera and virtual projection device) to encode the virtual geometry. Being that the resulting system is computer generated, everything can be

precisely controlled, including: surface reflectivity, ambient light in the scene, and the spatial sampling distribution. As previously mentioned, in real environments, at least three fringe images are required to recover adequate phase, however, given that all of these factors can be controlled in a virtual system, only two fringe images are necessary. Due to the nature of the HoloImage system, the resultant fringe stripes vary horizontally along the x direction and are vertical along the y direction, or visa versa. Since only two images are required to recover phase, a third image can then be used to support the unwrapping of the wrapped phase between the two fringe images. These three images, two fringe images and one stair image, are then packed into each respective channel of a 2D image which can be further compressed with existing 2D image compression techniques (e.g., PNG).

To recover geometry, the fringe images and stair image are extracted from the 2D image's color channels. The wrapped phase, ϕ, can be computed with the two fringe images similar to Eq. (5.4). The stair image can then be used to realign the 2π discontinuities such that a continuous phase map, Φ, is recovered. As discussed in Subsec. 3.25.2.1, (X, Y, Z) can then be recovered. Exact details of the HoloImage approach are covered in [41].

Although able to achieve high quality compression ratios, HoloImage techniques are not without their drawbacks. One crucial drawback is that HoloImage techniques are limited due to spatial resolution constraints. The fringe images which are virtually projected either vertically or horizontally along the virtual object must have a decent stair height, in practice 10 or more [46], to make the encoded images less sensitive to noise. Using a larger stair height means that there are fewer stairs to use across the image. Given this, dense fringe images are not adequately encoded and therefore high, spatial resolutions cannot be reached. To alleviate the constraint of spatial resolution, the depth Z itself can be directly encoded.

5.2.3 Direct Depth Z Encoding

To remove the limitations imposed by the spatial resolution of HoloImage techniques, where sampling is done along the x or y direction, the depth Z itself can be encoded. This is achieved with an approach similar to HoloImage except that the sampling takes place along the z direction. Following [46], two fringe images are still used to encode the data with a third image again being used as a stair image. These three images are then stored in the three channels of a 2D image to be further compressed with, traditionally, a lossless format (e.g., PNG). The three encodings to be stored in each channel can be described via

$$I_r(i,j) = 0.5\left[1.0 + \sin\left(\frac{2\pi Z}{P}\right)\right], \tag{5.5}$$

$$I_g(i,j) = 0.5\left[1.0 + \cos\left(\frac{2\pi Z}{P}\right)\right], \tag{5.6}$$

$$I_b(i,j) = S \times Fl(Z/P) + 0.5S + 0.5(S-2) \times \cos\left[2\pi \times \frac{Mod(Z,P)}{P_1}\right] \tag{5.7}$$

where (i,j) are the image pixel indices; where P is the fringe width, or the number of pixels per each fringe stripe; where $P_1 = P/(K+0.5)$ is the local fringe pitch where K is an integer; where $Fl(x)$ represents the $floor$ function; where S represents the grayscale stair height; and where $Mod(a,b)$ represents the modulus operator.

To recover 3D geometry from the 2D image, the image's different color channels are used. In the direct encoding method X and Y coordinates are uniformly sampled and are therefore proportional to their image indices (i,j). Given this, X and Y can be recovered by some user specified scaling constant, described as

$$X = j \times c, \tag{5.8}$$

$$Y = i \times c. \tag{5.9}$$

The depth map, Z, can be recovered from the stored 2D image via

$$Z = P\left[Fl\left(\frac{I_b - 0.5S}{S}\right) + \frac{1}{2\pi} \times \tan^{-1}\left(\frac{I_r - 0.5}{I_g - 0.5}\right)\right]. \tag{5.10}$$

The direct depth encoding technique works quite well [46] and eliminates the spatial resolution constraints imposed by the HoloImage technique; however, the third channel, I_b as given in Eq. (5.7), limits this approach when high levels of JPEG compression are used. When reconstruction takes place from low-quality JPEG images using this approach, the stair image may degrade which causes artifacts during phase unwrapping thus large filters must be used to remove such artifacts. Also, some ringing artifacts may be imposed due to the cosine-wrapping of the stair image. Further, in both lossy and lossless domains, this technique has a limited depth resolution. The grayscale stair height, S, constrains I_b as the range of the depth image and the number of stairs used to encode the geometry become larger. To alleviate these drawbacks, the third channel needs to be replaced with an alternative that is less sensitive to lossy, JPEG compression and can encode large depth resolutions.

5.2.4 Multi-wavelength Depth (MWD) Encoding

This chapter presents a novel method for 3D range data compression based on direct depth encoding and utilizes principles of multi-frequency phase-shifting techniques [80]. In the proposed approach, two different fringe widths are used to generate four fringe encodings: one of a shorter, user defined, width P, and one longer with a width equivalent to the range of Z, $Rng(Z)$. Mathematically, the encoding of each fringe image can be described via

$$I_r(i,j) = 0.5\left[1.0 + \sin\left(\frac{2\pi Z}{P}\right)\right], \tag{5.11}$$

$$I_g(i,j) = 0.5\left[1.0 + \cos\left(\frac{2\pi Z}{P}\right)\right], \tag{5.12}$$

$$I_b^{sin}(i,j) = 0.5\left[1.0 + \sin\left(2\pi \times \frac{Mod(Z, Rng(Z))}{Rng(Z)}\right)\right], \tag{5.13}$$

$$I_b^{cos}(i,j) = 0.5\left[1.0 + \cos\left(2\pi \times \frac{Mod(Z, Rng(Z))}{Rng(Z)}\right)\right]. \tag{5.14}$$

The first two fringe encodings stored in the red and green color channels are identical to the red and green channels of the direct depth method. Instead of storing a cosine-wrapped stair image in the third channel, however, the wrapped phase of the two fringes encoded with the longer wavelength is stored and can be computed via

$$\phi_b(i,j) = \tan^{-1}\left(\frac{I_b^{sin} - 0.5}{I_b^{cos} - 0.5}\right). \tag{5.15}$$

It should be noted that resulting phase map encoded within ϕ_b has a range between $(-\pi, \pi]$. To represent this range instead within a range $(0, 1]$ such that it can be stored in the color channel properly, the following is done:

$$I_b(i,j) = \frac{\phi_b + \pi}{2\pi}. \tag{5.16}$$

The resulting 2D image can then be stored within a lossless format such as PNG or within a lossy format, such as JPEG, at varying levels of compression.

5.2.5 Multi-wavelength Depth (MWD) Decoding

To recover geometry from a compressed 2D image, the wrapped phase map between the two fringe images of the shorter wavelength is first computed via

$$\phi_{rg}(i,j) = \tan^{-1}\left(\frac{I_r - 0.5}{I_g - 0.5}\right). \tag{5.17}$$

Now, as the $Rng(Z)$ was used as the fringe width for the longer wavelength, the encoding of each fringe image generated with this longer width spans the entire depth of the scene. Given this, in the wrapped phase map ϕ_b stored within I_b, there will be

no 2π jumps; it will be continuous. This continuous wrapped phase map, ϕ_b, can be used to unwrap the more dense phase map, ϕ_{rg}, which does contain 2π discontinuities.

The first step is to convert the wrapped phase information in I_b back into its original range of $(-\pi, \pi)$ from its current range of $(0, 1)$ via

$$\phi_b(i,j) = I_b \times 2\pi - \pi. \tag{5.18}$$

Next, a stair image is computed from the continuous ϕ_b which in turn can be used to reversely unwrap the phase of the shorter wrapped phase, ϕ_{rg}. The stair image is described via

$$K(i,j) = \frac{\phi_b \times Rng(Z)/P - \phi_{rg}}{2\pi}. \tag{5.19}$$

The stair image K's values determine the scale of how many 2π must be added to ϕ_{rg} to remove its 2π discontinuities. Removing the phase jumps results in a continuous, unwrapped phase, Φ, of the shorter wavelength. This process can mathematically be described as

$$\Phi(i,j) = \phi_{rg} + 2\pi \times Rnd(K), \tag{5.20}$$

which is finally used to recover depth Z via

$$Z(i,j) = \frac{\Phi \times P}{2\pi}. \tag{5.21}$$

X and Y can be recovered as they were in Eq. (5.8) and (5.9).

5.3 Experimental Results

In our primary experiments, an ideal, computer-generated semi-sphere was used to verify the performance of the multi-wavelength depth (MWD) encoding method. After encoding the 3D geometry into a 2D image of 512 × 512 as described in Subsec. 3.25.2.4, the image was either saved out in a lossless format (PNG) or a lossy format (JPEG) at different levels of compression using Matlab (2014b).

Figure 5.2 shows each channel and step along the encoding and decoding process of the MWD technique. Figure 5.2(a) shows the compressed, encoded 2D color image with Fig. 5.2(b)-Fig. 5.2(d) showing the image's individual red, green, and blue color channels, respectively. To recover geometry from the saved out image, the wrapped phase is needed between the fringe images of the shorter wavelength; the resulting wrapped phase is shown in Fig. 5.2(e). As described previously, the continuous, wrapped phase of the longer wavelength, shown in Fig. 5.2(d), can be used to derive a stair image, shown in Fig. 5.2(f). The stair image is then used to unwrap the phase of the shorter wavelength to get the unwrapped phase map, Fig. 5.2(g), which can then be used to recover 3D geometry, as displayed in Fig. 5.2(h).

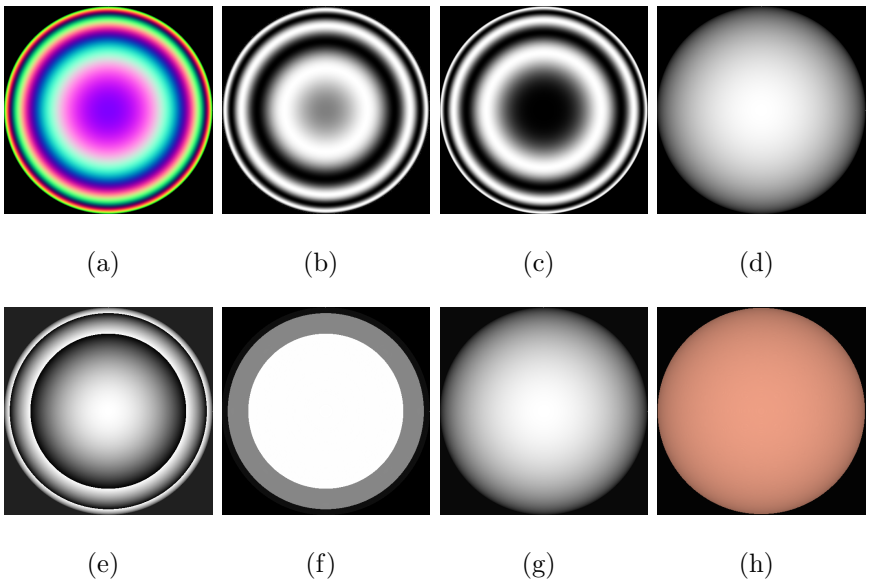


Fig. 5.2. Experimental results of encoding and decoding an ideal sphere using the multi-wavelength depth encoding method (a) encoded 2D color image; (b)-(d) the three encoded color channels, red, green, blue, respectively; (e) wrapped phase between the two shorter wavelength fringe images, (b) and (c); (f) stair image derived from the wrapped phase map of the longer, continuous wavelength, (d); (g) the unwrapped phase map; (h) recovered 3D result.

To help demonstrate the results of the proposed multi-wavelength depth (MWD) encoding method, comparisons were made with a similar and current state-of-the-art 3D range geometry compression method, the direct depth (DD) encoding [46] technique. It is important to note that all comparisons between MWD and DD were done so on the same geometry and with an identical experimental setup; the only difference being the principle behind the encoding methods themselves. In both cases, a semi-sphere was encoded into a 512×512 image using 4 stairs.

Figure 5.3 shows comparisons of the visual quality of the recovered data using different JPEG compression levels with each column representing JPEG 100%, JPEG 80%, JPEG 60%, JPEG 40%, and JPEG 20%, respectively, using Matlab 2014a JPEG compression. The top row displays the recovered sphere as encoded via direct depth (DD) method without any filter applied to recover from JPEG compression artifacts. The second row shows the same DD encoding method with a 25×25 filter applied to address compression artifacts. The third row demonstrates the quality of the proposed MWD encoding as no filter was applied to achieve the results. It can be seen that the quality of the recovered geometry using direct depth quickly degrades under JPEG compression and requires a large amount of filtering to recover geometry. MWD on the other hand remains spherical and uniform at high levels of compression without the need for filtering. Even using JPEG 20%, which stores the compressed sphere using only 14 KB for a 512×512 image, the recovered data using the MWD approach maintains its original geometric shape and does not require any filtering to do so.

It can be seen visually that the recovered data remains spherical, even at low JPEG qualities. To empirically verify the accuracy of the recovered 3D sphere, a cross section of the ideal data was compared with the same cross section of the decoded data. Figure 5.4 shows the results for both the DD and MWD approach where 3D data was recovered from a PNG image as well as JPEG images stored at various levels of compression: 100%, 80%, and 60%, respectively.

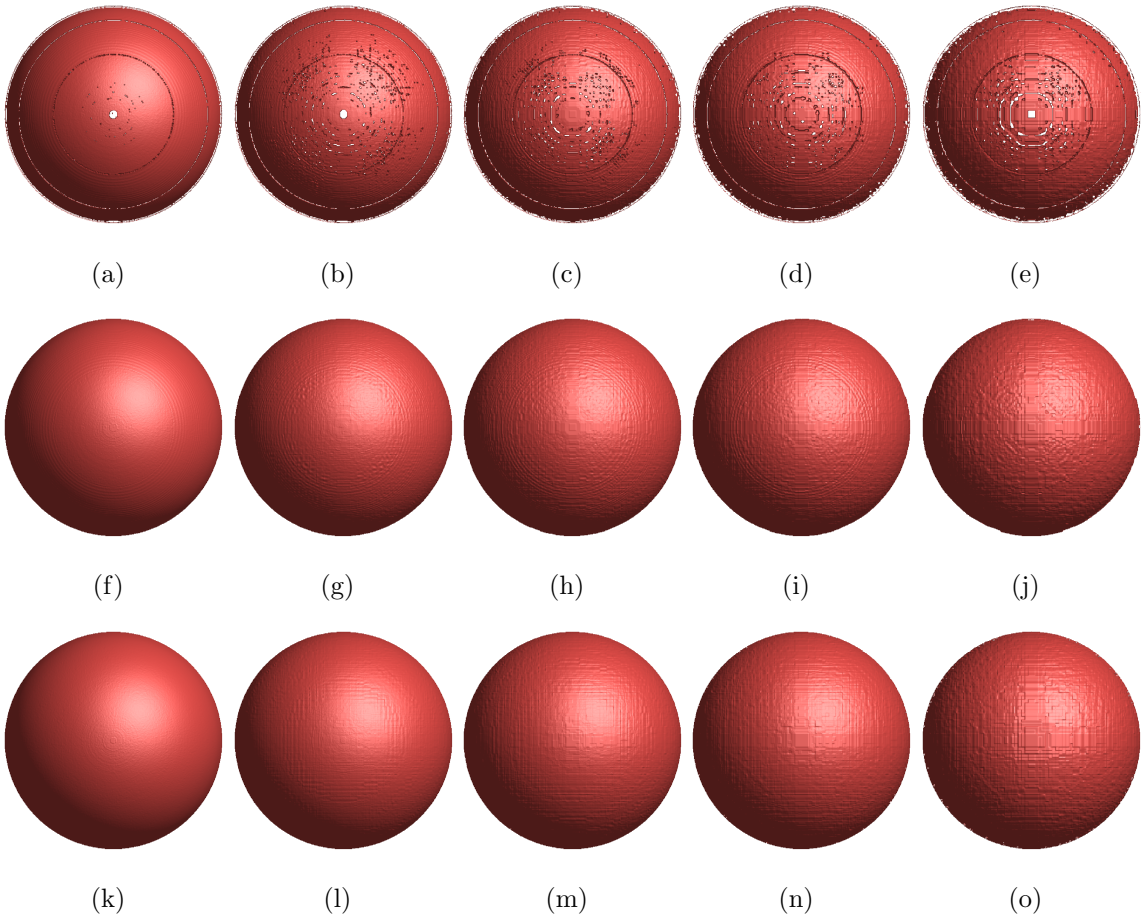


Fig. 5.3. Visual comparison of recovered sphere geometry using different encoding methods, JPEG compression levels, and filtering amounts. Each column represents JPEG storage levels 100%, 80%, 60%, 40%, and 20%, respectively, using Matlab 2014a JPEG compression. (Row 1) direct depth method without a filter applied to the recovered geometry; (Row 2) direct depth method with a 25×25 filter applied to the recovered geometry; and (Row 3) multi-wavelength depth method without any filter applied.

The normalized RMS error percentages of the visual results seen in Fig. 5.3 are given in Table 5.1; it should be noted that when calculating the error in both approaches the five boundary pixels were ignored to eliminate outliers. From this table, it can be seen that the DD and the proposed MWD approaches are identical when using lossless image compression, however, the accuracies differ when lossy compres-

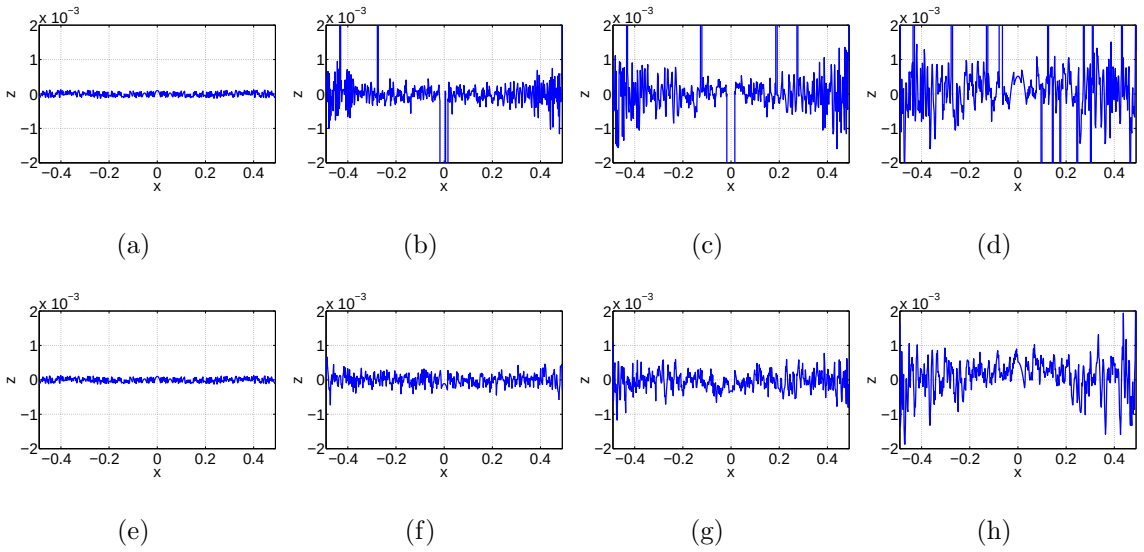


Fig. 5.4. Experimental results comparing the ideal sphere and the recovered 3D geometry, using DD or MWD, from different qualities of 2D images. Each column represents storage qualities PNG, JPEG 100%, JPEG 80%, and JPEG 60%, respectively; for each plot, the x-axis represents the normalized pixel coordinates along the cross section and the y-axis represents the normalized difference between the ideal and recovered geometries for each coordinate. (Row 1) direct depth cross section error difference versus ideal geometry. RMS error percentages for (a)-(d) are 0.0061%, 2.9557%, 3.4834%, and 3.397%, respectively; (Row 2) multi-wavelength depth cross section error difference versus ideal geometry. RMS error percentages for (e)-(h) are 0.0061%, 0.0167%, 0.0271%, and 0.0508%, respectively.

sion is used. This is due to direct depth's third channel which does not compress well and is the cause of lossy compression artifacts. The continuous nature of the third channel of the MWD approach, however, allows for less JPEG artifacts and thus higher geometry reconstruction accuracies.

To get extremely small sizes (7-10 KB), JPEG qualities can be reduced to 5-10%. At this very low quality, both approaches begin to exhibit compression artifacts in the recovered data. To remove these spikes from either approach, a median filter is traditionally used. Throughout our experiments with compressing the sphere and

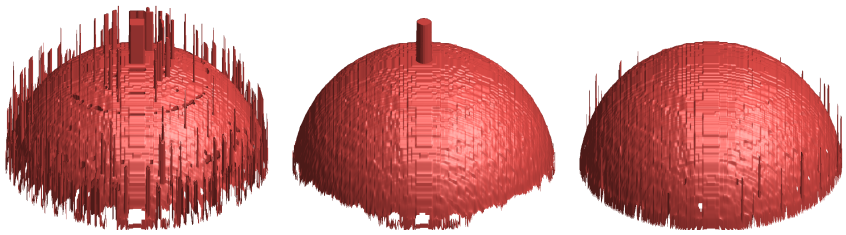
	PNG	JPG100	JPG80	JPG60	JPG40	JPG20
DD No Filter	0.0061%	2.9557%	3.4834%	3.3970%	3.8545%	4.5275%
DD 25x25 Filter	0.0061%	0.0263%	0.0438%	0.0578%	0.0725%	0.0924%
MWD No Filter	0.0061%	0.0167%	0.0271%	0.0508%	0.0651%	0.0928%

Table 5.1.
Comparison of RMS error between a normalized sphere compressed into a 512×512 image using the proposed MWD encoding versus the direct depth (DD) encoding. Associated JPEG levels represent the compression quality at a percentage out of 100 using Matlab 2014a.

other geometries, MWD always required a smaller filter size than DD when recovering data from JPEG encoded images; this results in a faster decode speed for MWD which would have great implications if in use within a real-time streaming environment where compressed 3D frames must be decoded and displayed as they are arriving. For example, as shown in Fig. 5.5, MWD performs better at the very low JPEG quality of 10% than DD when using the same filter size of 5×5. Even the very large filter size of 25×25 was unable to rid the DD method completely of artifacts due to such high levels of JPEG compression whereas MWD's 5×5 filter proved enough.

Not only does the MWD approach require less filtering, the subject of the filter is different from other state-of-the-art depth compression approaches. Traditionally, what is filtered to remove spikes is the final, recovered phase thus affecting the accuracy of the reconstruction. With the proposed MWD approach, however, only the third channel must be filtered in most cases, leaving the red and green channels, the ones containing the encoded geometry data, untouched. This contributes to a higher accuracy of data recovery.

As demonstrated above, the quality, both visually and statistically, of recovered data even at high levels of lossy compression is quite good for the proposed method. This robustness to such low JPEG qualities provides this technique the ability to represent 3D geometry using very little information thus achieving high compression



(a) (b) (c)

Fig. 5.5. Comparison of filtered recovered data from JPEG 10% encodings; (a) 3D recovery from DD at JPEG 10% using a 5×5 filter; (b) 3D recovery from DD at JPEG 10% using a 25×25 filter; (c) 3D recovery from MWD at JPEG 10% using a 5×5 filter. The RMS errors for (a), (b), and (c) are 6.13%, 3.93%, and 0.15%, respectively.

ratios. Table 5.2 below shows the ratios of MWD in comparison to popular 3D storage types. In each case, the geometry stored was the ideal sphere. As comparison, Table 5.3 shows the ratios of using DD. It has now been shown that in general, the proposed method has both larger compression ratios, or smaller data sizes, and lower error percentages.

	PNG	**JPG100**	**JPG80**	**JPG60**	**JPG40**	**JPG20**	**JPG10**
OBJ	146.8:1	244.3:1	935.3:1	1294.1:1	1583.3:1	2147.0:1	2834.1:1
PLY	52.0:1	86.5:1	331.0:1	458.0:1	560.3:1	759.8:1	1003.0:1
STL	103.7:1	172.6:1	660.7:1	914.1:1	1118.4:1	1516.6:1	2001.9:1

Table 5.2.
Compression ratios of MWD using PNG and different JPEG storage levels versus 3D mesh file formats.

To ensure that the proposed method worked on complex geometries, a statue was tested. Figure 5.6 compares the DD and MWD approaches to encoding the same complex geometry, each using 4 stairs to encode 512×512 resolution data into a

	PNG	JPG100	JPG80	JPG60	JPG40	JPG20	JPG10
OBJ	91.7:1	187.7:1	708.5:1	1069.5:1	1375.8:1	1968.1:1	2725.1:1
PLY	32.5:1	66.4:1	250.8:1	378.5:1	486.9:1	696.5:1	964.4:1
STL	64.8:1	132.6:1	500.5:1	755.4:1	971.8:1	1390.2:1	1924.9:1

Table 5.3.
Compression ratios of DD using PNG and different JPEG storage levels versus 3D mesh file formats.

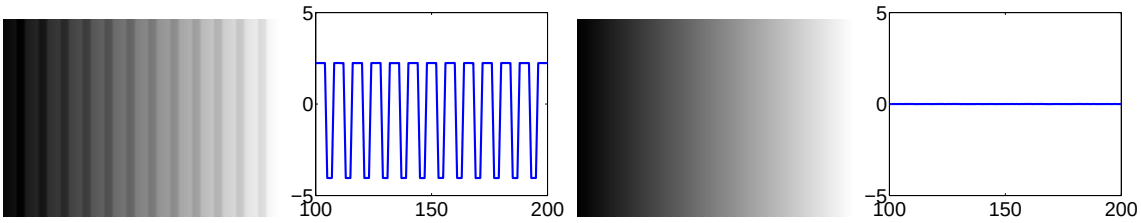
512 × 512 compressed image. The results shown in the top row compare DD with a 15×15 filter applied and MWD with a 5×5 filter applied both using JPEG 95%. The bottom row compare DD and MWD using the same filter sizes except this time using JPEG 40%. It can be seen that the much larger filter does not rid DD's recovered geometry of its compression artifacts whereas MWD's complex geometry is preserved and recovered from this low JPEG quality using only the small 5×5 filter.

(a) (b) (c) (d)

Fig. 5.6. Encoding complex geometries. (a) DD recovered geometry using a 15×15 filter and JPEG 95%; (b) MWD recovered geometry using a 5×5 filter and JPEG 95%; (c) DD recovered geometry using a 15×15 filter and JPEG 40%; (d) MWD recovered geometry using a 5×5 filter and JPEG 40%.

Additional experiments were conducted to test MWD's ability to encode high-resolution depth images. Instead of a sphere, geometry of a flat plane with a constantly increasing depth was encoded. It was found that the MWD approach was able to encode large depth ranges, with the largest tested being a 4,096 $\times$ 4,096 depth image having a depth resolution of 4,096 (over 16 million points at a 4K range). Similarly, values of 1,024 and 2,048 were tested, as well. The MWD method was able to retain high-qualities of information after decoding being that up to 256 stair values could be used to encode the depth. The same tests, however, did not bode well for the direct depth approach; as the number of stairs increase, the smaller the step height becomes. The result of this are phase unwrapping issues which induce 2π-tall spike artifacts in the recovered phase; these are unable to be removed with a simple filter. Even with lossless encoding, the direct depth method was unable to encode the plane at a depth resolution of 1,024 using 128 stairs, for example. Figure 5.7's first row shows a normalized small window of the unwrapped phase and the phase's cross section with its continuous slope removed when using the direct depth method to encode a scene with a depth range of 1,024 using 128 stairs into a PNG image. The second row shows the same small window and cross section with its slope removed, this time using MWD to encode the scene with the same depth range of 1,024 using 128 stairs. As it can be seen, the MWD approach correctly unwraps the phase and can therefore recover proper geometry.

For the direct depth approach to feasibly encode large depth ranges, the number of stairs must be lowered to increase the grayscale stair height which results in poor quality reconstruction at high-resolution depths. The MWD approach, however, was able to encode the plane at experimental resolutions of 1,024, 2,048, and 4,096 with up to 256 stairs. Using a 4,096 $\times$ 4,096 depth image of the experimental plane with a depth resolution of 4,096, the MWD method was able to encode the plane with 256 stairs at a normalized RMS error rate of 0.00015%. These high-resolution depth images could be encoded using lossy storage with the MWD approach, as well, yet a fewer number of stairs had to be used to avoid compression artifacts. For example,



(a) (b) (c) (d)

Fig. 5.7. Visual demonstration showing the DD's inability and MWD's ability to encode high-resolution depths. (a) A normalized subwindow of the recovered phase when using the direct depth technique to encode geometry with a depth resolution of 1,024 and 128 stairs; (b) a cross section of the recovered phase (a) with its slope removed, the resulting 2π-tall spikes leave artifacts within the data; (c) the normalized subwindow of recovered phase when using the multi-wavelength technique to encode the same geometry with the same number of stairs; (d) cross section of the recovered phase (c) with its slope removed.

using the MWD with 34 stairs along with JPEG 100%, the same 4,096 $\times$ 4,096 depth image with a depth resolution of 4,096 was able to be encoded and recovered with a RMS error of 0.0024%.

5.4 Discussion

The proposed multi-wavelength depth encoding method has the following advantages and merits over the previously proposed direct depth encoding technique:

- *Higher compression ratios.* The nature of the encoding within the proposed technique, being continuous in the third channel, can be represented with less information. The result of this is smaller file sizes both using PNG and JPEG to compress and store the color encoded 2D image.

- *Higher accuracy 3D data recovery.* Although identical under most circumstances when using PNG, the proposed method is much more robust, due to its more

continuous and uniform nature, to high levels of JPEG compression. This results in less compression artifacts due to high levels of lossy compression.

- *Little to no filtering required.* Depending on the data to be encoded, the level of JPEG compression, and the number of stairs used to encode the data, the proposed method requires little to no post-processing to fix artifacts; thus, decode computational complexity and time are lessened. Also, if post-processing must occur, only filtering the third channel was found to be necessary in most circumstances. Whereas in the direct depth approach the recovered data itself was filtered, only having to filter the third channel in the proposed technique keeps the integrity of the two data channels in tact. The result is an improvement of the overall quality of recovered data.

- *High-resolution depth encoding.* In the proposed technique, up to 256 stairs can be used to encode the data. The direct depth technique must use a lower number of stairs to account for the requirement of a decently large grayscale stair height. As more stairs can be used, high-resolution depth images can be encoded using the proposed technique. Experiments were able to encode geometries with depth resolutions of up to 4,096, for example.

It should be noted that the MWD technique proposed suffers from the same drawback as the aforementioned HoloImage and direct depth techniques in that only one side of a 3D surface is able to be encoded into a 2D image. This drawback is alleviated, however, if the proposed method is properly paired with a 3D scanning device (in which only one side of an object is captured). If the goal is to encode an object in 360 degrees, many encodings could be taken from differing positions *around* the object. These scans need to be decoded, registered, and fused together to regenerate the object's original geometry.

5.5 Summary

We have presented the multi-wavelength depth (MWD) encoding method for 3D range compression which represents 3D geometries in 2D images. These 2D images can be stored within both lossless and lossy formats to achieve additional data compression. In the past, traditional approaches may suffer compression artifacts to due high levels of lossy compression, yet it has been demonstrated that the proposed method works well with under various, and even extreme, levels of lossy compression. The result of this is high compression ratios, small file sizes, and high 3D data reconstruction accuracies with little to no filtering required.

6. HOLOSTREAM: HIGH-ACCURACY, HIGH-SPEED 3D RANGE VIDEO ENCODING AND STREAMING ACROSS STANDARD WIRELESS NETWORKS

The previous chapters have presented the details of this dissertation research's contributions toward (1) capturing high-resolution and high-quality 3D video data and (2) accurately and efficiently encoding 3D geometry information. Utilizing these contributions, this dissertation research aimed to address the long-standing challenge of high-quality, low-bandwidth 3D video communications. This chapter will introduce and detail *Holostream*, a novel platform for high-quality 3D video recording, encoding, compression, decompression, visualization, and interaction. This platform was able to successfully deliver video-rate, photorealistic 3D video content over standard wireless networks to mobile devices. In addition, to emphasize the efficiency of the platform, this dissertation research developed a fully mobile Holostream implementation that allows mobile devices to both send and receive 3D video content, in real-time, across wireless networks. The majority of content in this chapter was originally published in the 2018 proceedings of *Electronic Imaging* [81] (also listed as conference paper [C3] in "LIST OF PUBLICATIONS").

6.1 Introduction

Telecommunication methods evolved from telegraphs, telephones, and 2D video conferences with each advancement bringing an enhanced end user experience through more realistic interactions. None of these techniques are advantageous for truly natural interactions as users do not have context about the 3D world in which we live. Therefore, 3D video communication emerges as an interesting research subject because of the increased availability of novel 3D sensing technologies, enhanced network

bandwidths, and increased computational power, even on a mobile device (e.g., tablets and smart phones).

Existing techniques for 3D video communications primarily focus on the application of teleconferencing and immersive telepresence. In each, an object at one location is captured in 3D and transmitted to a remote location for redisplay. Although these techniques may successfully accomplish their goals, compared with their mature 2D video communication counterparts, the state-of-the-art 3D video communication platforms are constrained by three major factors: (1) the availability of commercial 3D sensors that can provide equivalent high quality and resolution; (2) the drastically increased 3D data sizes that exceed the bandwidths of existing wireless networks; and (3) the limited real-time 3D data compression platforms which can efficiently reduce data sizes whilst maintaining data quality. Due to these limitations, high-quality 3D video communication across standard wireless networks is still in its infancy.

This chapter proposes a novel modular platform, dubbed *Holostream*, which enables high-quality 3D video communications across existing standard wireless networks and existing mobile hardware devices (e.g., iPhones and iPads). Our novel platform advances the quality and capabilities of applications already utilizing real-time 3D data delivery (e.g., teleconferencing, telepresence). Further, the proposed platform could also enable applications where real-time delivery of high-resolution, high-accuracy 3D video data is especially critical, such as collaborative design and online facial behavior analysis.

6.1.1 Contributions

- We developed a novel 3D video compression method that can drastically reduce 3D video data size without substantially sacrificing data quality. With lossless frame by frame storage, the compression ratio is approximately 112:1 when compared with standard OBJ files. Lossy frame by frame storage can achieve a ratio of 518:1 without substantial quality reduction. When paired with the

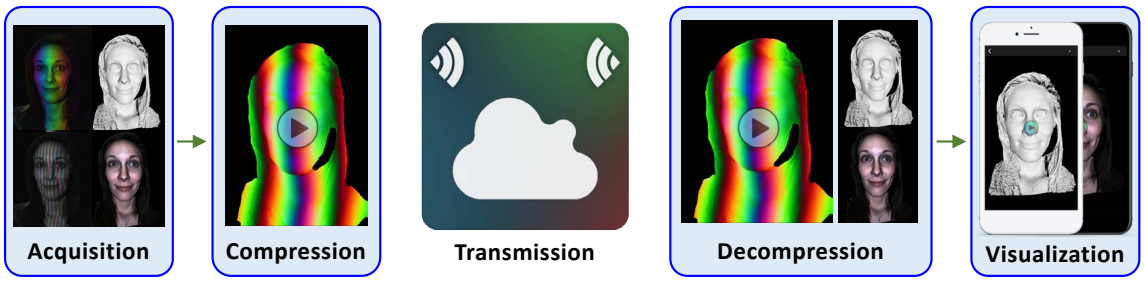


Fig. 6.1. *Holostream* pipeline starts with the *Acquisition Module* that captures 3D geometry and color texture in real time. The *Compression Module* encodes 3D data into 2D images that are packed into a video stream using a standard 2D video codec (e.g., H.264). The compressed 2D video is transmitted over existing standard wireless networks through the *Transmission Module*. The *Decompression Module* receives the compressed 2D video and decompresses it to recover the original 3D geometry and color texture. The *Visualization Module* on mobile devices (e.g., iPads, iPhones) visualizes 3D video in real time and allows the users to interact with the video instantaneously.

H.264 video codec, our method achieved a compression ratio of 1,602:1, with a bitrate of only 4.8 megabits per second (Mbps), while maintaining high-quality 3D video representations.

- We developed a novel and complete framework for 3D video recording, encoding, compression, decompression and visualization. The entire framework was tested using standard medium bandwidth networks to simultaneously deliver high-quality 3D videos to multiple mobile devices.

- We developed a demonstration system that achieved high-quality 3D sensing, compression, transmission, and visualization across standard wireless networks. The system wirelessly delivered precisely aligned coordinate and color data, consisting of over 300,000 vertices per frame, at 30 Hz to mobile phones and tablets.

6.2 Related Work

Much work has been done to enable realistic 3D telecommunications in the past several decades. Most recently, the Holoportation system [18] impressively acquires a colored 3D mesh of a user or object in one location and transmits it to remote users for visualization on augmented reality headsets. Although this system enables very natural user interactions, 1-2 Gbps of data must be transferred to realize communications at 30 Hz. This data rate greatly exceeds the bandwidth availabilities of today's existing wireless networks. In fact, the majority of the work in this area does not effectively address one of the major issues: how can 3D data sizes be drastically reduced? Instead, it is typically assumed that the network bandwidth will simply be available. Of course, this assumption does not hold true in practice and thus it is still very difficult to realize high-quality, real-time 3D telecommunications over standard wireless networks.

In a tangential community, researchers have attempted to compress 3D data, both within the mesh format [30, 31] and the point cloud format [82–86]. As pointed out by the authors of the Holoportation system [18], however, although both dynamic mesh [31] and point cloud compressions [82] are being actively explored, it is still currently challenging to achieve real-time compression at the lowest bitrates (less than low tens of Mbps).

In contrast, standard 2D image and video compression techniques are quite mature and enable today's modern 2D video communications over standard wireless networks. If 3D geometry can be efficiently and precisely converted into standard 2D images, existing 2D video communication platforms can be immediately leveraged for low bandwidth 3D video communications. Given this, there has been effort devoted to encode 3D geometry information within regular 2D images [41,44,45,87]. Although this direction is promising, current encoding methods may not be resilient—both in terms of accuracy and efficiency—to direct storage within a video codec. Further, current methods often only focus on encoding 3D geometry information, leaving other

attributes (e.g., texture) to be stored or transmitted separately. This chapter presents a novel method for the efficient and precise encoding of 3D video data and color texture into a regular 2D video format. This chapter also presents how existing 2D video communication platforms may be leveraged for high quality 3D video communication in real-time over standard wireless networks.

6.3 System Overview

As discussed above, existing 3D video communication technologies have limited applications and potentially limited adoption partially because of (1) the required specialized and often expensive hardware; (2) inflexible and complex system setup; (3) highly demanding computational resources for real-time realization; (4) low-resolution and low-accuracy 3D sensing; and/or (5) the required high speed network bandwidths for 3D content transmission due to inefficient 3D video data compression methods.

Our proposed Holostream platform achieved high-resolution and high-accuracy 3D video communication by developing: (1) high-accuracy and high-resolution 3D video capture hardware system; (2) a novel 3D video compression technique; (3) a novel computational framework for 3D video streaming and decompression; and (4) a 3D video visualization application for mobile devices. To our knowledge, this system is the first of its kind which can deliver dense and accurate 3D video content in real time across standard wireless networks to a remote mobile devices (e.g., iPhones and iPads).

Figure 6.1 shows the overall pipeline of the proposed Holostream system. A set of structured patterns are projected and captured by a structured light system to reconstruct dense and accurate 3D video data, including both geometry and color texture information (*Acquisition Module*); 3D video data is encoded frame by frame into a standard 2D image format that is further compressed using a standard 2D video compression technique (*Compression Module*); the compressed video is delivered across standard wireless networks to remote mobile devices (e.g., iPhones) (*Transmission*

Module); the mobile device decompresses the 2D video and decodes the original 3D data frame by frame (*Decompression Module*); and finally the mobile app visualizes 3D video contents in real time allowing the user to instantaneously interact with the 3D video data (*Visualization Module*). The rest of this section describes each individual module developed for the entire pipeline.

6.3.1 Acquisition Module

The Acquisition Module is a structured light scanner that acquires high-resolution 3D video data including 3D geometry and color texture in real time. The scanner consists of a single camera and a single projector, and uses the principle of triangulation for 3D shape reconstruction [88]. Each 3D reconstruction requires a sequence of structured patterns that vary sinusoidally along one direction and remain constant along the other direction. These types of sinusoidal patterns are often called fringe patterns; and this type of structured light technique is called digital fringe projection (DFP). Instead of directly using intensity information for 3D shape reconstruction, the DFP technique extracts the phase of phase-shifted fringe patterns for 3D shape reconstruction. Compared with conventional structured light techniques, the DFP technique has the advantage of simultaneously achieving high resolution (i.e., at camera pixel level) and high speed [89].

Our DFP system uses two sets of patterns with different frequencies, each having three phase-shifted patterns described as,

$$I_k^h(x,y) = I'(x,y) + I''(x,y)\cos\left(\phi^h + 2k\pi/3\right), \tag{6.1}$$

$$I_k^l(x,y) = I'(x,y) + I''(x,y)\cos\left(\phi^l + 2k\pi/3\right). \tag{6.2}$$

Here, $k = \{1,2,3\}$, $I'(x,y)$ is the average intensity, $I''(x,y)$ is the intensity modulation, and $\phi^h(x,y)$ and $\phi^l(x,y)$ is the high frequency phase and low frequency phase,

respectively. Simultaneously solving three equations with the same frequency phase leads to

$$I'(x,y) = I_t^g(x,y) = (I_1^h + I_2^h + I_3^h)/3, \tag{6.3}$$

$$I''(x,y) = \sqrt{3(I_2^h - I_1^h)^2 + (2I_3^h - I_2^h - I_1^h)^2}/3, \tag{6.4}$$

$$\phi^h(x,y) = \tan^{-1}\left[\sqrt{3}\left(I_2^h - I_1^h\right)\right] / \left[2I_3^h - I_2^h - I_1^h\right], \tag{6.5}$$

$$\phi^l(x,y) = \tan^{-1}\left[\sqrt{3}\left(I_2^l - I_1^l\right)\right] / \left[2I_3^l - I_2^l - I_1^l\right]. \tag{6.6}$$

Both low and high frequency phase maps obtained here are called wrapped phase maps since they are bounded by $(-\pi, \pi]$ due to the use of an inverse tangent function. To recover the original phase value, the 2π discontinuous locations need to be identified and corrected; this is done through a step called phase unwrapping. Our system combined these two frequency phase maps and used a two-frequency phase unwrapping method [90] to obtain an unwrapped phase map $\Phi(x,y)$ for the high-frequency fringe patterns. The unwrapped phase is further used to reconstruct (x,y,z) coordinates per pixel once the system parameters (e.g., focal lengths of the camera and the projector, the transformation from the projector coordinate system to the camera coordinate system) are pre-calibrated [79]. Besides acquiring 3D geometry, DFP techniques naturally come with a texture image $I_t^g(x,y)$, and if a single sensor color camera is used, $I_t^g(x,y)$ can be converted to a color image through the demosaicing process. Figure 6.2 shows an example of reconstructing a single 3D frame for a camera resolution of 480×640.

6.3.2 Compression Module

The goal of our 3D video communication is to deliver dense and accurate 3D range geometry information, over existing standard wireless networks, at the video rate the Acquisition Module captures (e.g., 30 Hz). To accomplish this goal, the raw data must be substantially compressed for efficient transmission. Our proposed 3D video compression includes two steps: the first step is to encode 3D geometry and

Fig. 6.2. Example of reconstructing 3D geometry and color texture from the Acquisition Module. Top row left to right: low frequency fringe patterns, high-frequency fringe patterns, low frequency wrapped phase ϕ^l, high-frequency wrapped phase ϕ^h; bottom row left to right: b/w texture I_t^g, unwrapped phase Φ, 3D geometry, color texture after demosaicing I_t^c.

color texture into standard 2D images frame by frame. The second step is then to compress the 2D image sequence using standard video compression codecs.

3D Data Encoding. As previously discussed, (x, y, z) coordinates of a given pixel (i, j) are recovered directly from the unwrapped phase $\Phi(i, j)$ if the system parameters of the scanner are pre-calibrated [79]. In other words, there exists a one-to-one mapping between the unwrapped phase $\Phi(i, j)$ of a point and its recovered (x, y, z) coordinate. Therefore, if 3D coordinates of a pixel are known, we can determine its phase value with known system parameters. This tells us that we can directly use

2D data to represent 3D geometry. For each pixel (i, j) within the unwrapped phase map, a scaled corresponding phase value $\Phi'(i, j)$ is encoded as

$$E_r(i,j) = 0.5 + 0.5 \sin \Phi'(i,j), \tag{6.7}$$

$$E_g(i,j) = 0.5 + 0.5 \cos \Phi'(i,j). \tag{6.8}$$

The third color channel, in this case the blue channel, is used to store the natural texture value, $E_b = I_t^g$. Once the unwrapped phase and texture data are encoded and stored into the output 2D image, E, it can be further compressed frame-by-frame via a lossless (e.g., PNG), or via various levels of lossy (e.g., JPEG), image encoding. The final result of this novel approach is a compressed 2D RGB image from which both 3D coordinates and color texture can be later recovered.

The left image of Fig. 6.3 shows an example of an image, E, which encodes the same 3D geometry and color texture shown in Fig. 6.2. If the 480×640 image is stored with lossless PNG, the file size of the mesh is reduced from 32 MB to 288 KB, achieving a compression ratio of approximately 112:1 versus storing the same information within the OBJ format. As shown in the last image of Fig. 6.3, the difference between the original geometry and the geometry reconstructed from the PNG image, E, appears to be random noise which may be caused by small amounts of quantization.

If a lossy JPEG encoder is used, the file sizes can be reduced even further. Compression ratios of 107:1, 267:1, 406:1, and 518:1 were achieved when using JPEG 100%, 95%, 90%, and 85%, respectively, to compress the geometry within a 480×640 image. The 3D reconstructions from this lossy encoded image incur some reduction in measurement accuracy; however, for visual-based applications (e.g., telepresence), there may not be a distinguishable difference.

3D Video Compression. The compression ratios offered by PNG or JPEG may be high enough to stream E directly frame-by-frame over most medium bandwidth networks, however, even higher compression ratios can be achieved if the frames are video encoded. This is especially advantageous in terms of preparing E for delivery

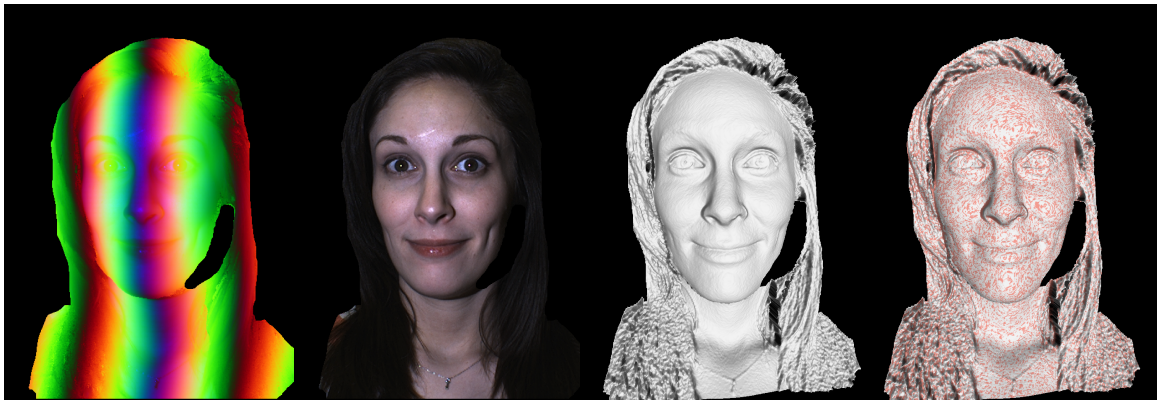


Fig. 6.3. Encoding and decoding 3D geometry and color texture. A lossless PNG format results in a compression ratio of approximately 112:1. Images from left to right respectively show the encoded RGB PNG image, the recovered color texture, the 3D reconstructed geometry, and an overlay of the reconstructed 3D geometry on top of the original 3D geometry (gray color represents recovered geometry and red represents the original geometry).

to mobile devices. In our platform, we encode E into an H.264 video stream using the x264 library [91]; this stream is then segmented into shorter transport stream files for delivery via HTTP Live Streaming (HLS). Figure 6.4 shows some example reconstructions decoded from the data stored using various H.264 qualities. When losslessly encoding a 10 second 3D video sequence consisting of 300 frames, a compression ratio of 129:1 was achieved resulting in 75 MB of data. When using H.264 to lossy encode video at various levels, the color texture was placed in a mosaic fashion to the right of the encoded 3D image, E. Encoding these same 300 3D video frames using 4:2:0 subsampling and a constant rate factor (CRF) of 6 provided a 551:1 compression ratio and a file size of 17.5 MB. Live streaming this encoded video sequence only required a 14 Mbps connection. A CRF value of 12 encoded the sequence into 6 MB of data giving a 1,602:1 compression ratio. As can be seen in Fig. 6.4, even at the very low bitrate of 4.8 Mbps the resulting 3D reconstructions are still of great quality in both geometry and color texture.

Fig. 6.4. Compressing encoded 3D data with the H.264 video codec at different qualities. First row: four frames decoded from the video stored with lossless H.264 (compression ratio 129:1); second row: four frames decoded from H.264 video using 4:2:0 subsampling and a constant rate factor of 6 (compression ratio 551:1); and third row: the corresponding frames using the same encoding instead with a constant rate factor of 12 (compression ratio 1,602:1).

6.3.3 Transmission Module

The Transmission Module was implemented within an intermediary web server—built on top of HTTP and WebSocket technologies—which acts as a middleware between the Acquisition Module and Visualization Module(s). When a client connects to the server, an initialization message, M, is constructed and sent over a WebSocket. This message contains a few important parameters, such as the camera's demosaicing format and resolution, the phase scaling factor, the system's capture volume, and the DFP system's calibration data. Encoded live video streams are delivered over HTTP to connected clients within small video segments (transport streams) via HTTP Live Streaming (HLS).

6.3.4 Decompression Module

To highlight the efficiency of the proposed 3D video encoding method, our primary Decompression Module was implemented within a native iOS application responsible for receiving compressed 2D video data, decompressing it, and then reconstructing 3D geometry and color texture.

Initialization. When a Decompression Module client first connects to the Transmission Module's server, it receives properties about the incoming video stream, including the DFP system's calibration parameters and the video frame dimensions. Using these dimensions, a 2D mesh plane of 3D coordinates is initialized. For example, if the frame dimensions are 480×640, a regular mesh plane consisting of 307,200 vertices will be constructed. This initialization ensures that any pixel within the incoming encoded image, E, corresponds to a single 3D vertex within the mesh. The mesh is also created such that it is modifiable via vertex and fragment GPU shaders. All other properties within the initialization message, M, are sent to the shaders as uniforms.

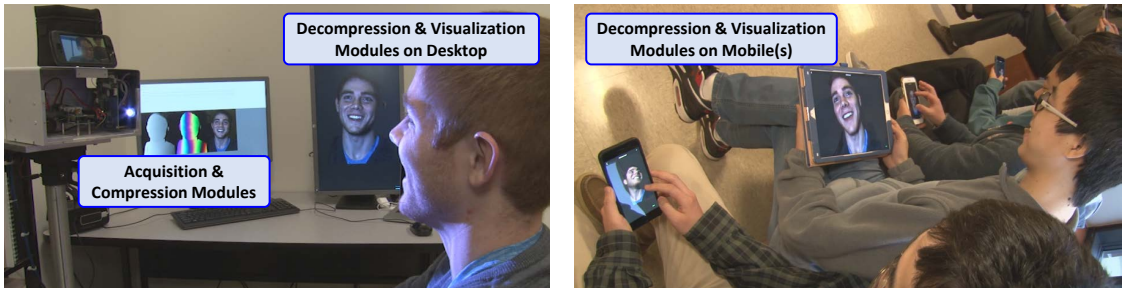


Fig. 6.5. Successful implementation of our proposed Holostream platform. The left image shows the Acquisition and Compression Modules. The right image shows multiple users on their mobile devices receiving and interacting with the live 3D video stream delivered across a standard wireless network. A video demonstration of our system can be found at `http://www.xyztlab.com/holostream-ei-2018`.

Reconstruction. For each vertex (u, v) of the mesh, a wrapped phase value, ϕ_r, can be computed for the data stored within the two chosen color channels of E (e.g., red and green):

$$\phi_r(u, v) = \tan^{-1}\left[(E_r - 0.5)/(E_g - 0.5)\right]. \tag{6.9}$$

The wrapped phase is then unwrapped using the minimum phase unwrapping technology developed by An et al. [71]. This is done on the fly in the vertex shader using the DFP system's calibration parameters, the phase scaling factor, and the minimum z value of the capture volume—all of which were made available to the connected client upon initialization. Once the unwrapped phase $\Phi(u, v)$ has been computed it can be converted into a 3D coordinate (x, y, z) using the calibration parameters. Finally, the vertex (u, v) associated with the vertex shader updates its position attribute to the newly derived 3D coordinate.

The color texture image can also be decoded from the received encoded image, E, and this is done within the mesh's fragment shader. To recover a color texture value, we perform a demosaicing operation for each (u, v), using the grayscale texture encoded in E_b.

6.3.5 Visualization Module

To demonstrate the efficiency of our 3D video encoding method and the wireless nature of our platform, our main Visualization Module was implemented within a native iOS application. The app, upon connecting to a stream, receives an initialization message, M, from the Transmission Module over a WebSocket connection. Using this information, a 3D scene is initialized and a mesh is prepared for modifications, as described above. The H.264 encoded video stream from the Transmission Module is received via HLS. Geometry and color texture is then decompressed from encoded frames within custom Metal shaders. The end result is the ability to visualize and interact with live 3D video content on both iPhones and iPads.

6.4 Results

We developed a demonstration system to verify the performance of our proposed 3D video communication methods. The platform detailed above was implemented with a single DFP 3D capture system on a single PC. The PC, which ran the Acquisition, Compression, and Transmission Modules, had an Intel Core i7 (3.40 GHz) CPU, 16 GB of RAM, and a single NVIDIA GeForce GTX 980 Ti GPU. The DFP capture system consisted of a single camera (PointGrey Grasshopper3 GS3-U3-23S6C) and a single digital-light-processing (DLP) projector (Texas Instruments LightCrafter 4500). The resolution of the camera was set to 480 $\times$ 640 (allowing for up to 307,200 unique 3D coordinates) and the projector's resolution was 912 $\times$ 1140. We used the method developed by Zhang and Huang [79] to obtain the calibration parameters for our DFP system. The Acquisition and Compression Modules were implemented on the GPU within custom CUDA kernels in order to maintain a streamable frame rate of at least 30 (3D) Hz. Figure 6.5 shows two photographs of the successful implementation of our proposed Holostream platform. Note that the left image also shows an additional desktop implementation of the Decompression and Visualization Modules. A video demonstration of our system can be found at `http://www.xyztlab.com/holostream-ei-2018`.

To evaluate the proposed 3D range encoding method, the sequence of 300 3D face data frames (Fig. 6.4) were encoded and stored within H.264 video streams. CRF values of 0, 6, and 12 were used to encode the H.264 videos. Each frame of each video was then decoded using the proposed method and measured against its original reconstructed 3D frame. The mean error between the original and reconstructed geometries across the entire 10 second video sequence (representing the 300 encoded 3D frames) was 0.38 mm, 0.65 mm, and 0.69 mm for CRF values of 0, 6, and 12, respectively. The respective standard deviations were 0.50 mm, 0.51 mm, and 0.54 mm. It should be noted that these measurements excluded large boundary outliers.

To test the limits of our approach, the same 10 second video sequence was encoded with the H.264 codec using a CRF value of 25. Figure 6.6 shows two decoded video frames from this video stream. The 3D geometry data quality is noticeably worse, however, a compression ratio of 16,279:1 was achieved, resulting in only 0.59 MB of data for the 10 second video sequence. This means that only 0.48 Mbps of network bandwidth was required to deliver 300 3D frames at 30 Hz. Even at this heightened compression ratio, the color texture quality remains reasonably high (right two images). The apparent robust nature of the proposed 3D encoding method may make this technology valuable for 3D data delivery to remote locations or to throttled mobile devices.

Fig. 6.6. Compressing encoded 3D data with the H.264 video codec at very lossy (CRF of 25) qualities achieving a 16,279:1 compression ratio.

6.5 Mobile Holostream

The proposed Holostream platform was able to successfully address the long-standing challenge of achieving high-quality, low-bandwidth 3D video communications. To highlight the efficiency of the Holostream platform, while lowering the barrier of entry into 3D communications, this dissertation research also developed a fully mobile Holostream platform. This mobile Holostream platform used an iPad

Pro device equipped with a Structure Sensor (Occipital, Inc.), as shown in Fig. 6.7, to capture 3D geometry information; color texture information was also available via the iPad's rear facing camera. This *Mobile Holostream* platform was able to perform the acquisition, compression, and transmission of 3D range video data, from one iPad device to another, for visualization and interaction. Given that an increasing number of consumer mobile devices now include depth or 3D imaging capabilities, such a platform could not only change how we communicate every day, but it may also enable many new applications, including those in telemedicine and the digital arts.

Fig. 6.7. The Structure Sensor (Occipital, Inc.) attached to an iPad Pro. The Structure can provides the iPad with 640 × 480 depth maps at 30 Hz. In addition, the iPad's color camera can be used to provide color texture information.

The details of the Mobile Holostream platform implementation are similar to the aforementioned Holostream platform. Once 3D geometry data has been acquired by the Structure Sensor connected to the iPad, it is encoded within the color channels of a 2D RGB image using the method presented in Chapter 5. The output 2D images, consisting of the encoded geometry and color texture, are then transmitted within an H.264 video stream to a receiving iPad device via WebRTC (Web Real-

Time Communication). Once received by the remote iPad device, encoded images can be decoded to recover the 3D geometry and color texture information needed for visualization and interaction.

Currently, the Mobile Holostream platform is implemented within a native iOS application. This application interfaces with the Structure Sensor at 30 Hz to retrieve depth maps and color texture images that each have resolutions of 640×480. As mentioned above, this data is then encoded and delivered within an H.264 video stream transmitted via WebRTC with transmission latencies typically between 0.1-2.0 seconds. Once received, the encoded images are decoded pixel-by-pixel on the receiving iPad's GPU for 3D reconstruction and visualization within a virtual 3D environment. If the receiving iPad device were also equipped with a Structure Sensor, the same iOS application could allow for two-way 3D video communications. Figure 6.8 shows two photographs of the Mobile Holostream platform in use.

Fig. 6.8. The Mobile Holostream platform in use. The left image shows the Mobile Holostream platform that uses an iPad Pro and a Structure Sensor to capture, compress, and wirelessly deliver 3D video data of a person in real-time. The right image shows another person using the Mobile Holostream platform on another iPad to receive, decompress, visualize, and interact with the reconstructed 3D video data in real-time.

To evaluate the Mobile Holostream platform, we first transmitted 3D video of a sphere with a 304.8 mm (12 inch) diameter. As the 3D data and color texture

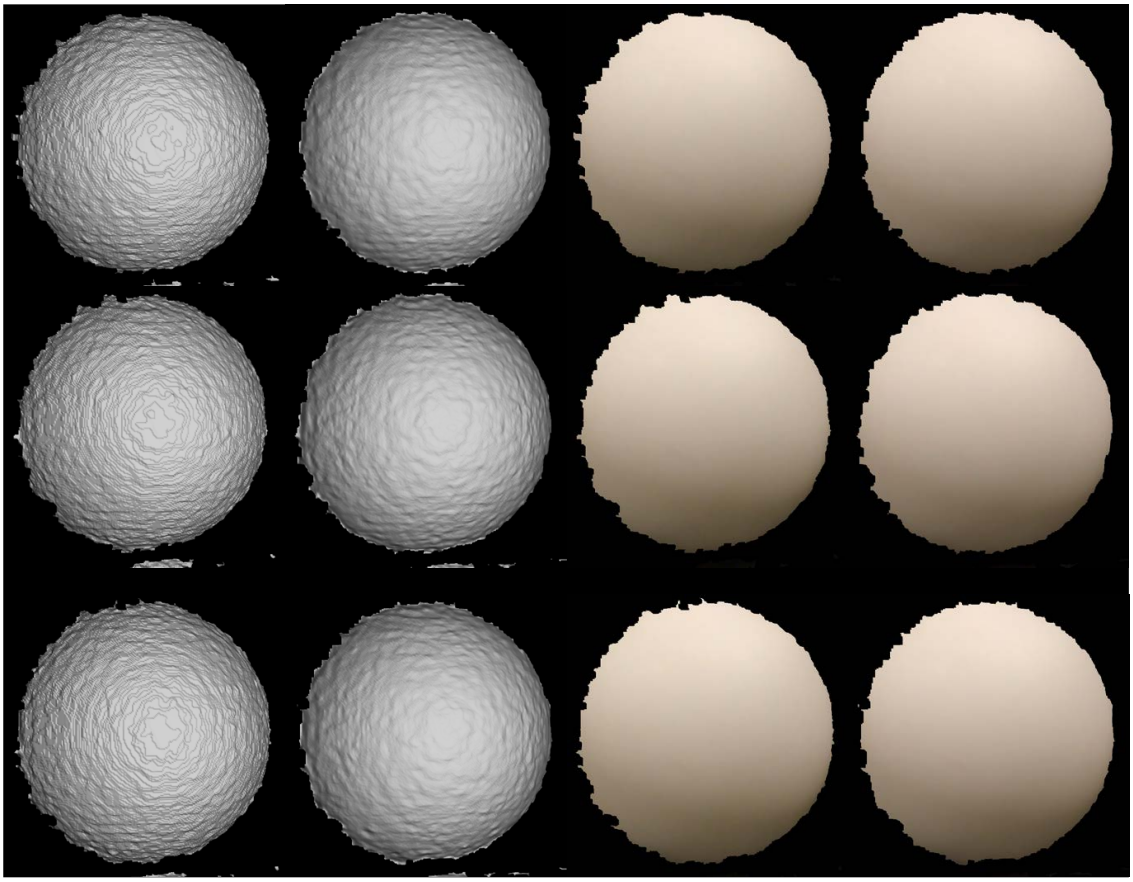


Fig. 6.9. The Mobile Holostream platform was used to capture, compress, and transmit 3D video of a 304.8 mm (12 inch) sphere. Each row shows reconstructions of the sphere at different times. Column one: lossless reconstruction of the 3D data as captured by the sending iPad; column two: 3D data reconstructed from lossy H.264 video received by the receiving iPad; column three: lossless reconstruction of the 3D data as captured by the sending iPad with color texture mapping; column four: 3D data reconstructed from lossy H.264 video with color texture mapping.

of the sphere were being captured, they were encoded into 1280 $\times$ 480 images and transmitted with WebRTC. Figure 6.9 shows reconstructions from several different compressed 3D video frames of the sphere, with each row showing a different frame. The first column shows the lossless reconstruction of the 3D data as captured by the sending iPad; the second column shows the 3D data reconstructed from lossy H.264

video received by the receiving iPad; the third column shows the lossless 3D data with color texture mapping; and the fourth column shows the 3D data reconstructed from lossy H.264 video with color texture mapping. As shown in Fig. 6.9, the spheres reconstructed from lossy H.264 video retain their spherical geometry, and the color texture quality remains reasonably high.

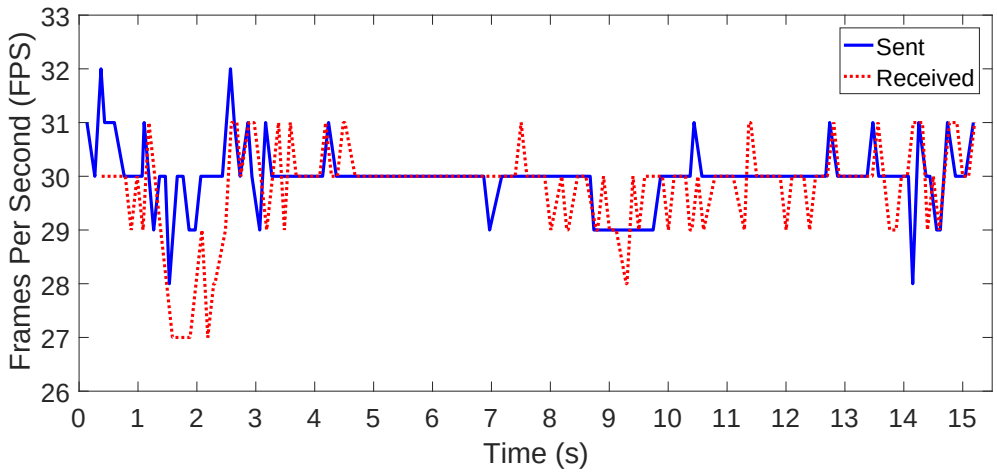


Fig. 6.10. Frames per second (FPS) over time. The encoded FPS transmitted over WebRTC by the sending iPad (blue line) varied from the FPS received and decoded on the receiving iPad (red line). There are many factors that could affect the FPS at each device, including each device's computational load and the bandwidths provided by the wireless network.

When performing further evaluation of the Mobile Holostream platform paired with the Structure Sensor, we found that the encoded frames per second (FPS) transmitted over WebRTC by the sending iPad varied from the FPS received and decoded on the receiving iPad. Visualized in Fig. 6.10, the FPS discrepancy may be attributed to a number of factors, including the current computational load of each device (due to encoding frames, decoding frames, rendering 3D data, etc.) and the current bandwidths provided by the wireless network. One side-effect of this was that performing analysis and comparisons between frames sent by WebRTC and frames received proved to be a difficult task. For instance, a frame transmitted from the send-

ing iPad at any given time will not arrive to the receiving iPad at a predictable time, or it may be dropped altogether. Given this, performing absolute frame-to-frame comparisons between the same frame on either end was a challenge. Future work on the Mobile Holostream platform could explore how to perform such evaluations using WebRTC, overcoming the above obstacles.

Fig. 6.11. The Mobile Holostream platform was used to capture, compress, and transmit 3D video of a dynamic scene representing a 3D video conference. Each row shows reconstructions of the individual at different times. Column one: lossless reconstruction of the 3D data as captured by the sending iPad; column two: 3D data reconstructed from lossy H.264 video received by the receiving iPad; column three: lossless reconstruction of the 3D data as captured by the sending iPad with color texture mapping; column four: 3D data reconstructed from lossy H.264 video with color texture mapping.

To expand upon the evaluation of the sphere mentioned above, we also used the Mobile Holostream platform to transmit 3D video of a dynamic scene portraying an individual participating in a 3D video conference. As the dynamic scene was being captured, frames were encoded into 1280 $\times$ 480 images and transmitted from the sending iPad to the receiving iPad. Figure 6.11 shows several reconstructions of the individual being captured that were taken at different times throughout the 3D video stream. Same as above, the first column of this figure shows the lossless reconstruction of the 3D data as captured by the sending iPad; the second column shows the 3D data reconstructed from lossy H.264 video received by the receiving iPad; the third column shows the lossless 3D data with color texture mapping; and the fourth column shows the 3D data reconstructed from lossy H.264 video with color texture mapping. These results help demonstrate the Mobile Holostream platform's potential to enable new applications for remote communications, collaboration, telepresence, and telemedicine.

6.6 Summary

This chapter presented Holostream, a novel platform for high-quality 3D video communication across existing standard wireless networks and with existing mobile hardware devices (e.g., iPhones, iPads). Our novel 3D video compression method when paired with the H.264 codec achieved compression ratios of 1,602:1 while maintaining high-quality 3D video with color texture. This allowed for transmission of 3D video and color texture content over existing wireless networks using only 4.8 Mbps. We also developed an implementation of our entire 3D video acquisition, encoding, compression, decompression, and visualization framework. This system wirelessly delivered coordinate and color data, consisting of up to 307,200 vertices per frame, at 30 Hz to multiple mobile phones and tablets. We also showed that a bitrate of only 0.48 Mbps was sufficient to deliver lower quality 3D geometry while maintaining high quality color texture. Lastly, we developed a completely mobile Holostream

platform that enabled iPad devices to devices to send and receive 3D video content, acquired via a commercially available structured light scanner, in real-time across wireless networks.

7. SUMMARY AND FUTURE DIRECTIONS

7.1 Summary of Contributions

In pursuit of realizing high-quality, low-bandwidth 3D video communications, this research has made the following contributions:

- **Developed a novel method for out-of-focus camera calibration with applications to large-range structured light system calibration.** The primary contribution of this method was using an LCD monitor to generate fringe patterns that encode the feature points, needed for the accurate calibration of a camera, into the carrier phase. These feature points were able to be decoded, even if the fringe patterns were substantially blurred due to the camera being out-of-focus. Thus, a regular LCD monitor could be used to perform accurate camera calibration, at a close (out-of-focus) range, instead of using a large, difficult to fabricate, and expensive planar object at a far (in-focus) range. The method was shown to accurately calibrate camera intrinsic parameters regardless of the amounts of defocusing. For example, the focal length distance is approximately 0.2% when the camera is focused versus out-of-focus. Further, this method has also been utilized to aid in the calibration of a structured light system, extending its 3D measurement range. These works were each published in the Journal of *Applied Optics* and were discussed in Chapter 3.

- **Developed a method of encoding 3D range geometry, along with its respective texture information, that is highly resilient to lossy compression artifacts.** The primary contribution of this method was utilizing the geometric constraints, inherent to a structured light 3D scanning device, to reduce the amount of data which need be stored within the encoded 2D im-

age. The proposed method was thus able to accurately encode a floating-point unwrapped phase map, representative of 3D geometry information, into a regular 2D image; requiring only two color channels to do so. This left one color channel of the output image free to store additional information, such as the 3D geometry's respective texture information. This method was able to achieve both very high compression ratios (i.e., small file sizes) and high reconstruction accuracies (i.e., low reconstruction error rates). For instance, a compression ratio of 3,038:1 was achieved with JPEG 80, versus the STL format, with an RMS error of 0.47%. This work was published in the Journal of *Applied Optics* and its details were discussed in Chapter 4.

- **Developed a method of encoding 3D range geometry that is highly resilient to lossy compression artifacts that has a computationally inexpensive decoding process.** The primary contribution of this method was encoding fringe order information (necessary for data decoding) within a low frequency, continuous phase map. In terms of intensity, all three channels have a rather smooth profile, and are thus affected less by lossy compression artifacts. This allows the method to achieve both very high compression ratios (i.e., small file sizes) and high reconstruction accuracies (i.e., low reconstruction error rates), even if high levels of lossy JPEG compression are used. For instance, when using this method to encode 3D range data into a 2D image compressed with JPEG 80, a compression ratio of 951:1 was achieved versus the OBJ file format with an error rate of 0.027%. Given these low error rates, little-to-no post-processing of the reconstructed data is necessary, making our method very efficient for real-time implementation on either CPU- or GPU-based software architectures. Lastly, the method's resiliency to lossy image compression enables its natural extension to video encoding, further enabling high-quality 3D video transmission. This work was published in the Journal of *Applied Optics* and its details were introduced in Chapter 5.

- **Developed *Holostream*, a novel platform for high-accuracy, high-speed 3D range video encoding and streaming over wireless networks.** The aforementioned technologies helped make it possible to (1) capture high-resolution and high-quality 3D video data and (2) accurately and efficiently encode the 3D geometry information. With these advancements, we developed the novel Holostream platform for high-quality 3D video recording, encoding, compression, decompression, visualization, and interaction. A demonstration system successfully delivered video-rate photorealistic 3D video content over standard wireless networks to mobile (e.g., iPhone, iPad) devices. Utilizing the above innovations and mature video compression techniques, Holostream was able to deliver high-quality 3D video and color texture data to mobile devices using only 4.8-14 Mbps. Even under extremely poor network conditions, structurally representative 3D geometry and reasonably high-quality texture information could be delivered using only 0.48 Mbps. To emphasize the efficiency of this platform, this dissertation research also developed a fully mobile Holostream platform implementation that enabled mobile iPad devices to both send and receive 3D video content, acquired via a commercially available structured light scanner, in real-time across wireless networks. The majority of this work was published in the 2018 proceedings of *Electronic Imaging* [81] and its details were introduced in Chapter 6.

7.2 Future Directions

This dissertation research has developed technologies contributing toward the development of a high-quality, low-bandwidth 3D video communications platform. With such a platform technology, many new research areas and applications may now be enabled. This section will briefly introduce several future directions in which this dissertation research could be further developed or applied.

7.2.1 Mobile Holostream

As discussed in Chapter 6.5, the Mobile Holostream platform currently has a difference between the encoded frames per second (FPS) transmitted over WebRTC by the sending iPad and the FPS received and decoded on the receiving iPad. There are many factors that can affect this discrepancy, including the computational load of each device and the bandwidths provided by the wireless network. One side-effect of the variability in FPS was that performing absolute frame-to-frame comparisons between the same frame on either end proved to be challenging. Future work on the Mobile Holostream platform could explore how to perform such evaluations using WebRTC. Once absolute comparisons can be made between the same frames on either end, more precise analysis of 3D data reconstructed from compressed video can be conducted. Such analysis could provide insight into new methods for the optimal encoding of lower-quality (i.e., higher-entropy) data provided by today's commercially available 3D imaging devices.

In addition, expanding the Mobile Holostream research to allow multiple users to transmit 3D video information to a one another, or to a single shared environment, could potentially enable new methods of remote collaboration and telepresence. To perform such an extension of this dissertation research, a web server could be used to receive compressed 3D video streams from multiple users. Depending on the desired application, the web server could potentially (1) perform a virtual reconstruction of the environment from the multiple received 3D video streams (if the users are each capturing a different perspective of the same scene); or (2) allow users to create a unique virtual scenes composed of the received 3D streams. Regardless, after the virtual scene has been constructed on the web server, it can be compressed and delivered to receiving devices for visualization in the preferred method (e.g., mobile device, augmented reality, virtual reality). For this extension of the Mobile Holostream platform to be successful, techniques to achieve very low latency 3D video communications to and from the server would be necessary. Any delay in transmitting multiple 3D video

streams or in constructing, compressing, and transmitting a shared virtual environment could cause significant impact to the visualization and interaction capabilities of the platform. If achieved, however, this extension could enable many new applications within collaboration and telepresence. It could also be used to enable applications for live, immersive sporting events and theatre.

7.2.2 Homeland Security

In the context of homeland security, the ability to quickly identify a dangerous or wanted individual, say in an airport or train station, may be critical. To perform this identification, 3D facial recognition could be used, however this may be difficult to do efficiently and accurately due to the data size (both of the captured individual and the data set to perform the recognition against) and available computational resources. Another factor would be how comprehensive (i.e., populated) the data set of individuals to perform the recognition against was, especially since this data set could vary from one venue to the next.

As described in Chapter 6, this dissertation research currently delivers compressed 3D video to a web server for transmission to receiving device. Currently, this web server acts as a simple relay, receiving the compressed data from a sending device and transmitting it to a receiving device. This research could potentially be extended, however, to perform analysis on the 3D data before transmitting it to a receiving device. Regarding the application of homeland security, performing facial recognition on the compressed 3D video received by the web server could overcome some of the aforementioned challenges. First, performing the analysis on the compressed data could reduce the size and complexity of the data to be recognized and the data set to perform the recognition against. Second, if the recognition task were performed remotely on distributed web servers, this could potentially allow for faster recognition times and could increase the comprehensiveness of the data set (many venues could share one large, distributed data set to perform the recognition against). Upon re-

ceiving the compressed 3D data and performing recognition, the identification results could be delivered back to the venue in which the data was taken, possibly providing information on if an individual was on a wanted list or is potentially dangerous, for example.

7.2.3 Telemedicine

As described in detail in Chapter 1 and 2, one application area that could benefit from the contributions of this dissertation research is telemedicine. Current 2D technologies do not allow for natural and interactive communications, for accurate monitoring, or for measurements to be taken for assessment. The high-quality, wireless, and potentially mobile 3D video communication platform proposed by this dissertation may be able to overcome these challenges, however, especially as an increasing number of consumer mobile devices are equipped with 3D sensors. Given this, medical patients would ideally be able to use their personal devices to receive care via a 3D telemedicine application.

To advance upon this dissertation research, novel methods of encoding low-quality (i.e., high-entropy) 3D geometry data gathered from consumer devices could be investigated. Further, the introduction of a new technology like 3D video telemedicine to the medical field will require developing new best-practices for providing the best care with it. To address this, novel methods of visualizing and interacting with transmitted 3D video data could be investigated. Visualizing 3D video telemedicine data within augmented and virtual reality environments, for example, could lead to new best-practices for remote and collaborative medical care. In addition, since such a 3D telemedicine system would potentially be transmitting sensitive medical information, methods for 3D data encryption and security could be explored.

Lastly, this dissertation research's ability to accurately capture and transmit 3D video data could potentially be used to extend the capabilities of remote surgery. One or more high-quality 3D scanning devices could be used to capture real-time

3D video of a surgery as its being performed. Now, consider there is a more experienced, or expert, surgeon who practices within a hospital of another country. This dissertation research could be used to deliver 3D video of the surgery to the remote surgeon in real-time. The remote surgeon could then visualize or perform analysis on the surgery within augmented or virtual reality, providing immediate feedback and guidance to the surgeon(s) performing the operation. Next, consider minimally invasive robotic surgery systems (e.g., the *da Vinci Surgical System* [92]) that allow surgeons to perform operations comfortably using peripheral devices connected to a computer. If such surgical systems are equipped with one or more high-quality 3D scanning devices, this dissertation research could be used to deliver 3D video data of the surgery to this computer, potentially enabling the surgeon to perform operations from a remote location.

REFERENCES

REFERENCES

[1] Association of American Medical Colleges, Center for Workforce Studies, *Recent Studies and Reports on Physician Shortages in the U.S.* Association of American Medical Colleges, 2012.

[2] IHS Markit, *The Complexities of Physician Supply and Demand 2017 Update: Projections from 2015 to 2030.* Association of American Medical Colleges, 2017.

[3] Wiktionary, "underserved," https://en.wiktionary.org/wiki/underserved.

[4] J. Kvedar, M. J. Coye, and W. Everett, "Connected health: A review of technologies and strategies to improve patient care with telemedicine and telehealth," *Health Affairs*, vol. 33, no. 2, pp. 194–199, 2014.

[5] S. D. Anker, F. Koehler, and W. T. Abraham, "Telemedicine and remote management of patients with heart failure," *The Lancet*, vol. 378, no. 9792, pp. 731 – 739, 2011.

[6] Y. A. A. Tout, C. Taleb, M. Kosayer, and P. Liverneaux, "Telemedicine and hand injuries: A feasibility study," *Annales de Chirurgie Plastique Esthétique*, vol. 55, no. 1, pp. 8 – 13, 2010.

[7] S. Gardiner and T. L. Hartzell, "Telemedicine and plastic surgery: A review of its applications, limitations and legal pitfalls," *Journal of Plastic, Reconstructive & Aesthetic Surgery*, vol. 65, no. 3, pp. e47 – e53, 2012.

[8] C. Chakraborty, B. Gupta, S. K. Ghosh, D. K. Das, and C. Chakraborty, "Telemedicine supported chronic wound tissue prediction using classification approaches," *Journal of Medical Systems*, vol. 40, no. 3, p. 68, 2016.

[9] F. L. Bowling, L. King, J. A. Paterson, J. Hu, B. A. Lipsky, D. R. Matthews, and A. J. Boulton, "Remote assessment of diabetic foot ulcers using a novel wound imaging system," *Wound Repair and Regeneration*, vol. 19, no. 1, pp. 25–30, 2011.

[10] B. S. B. Rasmussen, J. Froekjaer, L. B. Joergensen, U. Halekoh, and K. B. Yderstraede, "Validation of a new imaging device for telemedical ulcer monitoring," *Skin Research and Technology*, vol. 21, no. 4, pp. 485–492, 2015.

[11] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs, "The office of the future: A unified approach to image-based modeling and spatially immersive displays," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. New York, NY: ACM, 1998, pp. 179–188.

[12] H. Towles, W. chao Chen, R. Yang, S. uok Kum, H. F. N. Kelshikar, J. Mulligan, K. Daniilidis, H. Fuchs, C. C. Hill, N. K. J. Mulligan, L. Holden, B. Zeleznik, A. Sadagic, and J. Lanier, "3d tele-collaboration over internet2," in *International Workshop on Immersive Telepresence, Juan Les Pins*, 2002.

[13] M. Gross, S. Würmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. Van Gool, S. Lang, K. Strehlke, A. V. Moere, and O. Staadt, "Blue-c: A spatially immersive display and 3d video portal for telepresence," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 819–827, 2003.

[14] S. Zhang and P. S. Huang, "High-resolution real-time three-dimensional shape measurement," *Opt. Eng.*, vol. 45, no. 12, p. 123601, 2006.

[15] A. Jones, M. Lang, G. Fyffe, X. Yu, J. Busch, I. McDowall, M. Bolas, and P. Debevec, "Achieving eye contact in a one-to-many 3d video teleconferencing system," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 64:1–64:8, 2009.

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

[17] S. Beck, A. Kunert, A. Kulik, and B. Froehlich, "Immersive group-to-group telepresence," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 4, pp. 616–625, 2013.

[18] S. Orts-Escolano, C. Rhemann, S. Fanello, W. Chang, A. Kowdle, Y. Degtyarev, D. Kim, P. L. Davidson, S. Khamis, M. Dou, V. Tankovich, C. Loop, Q. Cai, P. A. Chou, S. Mennicken, J. Valentin, V. Pradeep, S. Wang, S. B. Kang, P. Kohli, Y. Lutchyn, C. Keskin, and S. Izadi, "Holoportation: Virtual 3d teleportation in real-time," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ser. UIST '16. New York, NY: ACM, 2016, pp. 741–754.

[19] M. Deering, "Geometry compression," in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '95. New York, NY, USA: ACM, 1995, pp. 13–20.

[20] M. M. Chow, "Optimized geometry compression for real-time rendering," in *Proceedings of the 8th Conference on Visualization '97*, ser. VIS '97. Los Alamitos, CA, USA: IEEE Computer Society Press, 1997, pp. 347–354.

[21] C. L. Bajaj, V. Pascucci, and G. Zhuang, "Single resolution compression of arbitrary triangular meshes with properties," *Computational Geometry*, vol. 14, no. 1–3, pp. 167–186, 1999.

[22] G. Taubin and J. Rossignac, "Geometric compression through topological surgery," *ACM Trans. Graph.*, vol. 17, no. 2, pp. 84–115, 1998.

[23] C. Touma and C. Gotsman, "Triangle mesh compression," in *Proceedings of the Graphics Interface 1998*, 1998, pp. 26–34.

[24] P. Alliez and M. Desbrun, "Valence-driven connectivity encoding for 3d meshes," *Computer Graphics Forum*, vol. 20, no. 3, pp. 480–489, 2001.

[25] S. Gumhold and W. Straßer, "Real time compression of triangle mesh connectivity," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. New York, NY: ACM, 1998, pp. 133–140.

[26] J. Rossignac, "Edgebreaker: connectivity compression for triangle meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 1, pp. 47–61, 1999.

[27] J. Peng, C.-S. Kim, and C.-C. J. Kuo, "Technologies for 3d mesh compression: A survey," *Journal of Visual Communication and Image Representation*, vol. 16, no. 6, pp. 688 – 733, 2005.

[28] A. Maglo, G. Lavoué, F. Dupont, and C. Hudelot, "3d mesh compression: Survey, comparisons, and emerging trends," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 44:1–44:41, 2015.

[29] B. Kronrod and C. Gotsman, "Optimized compression of triangle mesh geometry using prediction trees," in *Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission*, 2002, pp. 602–608.

[30] R. Mekuria, M. Sanna, E. Izquierdo, D. C. A. Bulterman, and P. Cesar, "Enabling geometry-based 3-d tele-immersion with fast mesh compression and linear rateless coding," *IEEE Transactions on Multimedia*, vol. 16, no. 7, pp. 1809–1820, 2014.

[31] A. Collet, M. Chuang, P. Sweeney, D. Gillett, D. Evseev, D. Calabrese, H. Hoppe, A. Kirk, and S. Sullivan, "High-quality streamable free-viewpoint video," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 69:1–69:13, 2015.

[32] E. Darakis and J. J. Soraghan, "Compression of interference patterns with application to phase-shifting digital holography," *Appl. Opt.*, vol. 45, no. 11, pp. 2437–2443, 2006.

[33] ——, "Reconstruction domain compression of phase-shifting digital holograms," *Appl. Opt.*, vol. 46, no. 3, pp. 351–356, 2007.

[34] M. Burrows and D. J. Wheeler, "A block-sorting lossless data compression algorithm," Digital Systems Research Center, Palo Alto, California, Tech. Rep., 1994.

[35] Y. Xing, B. Pesquet-Popescu, and F. Dufaux, "Compression of computer generated hologram based on phase-shifting algorithm," in *European Workshop on Visual Information Processing (EUVIP)*, 2013, pp. 172–177.

[36] ——, "Compression of computer generated phase-shifting hologram sequence using avc and hevc," *Proc. SPIE*, vol. 8856, pp. 88 561M–88 561M–8, 2013.

[37] T. Nishitsuji, T. Shimobaba, T. Kakue, and T. Ito, "Fast calculation techniques for computer-generated holograms," in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, 2016, pp. 550–555.

[38] T. Shimobaba, T. Ito, N. Masuda, Y. Ichihashi, and N. Takada, "Fast calculation of computer-generated-hologram on amd hd5000 series gpu and opencl," *Opt. Express*, vol. 18, no. 10, pp. 9955–9960, 2010.

[39] P. Tsang, W.-K. Cheung, T.-C. Poon, and C. Zhou, "Holographic video at 40 frames per second for 4-million object points," *Opt. Express*, vol. 19, no. 16, pp. 15 205–15 211, 2011.

[40] J. Weng, T. Shimobaba, N. Okada, H. Nakayama, M. Oikawa, N. Masuda, and T. Ito, "Generation of real-time large computer generated hologram using wavefront recording method," *Opt. Express*, vol. 20, no. 4, pp. 4018–4023, 2012.

[41] N. Karpinsky and S. Zhang, "Composite phase-shifting algorithm for three-dimensional shape compression," *Opt. Eng.*, vol. 49, no. 6, p. 063604, 2010.

[42] ——, "Holovideo: Real-time 3d video encoding and decoding on gpu," *Opt. Laser Eng.*, vol. 50, no. 2, pp. 280–286, 2012.

[43] ——, "3d range geometry video compression with the h.264 codec," *Opt. Laser Eng.*, vol. 51, no. 5, pp. 620–625, 2013.

[44] Y. Wang, L. Zhang, S. Yang, and F. Ji, "Two-channel high-accuracy holoimage technique for three-dimensional data compression," *Opt. Laser Eng.*, vol. 85, pp. 48–52, 2016.

[45] Z. Hou, X. Su, and Q. Zhang, "Virtual structured-light coding for three-dimensional shape data compression," *Opt. Laser Eng.*, vol. 50, no. 6, pp. 844–849, 2012.

[46] S. Zhang, "Three-dimensional range data compression using computer graphics rendering pipeline," *Appl. Opt.*, vol. 51, no. 18, pp. 4058–4064, 2012.

[47] P. Ou and S. Zhang, "Natural method for three-dimensional range data compression," *Appl. Opt.*, vol. 52, no. 9, pp. 1857–1863, 2013.

[48] T. Bell, J. Xu, and S. Zhang, "Method for out-of-focus camera calibration," *Appl. Opt.*, vol. 55, no. 9, pp. 2346–2352, 2016.

[49] C. B. Duane, "Close-range camera calibration," *Photogrammetric engineering*, vol. 37, no. 8, pp. 855–866, 1971.

[50] I. Sobel, "On calibrating computer controlled cameras for perceiving 3-d scenes," *Artificial Intelligence*, vol. 5, no. 2, pp. 185–198, 1974.

[51] R. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 4, pp. 323–344, 1987.

[52] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000.

[53] J. Lavest, M. Viala, and M. Dhome, "Do we really need an accurate calibration pattern to achieve a reliable camera calibration?" in *Computer Vision ECCV'98*. Springer, 1998, pp. 158–174.

[54] A. Albarelli, E. Rodolà, and A. Torsello, "Robust camera calibration using inaccurate targets," *Trans. Pattern Anal. Mach. Intell*, vol. 31, no. 2, pp. 376–383, 2009.

[55] K. H. Strobl and G. Hirzinger, "More accurate pinhole camera calibration with imperfect planar target," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1068–1075.

[56] L. Huang, Q. Zhang, and A. Asundi, "Flexible camera calibration using not-measured imperfect target," *Applied optics*, vol. 52, no. 25, pp. 6278–6286, 2013.

[57] C. Schmalz, F. Forster, and E. Angelopoulou, "Camera calibration: active versus passive targets," *Optical Engineering*, vol. 50, no. 11, pp. 113 601–113 601, 2011.

[58] L. Huang, Q. Zhang, and A. Asundi, "Camera calibration with active phase target: improvement on feature detection and optimization," *Optics letters*, vol. 38, no. 9, pp. 1446–1448, 2013.

[59] W. Li and Y. F. Li, "Single-camera panoramic stereo imaging system with a fisheye lens and a convex mirror," *Opt. Express*, vol. 19, no. 7, pp. 5855–5867, 2011.

[60] B. Li, N. Karpinsky, and S. Zhang, "Novel calibration method for structured-light system with an out-of-focus projector," *Appl. Opt.*, vol. 53, no. 16, pp. 3415–3426, 2014.

[61] S. Zhang, "Flexible 3d shape measurement using projector defocusing: Extended measurement range," *Opt. Lett.*, vol. 35, no. 7, pp. 931–933, 2010.

[62] L. Yong, "A correspondence finding method based on space conversion in 3d shape measurement using fringe projection," *Opt. Express*, vol. 23, no. 11, pp. 14 188–14 202, 2015.

[63] L. Ekstrand and S. Zhang, "Three-dimensional profilometry with nearly focused binary phase-shifting algorithms," *Opt. Lett.*, vol. 36, no. 23, pp. 4518–4520, 2011.

[64] Y. An, T. Bell, B. Li, J. Xu, and S. Zhang, "Method for large-range structured light system calibration," *Appl. Opt.*, vol. 55, no. 33, pp. 9563–9572, 2016.

[65] T. Bell, B. Vlahov, J. P. Allebach, and S. Zhang, "Three-dimensional range geometry compression via phase encoding," *Appl. Opt.*, vol. 56, no. 33, pp. 9285–9292, 2017.

[66] S. Zhang, "Recent progresses on real-time 3-d shape measurement using digital fringe projection techniques," *Opt. Laser Eng.*, vol. 48, no. 2, pp. 149–158, 2010.

[67] A. Alfalou and C. Brosseau, "Optical image compression and encryption methods," *Adv. Opt. Photon.*, vol. 1, no. 3, pp. 589–636, 2009.

[68] F. Dufaux, Y. Xing, B. Pesquet-Popescu, and P. Schelkens, "Compression of digital holographic data: an overview," *Proc. SPIE*, vol. 9599, pp. 95 990I–95 990I–11, 2015.

[69] T. Bell and S. Zhang, "Multiwavelength depth encoding method for 3d range geometry compression," *Appl. Opt.*, vol. 54, no. 36, pp. 10 684–10 691, 2015.

[70] N. Karpinsky, Y. Wang, and S. Zhang, "Three-bit representation of three-dimensional range data," *Appl. Opt.*, vol. 52, no. 11, pp. 2286–2293, 2013.

[71] Y. An, J.-S. Hyun, and S. Zhang, "Pixel-wise absolute phase unwrapping using geometric constraints of structured light system," *Opt. Express*, vol. 24, no. 15, pp. 18 445–18 459, 2016.

[72] R. Hunt, *The Reproduction of Colour*, 6th ed. John Wiley & Sons, Ltd, 2005.

[73] *Still-image compression – JPEG-1 Extensions. Information Technology – Digital Compression and Coding of Continuous-tone Still Images: JPEG File Interchange Format (JFIF)*, Telecommunication Standardization Sector of ITU (ITU-T), iTU-T Recommendation T.871.

[74] X. Gu, S. Zhang, L. Zhang, P. Huang, R. Martin, and S.-T. Yau, "Holoimages," in *ACM Solid and Physical Modeling*, UK, 2006, pp. 129–138.

[75] T. Bell, N. Karpinsky, and S. Zhang, *High-resolution, real-time 3D sensing with structured light techniques*, ser. Interactive Displays: Natural Human-Interface Technologies. Hoboken, NJ: John Wiley & Sons, 2014, ch. 5, pp. 181–213.

[76] H. Schreiber and J. H. Bruning, "Phase shifting interferometry," in *Optical Shop Testing*. Wiley, 2007, pp. 547–666.

[77] J. Novák, P. Novák, and A. Mikš, "Multi-step phase-shifting algorithms insensitive to linear phase shift errors," *Optics Communications*, vol. 281, no. 21, pp. 5302 – 5309, 2008.

[78] D. C. Ghiglia and M. D. Pritt, *Two-dimensional phase unwrapping: theory, algorithms, and software*. Wiley New York:, 1998, vol. 4.

[79] S. Zhang and P. S. Huang, "Novel method for structured light system calibration," *Opt. Eng.*, vol. 45, no. 8, p. 083601, 2006.

[80] Y. Wang and S. Zhang, "Superfast multifrequency phase-shifting technique with optimal pulse width modulation," *Opt. Express*, vol. 19, no. 6, pp. 5143–5148, 2011.

[81] T. Bell, J. P. Allebach, and S. Zhang, "Holostream: High-accuracy, high-speed 3d range video encoding and streaming across standard wireless networks," in *2018 IS&T International Symposium on Electronic Imaging*, Burlingame, California, 2018, pp. 3DIPM–425.1–3DIPM–425.6.

[82] R. L. de Queiroz and P. A. Chou, "Compression of 3d point clouds using a region-adaptive hierarchical transform," *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3947–3956, 2016.

[83] T. Golla and R. Klein, "Real-time point cloud compression," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 5087–5092.

[84] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 778–785.

[85] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 169:1–169:11, 2013.

[86] A. Mossel and M. Kroeter, "Streaming and exploration of dynamically changing dense 3d reconstructions in immersive virtual reality," in *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*, 2016, pp. 43–48.

[87] A. Bletterer, F. Payan, M. Antonini, and A. Meftah, "Point cloud compression using depth maps," in *IS&T International Symposium on Electronic Imaging 2016*, 2016, pp. 3DIPM–397.1–3DIPM–397.6.

[88] J. Salvi, S. Fernandez, T. Pribanic, and X. Llado, "A state of the art in structured light patterns for surface profilometry," *Patt. Recogn.*, vol. 43, no. 8, pp. 2666–2680, 2010.

[89] J. Geng, "Structured-light 3d surface imaging: a tutorial," *Advances in Opt. and Photonics*, vol. 3, no. 2, pp. 128–160, 2011.

[90] J.-S. Hyun and S. Zhang, "Enhanced two-frequency phase-shifting method," *Appl. Opt.*, vol. 55, no. 16, pp. 4395–4401, 2016.

[91] VideoLAN Organization, "x264," http://www.videolan.org/developers/x264.html.

[92] da Vinci Surgery, "The da vinci surgical system," http://www.davincisurgery.com/da-vinci-surgery/da-vinci-surgical-system.

VITA

VITA

Tyler Bell is a Ph.D. candidate at Purdue University in West Lafayette, Indiana. He received his B.S. and M.S. degrees from Iowa State University, where he received a Research Excellence Award. His research interests include 3D video communications; high-speed, high-resolution 3D imaging; virtual and augmented reality; human computer interaction; 3D data and 3D video compression; and multimedia on mobile devices. By the time of graduation, he has published 13 research articles, including seven journal papers and two book chapters, and has filed two patent applications. Tyler's research on 3D video communications, *Holostream*, was featured in numerous public media articles and was an *Innovation of the Year Award* finalist at the 2018 TechPoint Mira Awards.

LIST OF PUBLICATIONS

LIST OF PUBLICATIONS

Book Chapters

[BC2] T. Bell, B. Li, and S. Zhang, "Structured light techniques and applications," *Wiley Encyclopedia of Electrical and Electronics Engineering*, 1-24, 2016.

[BC1] T. Bell, N. Karpinsky, and S. Zhang, "Real-Time 3D Sensing with Structured Light Techniques," *Interactive Displays: Natural Human-Interface Technologies*, Chapter 5, Edited by A.K. Bhowmik, 2014.

Journal Articles

[J7] T. Bell, B. Vlahov, J.P. Allebach, and S. Zhang, "Three-dimensional range geometry compression via phase encoding," *Appl. Opt.* 56(33), 9285-9292, 2017.

[J6] B. Li, T. Bell, and S. Zhang, "Computer-aided-design-model-assisted absolute three-dimensional shape measurement," *Appl. Opt.* 56(24), 6770-6776, 2017.

[J5] Y. An*, T. Bell*, B. Li, J. Xu, and S. Zhang, "Method for large-range structured light system calibration," *Appl. Opt.* 55(33), 9563-9572, 2016. (* denotes co-first authors)

[J4] C. Jiang, T. Bell and S. Zhang, "High dynamic range real-time 3D shape measurement," *Opt. Express* 24(7), 7337-7346, 2016.

[J3] T. Bell, J. Xu, and S. Zhang, "Method for out-of-focus camera calibration," *Appl. Opt.* 55(9), 2346-2352, 2016.

[J2] T. Bell and S. Zhang, "Multiwavelength depth encoding method for 3D range geometry compression," *Appl. Opt.* 54(36), 10684-10691, 2015.

[J1] T. Bell and S. Zhang, "Towards superfast three-dimensional optical metrology with digital micro mirror device (DMD) platforms," *Opt. Eng.* 53(11), 112206, 2014.

Conference Papers

[C4] T. Bell and S. Zhang, "High-resolution 3D optical sensing and real-time 3D video data streaming," IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Auckland, New Zealand, Jul. 2018.

[C3] T. Bell, J.P. Allebach, and S. Zhang, "Holostream: High-Accuracy, High-Speed 3D Range Video Encoding and Streaming Across Standard Wireless Networks," *2018 IS&T International Symposium on Electronic Imaging*, 3D Image Processing, Measurement (3DIPM), and Applications, Burlingame, CA, Jan. 2018.

[C2] T. Bell and S. Zhang, "A comparative study on 3D range data compression methods," SPIE Defense and Commercial Sensing, *Proc. SPIE 9868*, Dimensional Optical Metrology and Inspection for Practical Applications V, Baltimore, MD, Apr. 2016.

[C1] T. Bell and S. Zhang, "Towards superfast 3D optical metrology with digital micromirror device (DMD) platforms," *Proc. SPIE 8979*, Emerging Digital Micromirror Device Based Systems and Applications VI, San Francisco, CA, Feb. 2014.

Patents

[P2] S. Zhang and T. Bell, "System Architecture and Method of Processing Data Therein," U.S. Patent Application No: 62/651,356 (Pending).

[P1] S. Zhang and T. Bell, "Method and System for Multi-Wavelength Depth Encoding for Three-Dimensional Range Geometry Compression," U.S. Patent Application No: 15/367,221 (Pending).