UC Berkeley
UC Berkeley Electronic Theses and Dissertations

Title
Dynamic Geometry Capture with a Multi-View Structured-Light System

Permalink
https://escholarship.org/uc/item/9t82s33z

Author
Garcia, Ricardo Rafael

Publication Date
2014

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Dynamic Geometry Capture with a Multi-View Structured-Light System

by

Ricardo Rafael Garcia

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Avideh Zakhor, Chair
Professor Jonathan Shewchuk
Professor Martin Banks

Fall 2014

Dynamic Geometry Capture with a Multi-View Structured-Light System

Copyright 2014
by
Ricardo Rafael Garcia

Abstract

Dynamic Geometry Capture with a Multi-View Structured-Light System

by

Ricardo Rafael Garcia

Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Avideh Zakhor, Chair

Human Motion capture has been an active area of research for many years and has applications in many fields such as gaming, entertainment, physical therapy, and ergonomics. Most commercially available motion capture systems use numerous markers to be placed on the body of the human subject requiring a significant setup time. In this dissertation, we develop the architecture and algorithms for markerless motion capture with a multi-view structured light system. In contrast to existing markerless approaches that use multiple camera streams, we reconstruct the scene by combining the views from three structured light stations using sinusoidal phase shift patterns, each equipped with one projector, a stereo pair of cameras for phase unwrapping, and a color camera. The three stations surround the subject and are time multiplexed to avoid interference. Phase-shifted sinusoidal patterns offer low decoding complexity, require as few as three projection frames per reconstruction, and are well suited for capturing dynamic scenes. In these systems, depth is reconstructed by determining the phase projected onto each pixel in the camera and establishing correspondences between camera and projector pixels. Typically, multiple periods are projected within the set of sinusoidal patterns, thus requiring phase unwrapping on the phase image before correspondences can be established.

There are three novel contributions to this dissertation; first, we present a novel phase unwrapping algorithm across space and time in order to generate a temporally consistent point cloud. Specifically, we combine a quality guided phase unwrapping approach with absolute phase estimates from the stereo cameras to solve for the absolute phase of connected regions. Second, we develop a calibration method for multi-camera-projector systems in which sensors face each other as well as share a common viewpoint. We use a translucent planar sheet framed in PVC piping as a calibration target which is placed at multiple positions and orientations within a scene. In each position, the target is captured by the cameras while it is being illuminated by a set of patterns from various projectors. The translucent sheet allows the projected patterns to be visible from both sides, allowing correspondences between devices that face each other. The set of correspondences generated between the devices using this target are input into a bundle adjustment framework to estimate calibration parameters. Third, we develop algorithms to reconstruct dynamic geometry of a human subject using a template generated by the system itself. Specifically, we deform the

template to each frame of the captured geometry by iteratively aligning each bone of the skeleton. This is done by searching for correspondences between the source template and the captured geometry, solving for rotation of bones, and enforcing constraints on each rotation to prevent the template from taking on anatomically unnatural poses. Once the geometry of the dynamic mesh is reconstructed, the template is textured using the color cameras from the multi-view structured-light system. We demonstrate the effectiveness of our approach both qualitatively and quantitatively for an actual sequence of a moving human subject by synthesizing arbitrary views of the dynamic scene.

To Ale, Rafael, Susanna, and Andrea.

Contents

List of Figures

Acknowledgments

During my time in Berkeley, I have been very fortunate to be surrounded by so many wonderful people. The relationships and friendships I have developed during my academic journey have meant so much to me. The support from those around me has been a large contribution to helping me complete my dissertation. I would first like to thank my advisor, Professor Avideh Zakhor. As an advisor, Avideh brings excitement and energy to research that encourages her students to push themselves everyday. In working with Avideh, I have seen myself grow in my ability to produce quality research and bring rigor to my research and writing. Her mentorship and guidance will serve as a positive influence in my future professional and intellectual journeys.

I would also like to thank Professors Martin Banks, Jonathan Shewchuk, and Ravi Ramamoorthi who served as members of my qualifying and dissertation committees. During my dissertation, they provided me with valuable support and conversations that helped define my research direction. I am very grateful for their guidance and support.

The funding for my dissertation was partly enabled through multiple grants including: AFOSR FA9550-08-1-0168, ARO W911NF-07-1-047, ARO W911NF-11-1-0088. The funding from these grants provided me with system hardware and personal financial support that in turn have helped make this dissertation possible. Additionally, I would like to thank Sheila Humphreys, Director of Diversity in EECS, who helped me find additional funding through the College of Engineering and university. She went above any beyond to ensure funding was never a problem during my studies.

One of the best parts of my time at UC Berkeley has been working with colleagues within the Video and Image Processing Lab. I would like to thank John Kua, Michael Krishnan, Shangliang Jiang, Nicholas Corso, Eric Turner, Richard Zhang, and Shicong Yang for their willingness to share their academic expertise, but more importantly for their friendship. My day-to-day work life was brightened by my interactions with each member of the lab. I will miss being part of such a great team. Beyond my lab, I was also able to make many great friends by being a part of the EECS family. I would particularly like to recognize Nikhil Naikal, Matthew Spencer, Achintya Madduri, Amit Lakhani, Dan Calderone, Gireeja Ranade, and Maryam Vareth. Additionally, I would like to thank Shirley Salanio, Associate Director of Graduate Matters. She continually went beyond her job descriptions to connect with me as an individual and help guide me through my final steps of graduation.

Outside of EECS, I also was able meet wonderful folks as a member of the Berkeley community. Whether through time spent in Toastmasters or playing squash, I was able to find support, mentoring, and friendship through so many wonderful people. In particular, I would like to thank Charles Browning, Andre Lewis, Nancy Tran, and Amy Honigman for the constant and continual support. These people helped keep me grounded even during the busiest and most stressful times.

Despite leaving my home state of Texas to come to Berkeley, my family has always been supportive of my decision to pursue a PhD. With my father, Rafael Garcia, serving as the best role model for what it means to be an engineer, I know it is largely through his influence that I chose a career in engineering. My mother and sister, Susanna Garcia and Andrea Garcia, while not engineers, have always let me know they believe in me and have shown their support every day.

Finally, I would especially like to thank my girlfriend, Alejandra Figueroa-Clarevega. Ale has managed to be incredibly supportive as I have wrapped up my dissertation despite seeing me less and less during the final months of work. Her support after long days in lab have been a blessing and I grateful to have a partner who always supports my work and my career.

It is amazing that the decision of selecting a graduate program years ago has shaped my life so drastically. I happy to have been enriched with great academics and great relationships during my time at UC Berkeley. Thank you to everyone who helped make this possible.

Chapter 1

Introduction

The rich media applications and experiences we encounter in our daily lives demand highly detailed and realistic content. This is especially true for applications such as gaming that integrate the 3D shape and movement of human subjects into new media. Systems that scan real-world objects or capture the movements of a human subject for applications in gaming and movies all attempt to improve the ease and accuracy with which reconstructions are captured. Beyond media applications, dynamic geometry of humans is also used in scientific and medical applications such as quantifying improvement in physical therapy and measuring unnatural poses in ergonomic studies. Specifically, dynamic geometry is a representation of the surface shape and position of a moving object over time. In all these applications, there is a push to improve the accuracy in which the pose and shape of the human subject are captured. This requires developing systems that can capture 3D geometry well.

A significant amount of work has focused on systems that generate accurate 3D models of real world scenes; these include laser scanners, stereo-camera systems, time-of-flight cameras, and structured-light systems [16]. Some of the methods are not well suited for capturing 3D models of dynamic scenes. Laser scanners, for example, have proven to be expensive and do not have the spatial or temporal resolution needed to capture the 3D depth of a dynamic scene. Time-of-flight cameras have improved in recent years, but still suffer from low resolution and noisy measurements. Stereo-camera based approaches generate 3D models of a scene by stereo matching across pairs, or multiple pairs, of cameras [78]. As shown in Fig. 1.1, if a point in the world can be identified by both cameras and the relative orientation of the two cameras is known, then the depth of the point in the scene can be determined through triangulation. Another method used to reconstruct 3D scenes is structured light (SL), shown in Fig. 1.2. SL systems replace one of the cameras in the traditional two-camera stereo method with a projector which projects patterns that uniquely illuminate points in the scene. The remaining camera identifies the pattern present at each observed pixel, from which correspondences between the camera and projector pixels can be established. From these correspondences, a 3D model is generated by triangulation [96].

We present a multi-view structured-light system capable of capturing the time-varying 3D shape of a human subject from multiple views. We use multiple cameras and projectors placed around a central capture volume as shown in Fig. 1.3. The cameras and projectors are separated into three

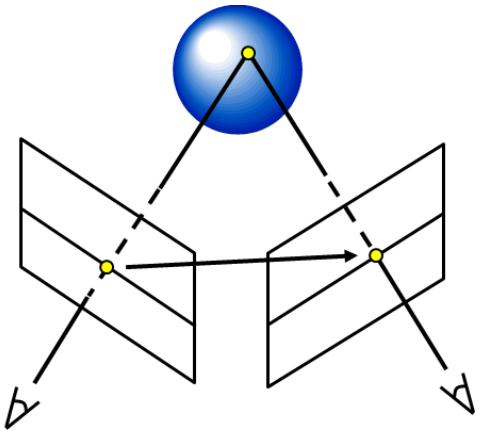


Figure 1.1: Diagram of a simple stereo system.

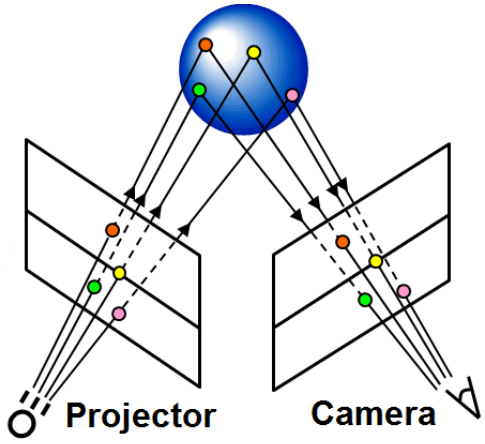


Figure 1.2: Diagram of a simple structured-light system.

structured-light stations each consisting of a projector, two black-and-white cameras, and one color camera. The system is considered a multi-view structured-light system since partial scans of the subject from the views of the multiple structured-light stations are merged together. The challenge with using multiple structured-light stations within a single system is avoiding interference between the projectors in the separate stations.

Our system is the first markerless motion capture system with multiple structured-light stations that can operate together without any interference between projectors. We accomplish this through a system design that strictly controls the timing of all projectors, preventing any projectors from simultaneously illuminating the human subject. With our design, structured-light systems can collaboratively be used in a multi-view system to capture 360-degree geometry in a way that was not previously possible. Previous applications of structured light mostly attempt to capture geometry using a single projector from a single view [77, 90, 96]. Some attempts have been made to remove the interference between multiple projectors in a multi-view structured-light system cap-

turing dynamic geometry[46, 28]. Griesser et al. [46] use two separate structured-light stations positioned facing each other with a human subject placed between the projectors of each station. The interference between the projectors is limited by restricting each projector to only project directly on the human subject surface. While partially successful, the limited coverage of the human subject surface leads to poorly reconstructed geometry. Dou et al. [28] use multiple commercial structured-light scanners, specifically Microsoft® Kinects®, in a single system. To limit interference between the projector devices, each device is vibrated at a unique frequency, thus blurring projected patterns to Kinects® operating different frequencies. The loud noise and potential mechanical issues create drawbacks to working with this system. Our proposed system has absolutely zero interference between stations, helping to ensure the overall quality of 3D scans that are captured.

An additional unique advantage of our system is that it performs markerless motion capture without the need for any additional template capture hardware. In contrast to many methods that require additional 3D scanning hardware to generate an initial 3D model of the human subject, we are able to directly create a watertight mesh, or template, of the human subject. This template can be deformed over time to provide a watertight representation of the human subject during a full capture sequence. Many existing markerless motion capture systems use multiple cameras positioned around a human subject to reconstruct the motion of the subject. While multi-camera setups can directly estimate depth by using stereo techniques, they rely on the unique texture of the scene in order to determine correspondences to calculate depth. By creating a markerless motion capture system that works with multi-view structured light, we are able to directly capture the depth information of a subject from multiple perspectives, even in cases with limited scene texture. This direct generation of surface geometry provides us with a distinct advantage over multi-camera based systems. For example, both Vlasic et al. [88] and de Aguiar et al. [9] use an initial scanned template generated from 3D laser scanners to generate their template for processing. Vlasic et al. attempt to use a template generated from visual hulls in some data sets, but the template quality suffers from this approach. Our system can directly capture the surface geometry of the human subject from 360-degree views, thus allowing us to create a template that accurately represents the shape of the subject.

Finally, our process for deforming the template to the each pose of a captured motion sequence is simplified since we are able to work directly with surface measurements of the human subject. Since the multiple structured-light stations can capture the majority of the human subject's surface, we can generate correspondences between our template and the captured surfaces to drive our template deformation process. In contrast, methods such as [88, 9] use visual hulls to guide the deformation of their templates. However, visual hulls are not able to capture concave portions of the human subject surface. In contrast, the structured-light stations can capture concave surfaces as long a camera and projector from a single structured-light station can observe the scene point. With a better understanding of the true surface shape of the human subject, the multi-view structured-light system can better understand and reconstruct the range of motions of a human subject.

Our proposed multi-view system has three synchronized structured-light stations. Each station consistently captures the dynamic surface of a human subject from a single view. To create a full representation of the subject, the reconstructed surfaces generated from each station need to be

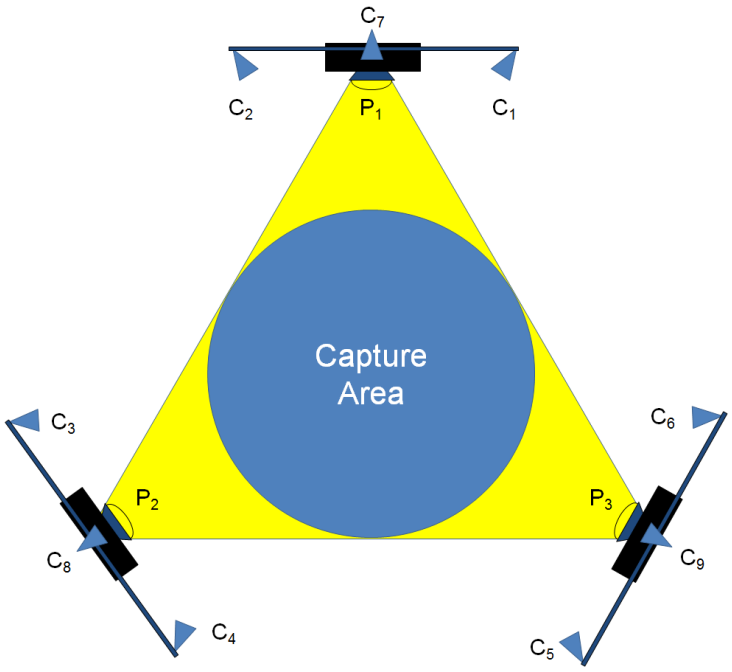


Figure 1.3: Top-down view of the three structured-light stations in our system.

properly aligned with each other. The devices in the system need to be accurately calibrated with respect to one another to enable this. The dynamic surfaces of the subject are captured and aligned to create a partial scan of the subject from multiple views surrounding the scene. The captured surfaces do not provide a full representation of all surface points due to occlusions and the limited views of the system devices. As such, a method to fill in the holes in a time-consistent and coherent way needs to be developed.

We address these four problems in this dissertation: a) designing and building the multi-view structured-light system, b) developing algorithms to capture consistent dynamic geometry in a single structured-light system, c) creating an accurate method for calibrating across all projector and camera devices, and d) developing a reconstruction algorithm to create hole-free representations of the dynamic geometry of a performer.

In this chapter, we provide context for our exploration into each of these problem areas. Specifically, in Section 1.1, we present previous works in structured light and specifically systems that use phase-shifted sinusoidal patterns. We introduce the problem of phase unwrapping along with previous methods for performing phase unwrapping in structured-light systems. We also present a high-level view of our approach to solving the phase unwrapping problem in the context of a structure-light system. In Section 1.2, we present the problem of multi-camera-projector calibration and present previous approaches to calibrating multiple camera and projector devices. We also describe the our proposed method for multi-camera-projector calibration and the associated benefits. Section 1.3 presents background into the problem of motion capture, and specifically

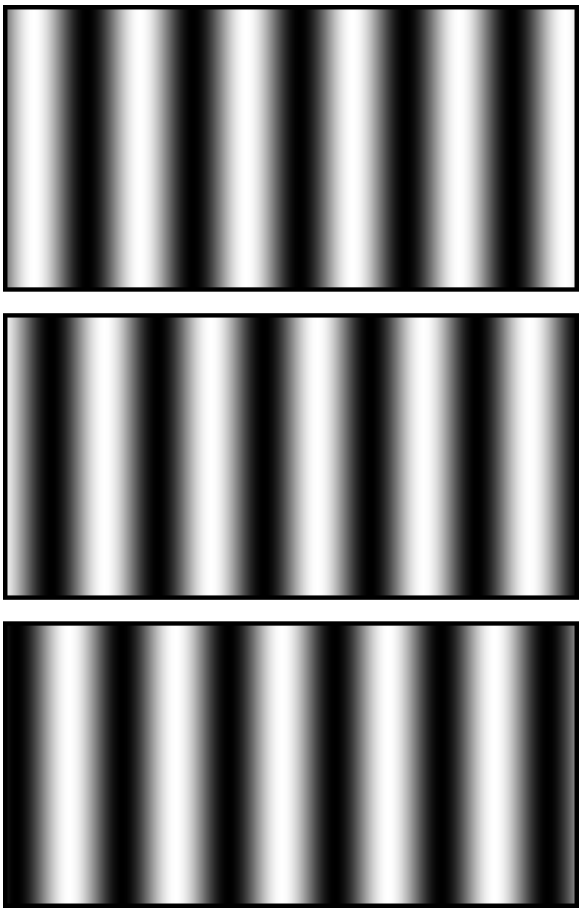


Figure 1.4: Examples of the three phase-shifted sinusoidal symbols used by our system.

markerless motion capture. We describe the operation of our system and the algorithm we developed to reconstruct human motion. In Section 1.4, we describe the contents of each chapter of the dissertation.

1.1 Sinusoidal Structured-Light Patterns and Phase Unwrapping

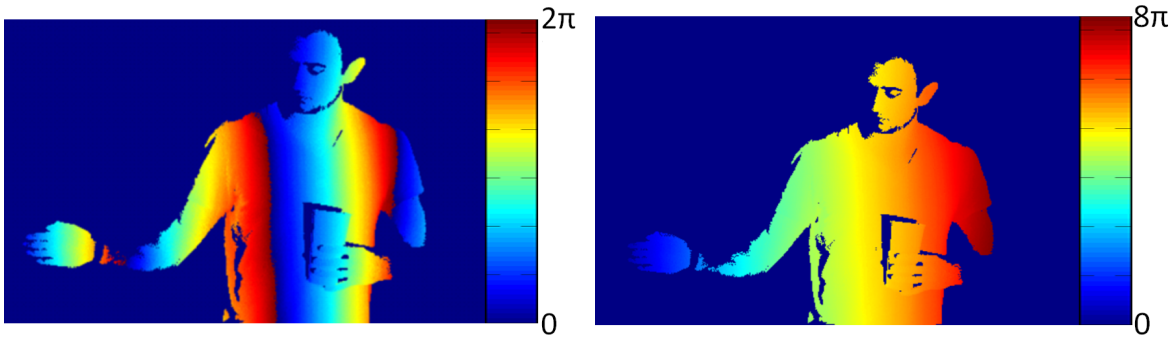
Several structured-light systems have been developed for both research and commercial applications over the years. Within the structured-light literature, there has been extensive research in designing effective SL illumination patterns that uniquely identify points within an illuminated scene [57, 47, 77]. Examples of pattern generation methods include temporal and spatial coding of projection patterns or "viewpoint-coded structured light" [27, 95, 94]. Structured-light patterns consist of anywhere from one projected frame to many more frames. Each frame is an individual projected image that is recorded to help determine correspondences between the camera and pro-

jector pixels. For capturing a dynamic scene using structured light, it is desirable to use illumination patterns that maximize the quality and resolution of each reconstruction while simultaneously minimizing the number of projected frames in the pattern. This is done to maximize the temporal update rate of the scene [77].

An important class of projection patterns in SL systems is phase-shifted sinusoids. In phase-shifted sinusoidal patterns the intensity of the projected pattern varies sinusoidally either from left to right, or top to bottom. These patterns are robust to depth-of-field effects and are simple to decode. In the most common setup, three sinusoidal images, each phase shifted by $2\pi/3$, are sequentially projected onto the scene, as shown in Fig. 1.4. The three observed intensities of a given pixel in the pattern projection are used to determine the wrapped phase of the corresponding pixel in the projector. That is, the phase of each pixel takes on a value from $[0, 2\pi)$. Performing this calculation on all points in the camera image allows a phase image to be generated from which the scene depth can be reconstructed [96, 51, 99]. In SL systems with phase-shifted sinusoidal patterns, it is common to project multiple periods of the sinusoids across the screen. This reduces the number of unique phase values that must be identified, thereby making the decoding process less susceptible to noise. At the same time, for an M-period projection there is an M-fold ambiguity in finding the corresponding point in the projector's image because only the wrapped phase of each pixel can be determined. This ambiguity is removed through the image phase unwrapping process [98].

There are many image processing applications where measured data represents the phase of a sinusoidal signal. Applications such as synthetic aperture radar, interferometry, and magnetic resonance imaging all work with phase based data [42]. Often, rather than recording the true phase of a signal, the data is restricted to taking on values from $[0, 2\pi)$ due to the periodic nature of sinusoidal signals. Significant information is lost when the true phase of a signal is restricted, or wrapped, to $[0, 2\pi)$. Phase unwrapping is the process in which the true phase of captured data is estimated given the wrapped phase data. Specifically, the input to an image phase unwrapping algorithm is an image with pixel values that are restricted to $[0, 2\pi)$. The output of the algorithm is an estimate of the true phase for each pixel in the image. The true unwrapped phase of an image is estimated by strategically removing 2π phase discontinuities between neighboring pixels in an image. Fig. 1.5a illustrates a phase image with wrapped phase values. Rather than exhibiting discontinuities in the phase as neighboring pixels transition from 2π to 0, an unwrapped phase image removes the discontinuities and allows the pixels to take on phase values outside of the $[0, 2\pi)$ range. An unwrapped phase image is shown in Fig. 1.5b. In the context of a structured-light system, each column of the projected sinusoidal patterns has a true phase, but our cameras are only able to directly calculate the wrapped phase value of each pixel. By performing phase unwrapping, correspondences between camera pixels and projector columns can be determined to allow the depths of scene points to be triangulated.

Two-dimensional phase unwrapping has been extensively studied in the signal processing literature [43, 17, 41, 42] in general and for sinusoidal structured-light systems in particular [98, 17, 55, 64, 80]. The approaches to phase unwrapping in sinusoidal systems can be classified into three main categories: temporal, period coded, and spatial phase unwrapping methods. Temporal unwrapping methods project additional patterns to remove the absolute phase ambiguity from the



(a) (b)

Figure 1.5: An example of a phase image that has (a) wrapped phase values and (b) unwrapped phase values.

phase-shifted sinusoidal patterns [14, 53]. In [14], gray-coded patterns are projected in addition to the phase-shifted sinusoidal patterns. The gray-code patterns encode the period of each projected pixel so that the absolute phase of each pixel can be directly determined. In another approach by Huntley and Saldner [53], the frequency of the projected sinusoidal pattern is increased as each pattern is projected. The absolute phase of each pixel is then determined by counting the number of 2π period crossings in each pixel's temporal sequence. In their experiments, 17 patterns are projected for a single reconstruction. Temporal methods allow for the absolute phase of each pixel to be determined at the cost of significantly increasing the number of projected patterns. As such, unless very high-speed cameras are used, temporal-based projection methods are not suitable for dynamic scene capture.

Period-coded methods embed the period information within the same patterns as the phase-shifted sinusoids. Liu et al. [66] add a single-period sinusoid to the high-frequency sinusoidal pattern. By projecting five phase-shifted versions of the summed sinusoidal patterns, the phase of both the high and low frequency sinusoid can be determined at each point. The two phase estimates can then be used to generate the absolute phase of each pixel. Once again, this comes at the expense of using additional patterns. The final approach to phase unwrapping is spatial unwrapping. In spatial unwrapping, the unwrapped phase of each pixel is determined by locally summing phase values along paths within the 2D image. Unlike temporal and phase-coded methods, spatial unwrapping methods do not always provide absolute phase results. It is often the case that the unwrapped results have an additional unknown $2\pi k$ offset, where k is an integer value. It is necessary to determine the absolute phase in order to calculate the true depth of an object.

To generate error-free unwrapped results, a number of assumptions are made about the content of the scene during the spatial phase-unwrapping process in sinusoidal SL systems [55, 64]. First, the scene is assumed to contain a single continuous object in space, i.e., all objects in the scene are connected together. Second, phase unwrapping via traditional spatial methods requires most neighboring points to have phase differences that are smaller than $|\pi|$. Without this assumption, the true phase differences cannot easily be determined because of the periodic nature of the phase val-

ues. This typically restricts the types of scenes that can successfully be reconstructed. For instance, scenes with multiple disconnected objects or large depth discontinuities cannot be unwrapped with spatial phase-unwrapping techniques alone.

As mentioned, spatial phase unwrapping allows for the recovery of the relative phase between pixels, but it does not directly anchor the phase of each pixel to an absolute value. Without absolute phase, there remains a constant unknown offset in the correspondence between camera pixels and projector columns. A variety of techniques have been developed to determine this constant in order to recover absolute depth via triangulation. Zhang and Yau [99] embed a small cross in the projected pattern to serve as a reference position with a known absolute phase. Once detected, all the remaining points are unwrapped with respect to the marker in order to determine the absolute phase of the entire image. The embedded marker serves as a valid reference point, but it can be difficult to determine the position of the marker if the projected pattern is out of focus or if there is significant texture within the scene.

While traditional SL systems consist of a single camera and projector, several SL systems with multiple cameras recently have been proposed [99, 48, 91]. In addition to using a marker to solve for absolute phase, Zhang and Yau [99] use two cameras to increase the reconstruction coverage of a scene. With the absolute phase recovered, the reconstructions from the two camera views can be accurately merged. Han and Huang [48] use a pair of stereo cameras with a sinusoidal SL system to achieve accurate stereo matches between the cameras. They search for correspondences between the cameras by using absolute unwrapped phase images. By not triangulating depths with respect to the projector, their method does not require projector calibration and is robust to distortions of the sinusoidal patterns caused by the projector. However, their method only works on scenes with no large depth discontinuities, since spatial unwrapping is required. A second camera can also be added to the single-camera, single-projector SL system [91] to overcome the phase discontinuity problem and allow for the reconstruction of scenes with multiple objects. In doing so, the absolute phase for many points in the scene can be determined. Specifically, the second camera resolves the absolute phase ambiguity as follows [91]: for each pixel in the first camera, the wrapped phase is calculated, and the possible corresponding locations in the projector are determined. Given M periods in the projected sinusoidal patterns, there are M columns in the projector that could possibly correspond to the identified camera pixel. These M points are triangulated in 3D space, and each 3D point is projected onto the image plane of the second camera. Each projected pixel's phase is then compared to the phase of the pixel in the first camera. The projected pixel with the closest phase to the reference pixel is identified as the correspondence. This approach can result in occasional absolute phase errors due to occlusions, calibration errors, "false" matches, and scene motion. The errors are cleaned up by an energy minimization algorithm that uses both cost components that are based on data and those that enforce smoothness across pixels. We refer to approaches that use a second camera to help in the phase unwrapping process as stereo-assisted phase unwrapping methods.

In a stereo SL system capturing multiple poses of a dynamic scene, it is important for the unwrapped image of each camera at each time to be consistent. Specifically, "viewpoint-consistency" means that the unwrapped phase values for corresponding pixels in each of two cameras viewing the same point in the scene will be assigned the same absolute phase. Similarly, "temporal-

consistency" implies successively unwrapped phase images have similar phase values for all corresponding points between consecutive frames.

Three-dimensional phase unwrapping methods have been explored in the literature [52, 26, 24, 54, 6, 76, 87, 8, 49, 68, 7, 40, 5, 61, 11, 20, 80, 44, 10, 23, 79]. Specifically, Su and Zhang [80] examine the three-dimensional unwrapping problem for a structured-light application. Despite generating phase images through Fourier transformation profilometry rather than through shifted sinusoids, they present several methods for unwrapping a three-dimensional phase volume that is generated by stacking consecutive phase images. In Huntley's work [52], a 3D volume is unwrapped by removing edges between pixels that likely have a phase difference, or discontinuity, greater than 2π. This method is capable of dealing with 2π discontinuities from noise, but cannot deal with true phase discontinuities.

We present two stereo-assisted phase unwrapping methods for two-camera stereo structured-light systems. The first method enforces viewpoint consistency by phase unwrapping in the projector domain. Loopy belief propagation is run over the graph of projector pixels to select pixel correspondences between the left and right camera that align in 3D space and are spatially smooth in each 2D image. The second method enforces temporal consistency by unwrapping across space and time. In this method, we combine a quality guided phase unwrapping approach with absolute phase estimates from the stereo cameras to solve for the absolute phase of the connected regions. The quality guided approach defines the order that pixels are unwrapped according to the relative phase difference between pixels connected by an edge. Both of our approaches are able to accurately unwrap phase images, which in turn leads to more accurate triangulated point clouds. In contrast to previous approaches, both of our methods take advantage of multiple observations of the phase data to better estimate how each individual pixel is unwrapped. With more observations, the uncertainty of unwrapping a single phase pixel decreases. We present results for both methods to show their effectiveness on real-world scenes.

1.2 Calibrating Multiple Projectors and Cameras

Calibrating multiple cameras and projectors is the process of estimating the intrinsic and extrinsic parameters of each camera and projector in a system. The lenses of cameras and projectors are modeled the same way, since the only difference between a camera and projector is in the side of the lens in which light is generated. The intrinsic parameters of a camera or projector include the focal length, camera center, pixel skew, and radial lens distortion parameters of a device. With these intrinsic parameters, the transformation between the 3D coordinates of a scene point and the corresponding device pixel coordinates onto which it is projected can be made. Extrinsic parameters specify the 3D location of each device along with the device's orientation. The position and direction of a device can be specified by 6 values: an X, Y, Z for device center and X', Y', Z' for device direction. Alternatively, the extrinsic parameters are commonly defined as a rotation and translation from the world coordinate system to the local coordinate system of a device, which can also be specified by six parameters. When calibrating multiple devices, the extrinsic parameters are all defined in a single world coordinate system to relate the devices to each other. In most

approaches that attempt to calibrate multiple camera or projector devices, points in a 3D scene are captured by the observing devices and the location of each point in each device is determined. For every 3D sample point, a set of corresponding image locations is generated across devices that can observe the point. Sets of these correspondences along with an estimate of the 3D location of each point are generated and fed into an optimization process. The output of this optimization is an estimate of the intrinsic and extrinsic parameters for each device. In a multi-view structured-light system, the calibration parameters are used when generating partial surface scans from each structured-light station. Accurately calibrated parameters ensure that the scans generated from each of the stations correctly represent the shape of the captured geometry and also correctly align the partial observations of the scene.

In a system with multiple structured-light stations it is important for all cameras and projectors to be accurately calibrated to ensure the captured geometry is well aligned. Camera and projector calibration is a crucial step in the deployment of many computer vision systems, especially in those with multiple cameras and projectors. Common applications of multi-camera-projector (MCP) systems include CAVE-like or large displays [75, 84] and multi-view geometry capture with structured light [46, 32]. There are currently publicly available tools such as Bouguet's Camera Calibration Toolbox [18] for calibrating cameras. Multi-camera calibration has also been addressed in the literature with publicly available tools [86, 83, 34]. Specifically, Svoboda et al. propose a multi-camera calibration method which tracks the position of an LED across synchronized images from each camera view [83]. Projector calibration has also been studied extensively. Since projectors cannot observe the scene, projector calibration has traditionally been preceded by camera calibration [97, 65, 12]. Alternatively, it is possible to calibrate a camera and projector simultaneously [19, 92]. There are also a number of approaches to calibrate multi-projector systems such as CAVE-like augmented reality systems [25] and flexible reconfigurable projection screens [93]. In the majority of these approaches, the intrinsic and extrinsic parameters are not directly computed. Rather, a mapping between the projector pixels and projection surface is estimated to reduce interference between projectors.

Several methods have been proposed for full intrinsic and extrinsic calibration of an MCP system [75, 45, 59]. In [75], cameras and projectors are arranged to create a large display surface. The geometry of the surface is reconstructed by using structured-light patterns to determine correspondences between a pair of stereo cameras. These points are triangulated to 3D positions that can then be used to calibrate the projector. This process is repeated for each projector separately. In [45], all the cameras are first calibrated; then each projector is calibrated by projecting fiducials onto a board, capturing images of the projection, and robustly detecting the 2D and 3D positions of the fiducials to calibrate the projector. They develop an automated process to make calibration easy to complete. With this approach, it is not possible to generate projector-projector correspondences, except in configurations with highly overlapped views between the projectors; it is also not possible to generate correspondences between all devices in the system. Kobayashi et al. [59] perform MCP calibration as well but require the cameras to be positioned such that they are directly illuminated by the projectors. This greatly restricts the possible camera-projector system configurations. The downside to each of these MCP calibration methods is that the projector intrinsic and extrinsic parameters are estimated separately from the camera parameters, thus propagating the error in the

camera calibration to the projector. Ideally, all parameters should be jointly recovered.

We describe a calibration method for multi-camera-projector systems in which sensors face each other as well as share a common viewpoint. In contrast to previous approaches, we are able to simultaneously calibrate all devices within our system. By calibrating all devices jointly, we ensure that any error present in the computed device parameters are equally distributed across all devices. Methods that first calibrate cameras and then attempt to individually calibrate projectors propagate the individual errors of the cameras to the following projector parameters. Additionally, with our approach, we are able to easily generate a dense set of correspondences throughout the entire capture volume, which results in calibrated parameters that work well across the capture volume. Our method allows us to generate subpixel reprojection error across all devices. In our approach, we use a translucent planar sheet framed in PVC piping as a calibration target which is placed at multiple positions and orientations within a scene. In each position, the target is captured by the cameras while it is being illuminated by a set of projected patterns from various projectors. The translucent sheet allows the projected patterns to be visible from both sides, allowing correspondences between devices that face each other. The set of correspondences generated between the devices using this target are input into a bundle adjustment framework to estimate calibration parameters. We demonstrate the effectiveness of this approach on a multi-view structured-light system made of three projectors and nine cameras.

1.3 Capturing Motion of Human Subjects

Our proposed calibrated multi-view structured-light system can capture dynamic geometry from multiple views around a performer and can be used to reconstruct the movement of a human subject. Capturing the dynamic geometry in this way provides benefits over existing marker-based motion capture systems that require the performer to put on a specialized suit with many trackable markers. While marker-based motion capture has been popular for a while, there has been continued desire to do away with markers, simplify the capture process, and improve realism by capturing actual subject color and shape. Even though a marker-based system with many markers can increase the realism of results, it comes at a significant setup cost where hundreds of markers must be placed directly on the body of the human subject [73].

In many systems, the geometry of the dynamic human subject is captured using multi-camera setups [21, 85, 9, 13, 88, 35]. These systems recover a representation of the human subject by using visual hulls generated from each camera view. Many approaches use a template of the human model to assist in the reconstruction of the dynamic geometry [21, 9, 13, 88, 35, 71, 22, 28]. The methods using templates can either use a skeleton [21, 13, 88, 35, 71, 22] or not [9, 28]. Vlasic et al. [88] generate a detailed template using a high-quality laser scanner. The template is then deformed according to constraints gathered from the observing cameras. De Aguiar et al. [9] similarly present a template-based method that starts with a high-quality laser scan, but they do not use a skeleton to deform the mesh. Rather, mesh-based deformation techniques such as those in [81] are used to deform the mesh according to the visual hulls of the observing cameras. The method of Vlasic [88] fits a template via a combination of coarse movements from skeleton

deformation, followed by local mesh deformation to capture the details of the scan. Gall et al. [35] use a similar approach, but do not require manual user intervention. Methods using highly detailed templates are sometimes criticized for "baking in" details that should not be present throughout the reconstructed sequence.

Besides multi-camera methods emphasizing the use of visual hulls, other capture methods have been proposed. In [89], dynamic scans from a photometric stereo system are used for surface reconstruction. While the geometry captured from this system is quite accurate and detailed, the system is complex and expensive. Specifically, it uses a geodesic sphere eight meters in diameter and with over 1,200 individually controllable light sources, making it inaccessible to most users. Unlike methods that work with visual hulls from multiple cameras, in [89] the 3D shape of the human subject's surface is directly captured. Such direct capture systems directly reconstruct the subject's surface without any prior information rather than using a template to estimate the pose and shape of a human subject. For instance, the approach in [89, 62] does not use templates to reconstruct watertight meshes. The results from these direct capture approaches often yield errors in the topology of the reconstructed geometry since there is no prior information on the shape of the model. Similar works have focused on the problem of registering and generating consistent geometry from sets of dynamic point clouds [69, 82, 74]. These methods only work with high temporal sampling and limited occluding geometry.

We present a multi-view structured-light system for markerless motion capture of human subjects. Our system is the first system to effectively remove interference between multiple structured-light stations to allow them to operate within a single motion capture system. The interference in our system is removed by tightly controlling the timing of the projectors and cameras operating within the system. We are able to ensure that only a single projector illuminates the scene at any point in time. Additionally, in contrast to other markerless motion capture approaches, we are able to directly create a model of the captured human subject without the need for additional hardware. This is in contrast to many systems that require additional 3D laser scanners to capture a template model. Since our system is able to generate 3D surface scans of the human subject, the process of deforming the template to the current pose is simplified. Multi-camera methods that do not directly measure surface geometry often create an estimate of the human subject shape by using visual hulls. These visual hulls are not able to directly estimate the surface of the human subject, and therefore have less unique information to constrain the deformation of the mesh. For example, visual hull approaches are not able to capture concave surfaces, where this is not a problem for our system. The use of true 3D scans rather than visual hulls allow us to efficiently and accurate deform our template to the current pose.

We develop algorithms to reconstruct dynamic geometry using a template generated by the system itself. The template is fitted with a internal skeleton that represents the basic human bone structure. By repositioning the bones of the skeleton, the pose of the template can be changed to align with scans from the system. Specifically, we deform the template to each frame of the captured geometry by iteratively aligning each bone of the skeleton. This is done by searching for correspondences between the source template and the captured geometry, solving for rotations of bones, and enforcing constraints on each rotation to prevent the template from taking on unnatural poses. Once the sequence of captured geometry is processed, the template is textured using color

images from the multi-view structured-light systems. We show the effectiveness of our system for a 50-second sequence of a moving human subject.

1.4 Contributions and Organization of the Dissertation

In this dissertation, we present a multi-view structured-light system for markerless motion capture of human subjects. This is the first interference free system consisting of multiple structured-light stations operating. Our design ensures strict timing between all cameras and projectors. With precise control over device timing, we can ensure that only a single projector illuminates the human subject at any time, thus preventing interference between individual structured light stations.

Additionally, our system is able to directly generate a watertight template of human subjects using the partial scans resulting from multiple structured-light stations. Many markerless motion capture systems deforming a template require additional 3D scanning hardware such as laser scanners to capture the initial template shape of the human subject. In our system, the subject is positioned in a relatively occlusion free pose from the perspective of the structured light stations, a watertight mesh can be fit over the partially observed subject. Our template provides an accurate representation of the shape and texture of the human subject.

Finally, a distinct advantage of our system over other markerless systems is in direct measurement of the shape of the human subject's surface. Specifically, existing multi-camera markerless motion capture systems do not directly capture the surface shape of the human subject, and they often resort to estimating the human subject's pose by using visual hulls. Direct capture the subject's surface allows for accurate correspondences between the template and surface scans, leading to a deformation method which consistently and accurately captures human motion.

The outline of this dissertation is as follows: in Chapter 2, we present an overview of the architecture, hardware, and software of our multi-view SL system. All algorithms and methods presented in this dissertation rely on the underlying ability of the system hardware to synchronously project structured-light patterns and capture images of the scene.

In Chapter 3, we present methods for unwrapping phase images generated from each structured-light station. Sinusoidal structured-light patterns require phase unwrapping before correspondences can be determined between the camera and projector pixels. Our two approaches for performing phase unwrapping in a structured-light system are more accurate than previous phase unwrapping approaches because we take advantage of redundancy in the observed wrapped phase values to generate better estimates for unwrapped phase values. We use multiple observations of phase data either by using two views of the phase image in the scene or by examining the phase at consecutive time instances. The multiple observations of the phase of each point in the scene allow us to more accurately estimate the absolute phase of a pixel with the uncertainty decreasing with each additional observation. By unwrapping the phase images in a consistent manner, either in time or space, accurate measurements of the scene's 3D surface can be captured.

In Chapter 4, we present a method for calibrating a system with multiple cameras and projectors. Our method uses a novel calibration target that allows us to increase the set of correspondences between our system devices. The target is a white bed sheet stretched over a frame made of PVC

piping. We project structured-light patterns onto the target, which are captured by the system cameras to determine correspondences between camera and projector devices. The translucent nature of the sheet allows the projected patterns to be visible from both sides of the target. This means that a large set of camera-projector pairs can generate correspondences. The result is a set of accurate correspondences that can be passed into an optimization framework to generate estimates of all calibrated parameters. Additionally, our target can easily be positioned throughout the entire capture volume to ensure the calibrated parameters are the best estimates for the entire volume. By using a novel calibration target and relying on the fundamental ability for structured-light systems to generate correspondences between camera and projector devices, we are able to simultaneously calibrate all cameras and projectors in the system. It is through accurate calibration of the system devices that the data from all of the stations can be merged into a single coordinate frame. In contrast to previous approaches, we are able to simultaneously calibrate all projectors and cameras in our system. By calibrating all devices together, any residual error in the calibrated parameters is evenly distributed across all devices. Additionally, our novel target allows us to generate correspondences between cameras and a projector, even in configurations where the cameras are directed towards the projector. This allows us to generate correspondence across a greater number of camera and projector devices, which in turns improves the accuracy of our calibrated results.

In Chapter 5, we present the overall system in its final form along with our method to capture the motion of a human subject. The output of this work is a dynamic colored model of a captured human subject. Our system is able to effectively combine partial 3D scans of the system that are captured by individual structured-light stations. This work illustrates the first system that completely removes interference between multiple structured-light stations in a single system. Additionally, our approach allows us to perform markerless motion capture using only our system. This is in contrast to many approaches that require additional scanning hardware to generate initial template scans of a performer. In our case, we directly generate our template from our system and deform it to fit the partial scans of the human subject's surface captured by the structured-light stations. Additionally, we create an efficient approach to deform and constrain a skeleton embedded in our template to fit the partial scans captured over time. Because we are able to directly capture the surface geometry of the human subject, unlike methods that use visual hulls, we can consistently constrain the pose of our template to fit our partial scans.We also present an approach to texturing our template that results in a realistic representation of the human subject. The end result is an accurate reconstructed model of a human subject with true surface texture.

In Chapter 6, we offer concluding remarks on the system and directions of future work.

Chapter 2

The Multi-View Structured-Light System

The algorithms and methods in this dissertation could not have been explored without first developing the hardware and software components that enable the system to capture data. In any structured-light system, the two most important components are the projection devices and the capturing cameras. Capturing dynamic geometry from multiple views requires us to develop a system in which multiple cameras and projectors can operate together. The key to enabling the integration of all devices into a single system is developing controls that ensure accurate synchronization.

Our system consists of three major subsystems: the projection, capture, and synchronization subsystems. The projection subsystem, presented in Section 2.1, is responsible for controlling and displaying the structured-light images that are projected onto the scene. In addition to the projectors, the projection subsystem consists of specialty graphics cards and a dedicated PC for driving images to the projectors. The capture subsystem, presented in Section 2.2, is responsible for gathering observations of the scene as the projectors illuminate the human subject. The capture subsystem includes the cameras and the associated hardware needed to stream all of the image data into a computer for data processing. The synchronization subsystem, presented in Section 2.3, ensures that the cameras are precisely timed to capture the scene as it is illuminated by each frame of the structured-light patterns. The synchronization subsystem has an embedded computer that receives timing information from the projector subsystem and feeds it into the capture subsystem. We present our approach to learning the timing of our projectors in Section 2.3. Each of these three subsystems relies on custom developed software to contribute to the overall function of the system.

Also, the system can be thought of as being made of three separate stations each of which holds components of the three subsystems that operate together to capture a partial view of the scene. Each station consists of a projector, two cameras for capturing geometry, and a color camera. In total, the system has three projectors and nine cameras. Additionally, each station receives timing signals that keep all devices operating in sync. A block diagram of our system is shown in Fig. 2.1

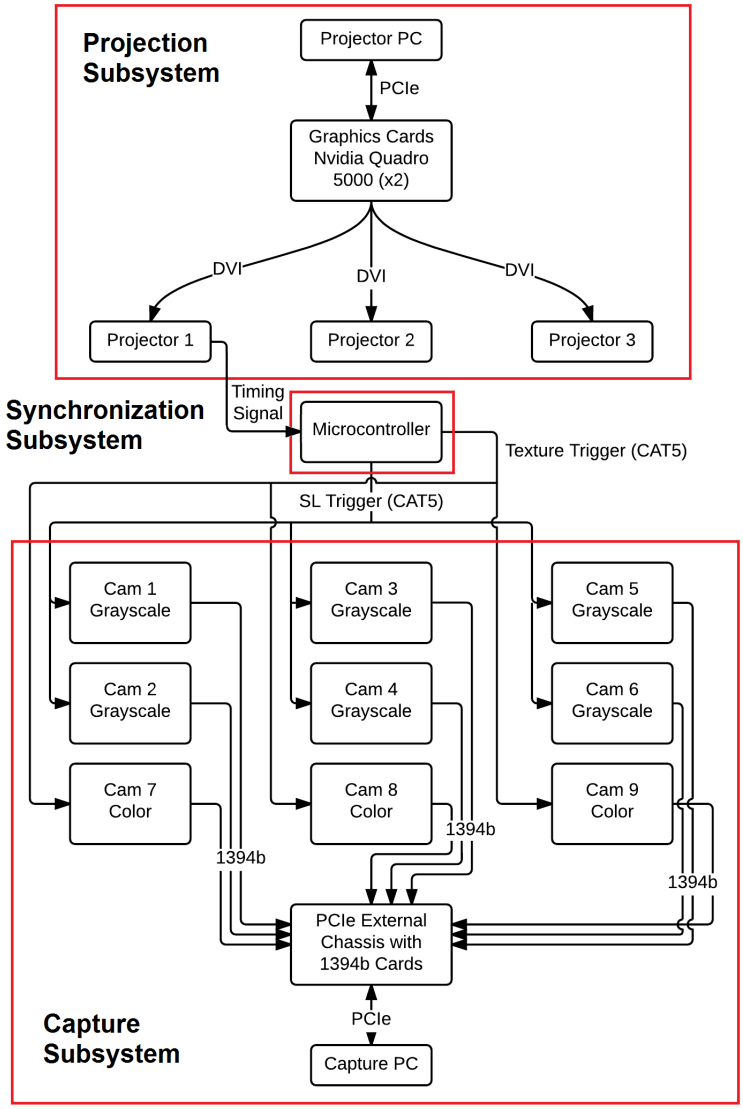


Figure 2.1: Block diagram of the components of our multi-view structured-light system.

2.1 The Projection Subsystem

In structured-light systems, the projector is responsible for providing the scene with unique texture through projected patterns that can be viewed by an observing camera. The unique texture of the projected patterns makes it possible for correspondences to be determined between camera and projector pixels. While structured-light systems are common, our desire to capture dynamic geometry from multiple views adds several requirements not seen in most structured-light systems. These requirements are:

1. The projection subsystem must drive all projectors in the system.
2. The projectors must be precisely synchronized to one another to control which projector is illuminating the scene at any point in time.
3. The projection system must control the images sent to each projector at any point in time.
4. The projection system must output timing information to the synchronization subsystem.

There are three main components in the projection subsystem that enable these functionalities possible: the projectors, the computer, and the synchronized graphics cards.

2.1.1 Projectors

There are a few key requirements in the selection of the projectors for our system. The most critical requirement is for the projectors to use DLP®, digital light processing, technology. The operation of DLP® projectors is different from other video projectors. Specifically, DLP® projectors generate their color images by successively displaying the red, green, and blue color channels of each video frame. For example, in a 60 Hz color video stream, the projector displays each color channel at least once per video frame, so the color channels are displayed at a rate of at least 180 frames per second. The DLP® projector uses a digital micro-mirror device (DMD) that is able to generate grayscale images when white light is projected onto the surface of the device and reflected out of a projector lens, as shown in Fig. 2.2. If the color of the white light is changed to red, blue, and green, the individual color channels of a full color video frame can be generated. To change the color of the projector's light, DLP® projectors typically use a color wheel, as shown in Fig. 2.2. The color wheel is a disk with glass color filters that sits in the optical path of the projector's light source. By rotating the color wheel, the color of the light illuminating the surface of the DMD device is rapidly changed. By removing the color wheel in a DLP® projector, the projector can be used as a high-speed grayscale video projector. If three grayscale images are loaded into the three channels of a single color image, the projector will project out the three grayscale images in the time it would have normally projected a single color image. With this slight modification, DLP® technology allows for standard off-the-shelf 60 Hz color video projectors to operate as 180 Hz grayscale projectors. This high speed operation is critical to our system.

We use three projectors in our system, one for each of the structured-light stations. Specifically, we use Optoma TX780 projectors because they offer a bright lamp with 4,000 lumens, a 60 Hz

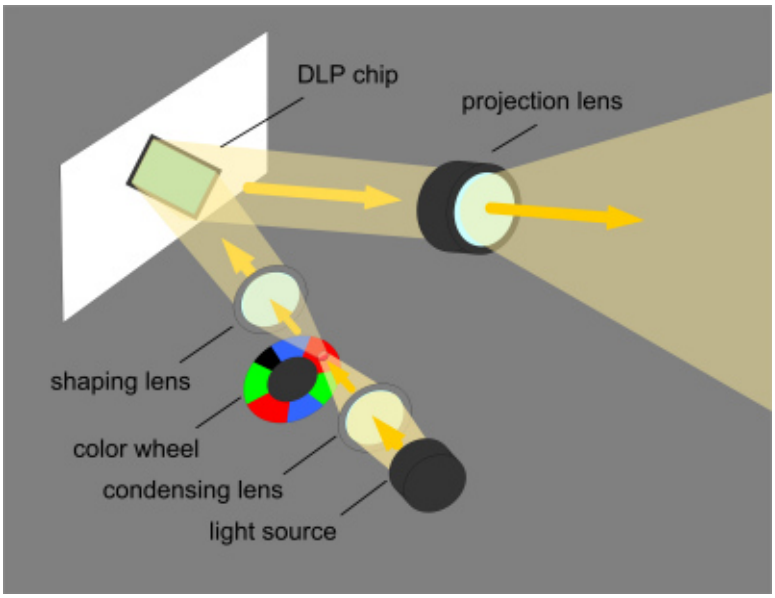


Figure 2.2: Internal layout of a DLP® projector. Source: www.bnoack.com

video display rate, and a resolution of 1024×768. Additionally, this projector offers a 28-33.6 mm zoom lens with an aperture of F2.4-2.6. With this lens, the field of view of each projector is large enough to illuminate the fields of view of the cameras in its station. Each projector is driven by a DVI-D connector via the graphics card connected to the projector computer.

2.1.2 Projection Computer

We use two different computers in the projection and capture subsystems. The capture computer has a significant processing load while streaming in images from all cameras in the system. We have empirically found that using a single machine to also drive the projectors is requires too much computational load and increases the changes of dropped frames and dropped images. To increase the stability and consistency in the projection process, a separate computer is added to the capture system.

The CPU performance requirements for the projector computer are minimal since it is only responsible for driving the image sequences to the projectors. We opt for a machine with dual PCIe ports and a full-size case in order to accommodate the two graphics cards in the system. Our final selection is a Dell T7500 with a 2.4 GHz Intel® Xeon® Processor and 4 GB of RAM. We run Windows® 7 on the machine in order to use DirectX® to control the data stream to the graphics cards.

2.1.3 Graphics Card

The most critical components of the projection subsystem for enabling mutli-view dynamic structured light are the graphics cards. There are several key requirements for the system graphics cards. First, we need to drive three projectors from the projector computer, and second, these three displays need to be synchronized with one another.

Even though graphics cards driving multiple monitors are quite common today, the majority of them are not suited for our system. Synchronization between multiple displays is not a commonly available feature in most graphics cards. For common display applications, there generally is not a requirement to ensure that all of the screens obtain each new video frame at exactly the same time. For some high-performance gaming and multi-monitor stitching of large displays, this feature is needed. NVIDIA® offers the ability to synchronize the output of multiple display ports through the use of a Quadro® G-Sync® daughter card. We use two Quadro® 5000 graphics cards and the additional G-Sync® card to ensure the video outputs are synchronized. Each card is capable of driving two displays. Therefore, with two cards we have a sufficient number of display ports to drive all three projectors.

To drive images to each display, the graphics cards are set to NVIDIA® Scalable Link Interface (SLI) Mosaic™mode, a setting where the graphics cards work together to emulate a single display. In this case, the dual cards emulate a single display with a width three times that of a single projector. Each third of the display corresponds to pixels of one of the three projectors. The advantage of this setup is that by displaying a single image of resolution 3072×768 we can specify exactly which frame of data is going to each projector at the same time.

2.1.4 Projection Software

To drive structured-light patterns to the synchronized displays, we develop custom software with Microsoft® DirectX®. Multiple functions and processes of the system, including multi-camera-projector calibration and multi-view dynamic geometry capture, require the ability to precisely control the patterns sent to each projector without dropping any video frames. To achieve this level of control, Microsoft® DirectX®, and in particular Direct2D®, is used to specify the images sent to the video buffer of the graphics card for each video frame.

2.2 The Capture Subsystem

The capture subsystem is responsible for streaming in image data and performing all calculations towards reconstructing dynamic geometry. The capture subsystem consists of three major components: cameras, a single computer, and an external PCIe chassis. Despite streaming in data from nine separate cameras, a solution with a single computer is chosen to reduce the overhead of transferring captured images between machines and from synchronizing image capture across multiple computers.

2.2.1 Cameras

The main requirement for selecting cameras is that each camera must have a high enough frame rate to capture each projected frame of the structured-light pattern. Additionally, the cameras need to be triggered externally and with exact timing. We require two separate types of cameras. Grayscale cameras are used in the system for capturing the structured-light patterns and generating the dynamic geometry. Color cameras are added into the system for texturing the captured dynamic geometry during reconstruction.

We use Point Grey cameras throughout the system. These cameras offer detailed API for controlling the cameras and operate at our desired frame rate. Point Grey offers a range of connection options, but we use 1394b buses for our cameras. The 1394b data bus offers data rates up to 800 Mb/s which ensures that the connection protocol is not the bottleneck in capturing images.

We use Point Grey Dragonfly Express® for our grayscale cameras. This camera provides 640×480 resolution images at up to 200 frames per second and is equipped with a Kodak $\frac{1}{3}$" progressive scan charge-coupled device (CCD). The large pixel size of $7.4\,\mu$m helps capture low noise images. We use six Dragonfly Express® cameras in the system for capturing geometry, two at each station.

We use Point Grey Flea®2 for our color cameras. The Flea®2 does not offer the same high frame rate options like the Dragonfly Express®, but it provides full color at up to 80 frames per second, fast enough for capturing texture. The Flea®2 has a native resolution of 648×488 in a $\frac{1}{3}$" sensor size. Additionally, both cameras offer a triggering option where the camera starts an exposure on the rise of a externally provided trigger signal. This means we can synchronize the capture of images by providing a hardware generated signal to all cameras.

We use the same lens for all cameras in the system: a Fujinon 1:1.4/9 mm HF9HA-1B. This lens offers a 30° horizontal and a 14.5° vertical field of view, which closely matches the field of view of the projectors.

2.2.2 Capture Computer

The computer used for the capture subsystem meets many specifications. Since this machine is not only used to capture image data, but also to store and process data, it needs to have significant computational performance and storage capacity. Additionally, it needs to interface with all nine cameras in the system and stream in their data.

A custom machine was purchased from Colfax International. This computer is equipped with two Intel® Xeon® Quad Core 3.0 GHz chips and 32 GB of RAM. The 32 GB of RAM is useful when processing large data sets for geometry reconstruction, and even more importantly, it allows us to stream all of the camera data directly to RAM before it is stored on disk. With 32 GB of RAM, dynamic sequences of approximately two minutes can be captured without having to store any data to disk. The computer is also equipped with multiple hard drives in a RAID configuration to improve the writing speed to disk.

2.2.3 PCIe Expansion Chassis

One of the main requirements of the computer is the compatibility to interface with all nine cameras in the system. The capture computer did not initially have enough PCIe slots for all IEEE 1394b cards needed to connect nine cameras. As such, we use a PCIe expansion chassis to handle this problem. Specifically, we use a Dolphin Express DHX510 Host Adapter with DXE410 I/O Expansion Chassis. The host adapter is a PCIe card installed in the main machine and provides two terminals to interface with the expansion chassis. The expansion chassis is a metal enclosure capable of holding up to 8 PCIe cards. In our case, we fill each slot with an IEEE 1394b PCIe card. This allows us to dedicate a single card to each camera, with the ninth camera being connected to a single IEEE 1394b PCIe card installed directly into the main machine.

2.3 Synchronization Subsystem

The largest challenge in developing the physical system is synchronization. At a basic level, structured light only requires capturing images of patterns that are projected onto the scene. However, when dealing with capturing dynamic geometry, the timing at which these images are captured relative to when the patterns are projected becomes extremely important. There are two important timing constraints that must be met by the system. First, we want to ensure that only one projector illuminates the scene with patterns at any time. This is to remove any interference that could occur if multiple projectors simultaneously project their patterns onto the scene. Second, the internal timing of the projector's color channels needs to be learned so the cameras can be synchronized to capture images as each channel is projected. In doing this, we can capture three grayscale images per video frame, effectively increasing the speed of the projector by $3\times$. We synchronize the cameras to the projector such that the exposure time of each image corresponds to the duration of a color channel in the projector.

2.3.1 Learning Projector Timing

When a DLP® projector displays a full color image, it sequentially displays the red, green, and blue color image components. If the color wheel is removed from the projector, then the projector can be treated as displaying three grayscale images within each 60 Hz frame of video. In [96], the color channels of the DLP® projector are used in this way.

The challenge of this approach is learning the internal timing of the projector to synchronize the cameras to only capture images within a single color channel. Specifically, we need to set the exposure time of each camera to start and end at the beginning and end of a projected color channel. To do this an experimental setup is configured to learn the internal timing of the projector. Two timing signals are learned: the timing of the 60 Hz input video signal and the timing of the individual color channels within each video frame. The color wheel is made of a central hub that is connected to a motor. The glass filters around the hub generate each color channel for the DLP® projector. Fig. 2.3 shows an actual image of a color wheel. The color wheel is rotated so that it remains in sync with the video signal being fed to the projector. To keep track of the position of

Figure 2.3: Image of a DLP® color wheel.

the rotating wheel, an optical sensor tracks the rotation of the hub of the color wheel. The hub is equipped with a black non-reflective stripe that registers a peak response as it passes the optical sensor. In our system, we use the signal from this optical sensor to synchronize the cameras to the 60 Hz video signal being fed to all of the projectors.

The glass filters of the color wheel in each of our projectors are removed so that the projectors only project in grayscale, rather than color. This modification was done by Optoma representatives to ensure correct functionality of the projector after modification.

The 60 Hz frame sync signal from the projector is not sufficient to determine when to trigger the exposure of each camera. This is because the color wheel does not simply display the red, green, and blue channels in even sequential steps by default. An assortment of factors contribute to the way in which the projectors operate. To use the individual color channels to encode each frame of our projected pattern, the order and duration of each color channel needs to be known.

As explained previously, DLP® projectors generate full color images by rapidly projecting the individual color channels of an image. Due to the slow response time of the human eye, the colors are integrated to create a single colored image. Despite the eye's ability to integrate the individual color channels into a single color image, there can be some artifacts from the channel switching especially when the user's eyes shift around the projected image. To combat this problem, engineers have created projectors that switch between the color channels at a faster speed than once per video frame. In particular, our projectors cycle through the color wheel twice for every 60 Hz frame, or at 120 Hz. For the viewer, the projected image would present fewer artifacts from the color switching. However, this also reduces the window of time the cameras are exposed to each color channel

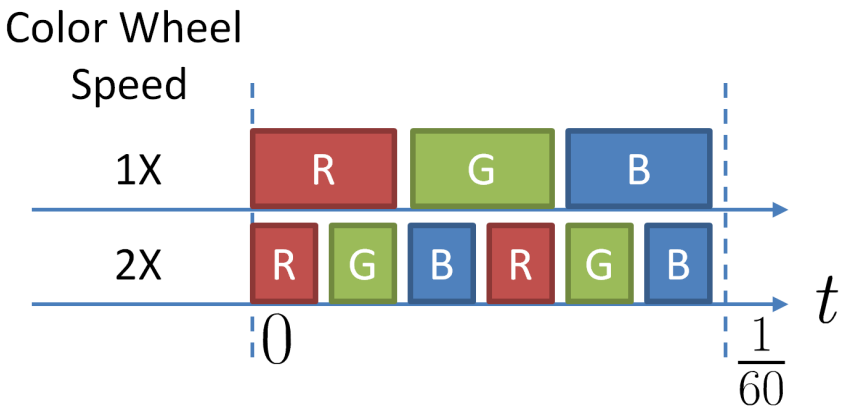


Figure 2.4: Example timings of a $1\times$ vs $2\times$ color wheel.

being displayed by a projector. Fig 2.4 shows an example of timing for color channels for a color wheel operating in $1\times$ or $2\times$ speed.

Additionally, the projector is designed to improve the vibrancy of the colors on the screen. This is done by using additional color channels beyond red, green, and blue. Typically, these additional color channels can include white and the secondary colors of yellow, magenta, and cyan. Our particular projectors have additional white and yellow color channels. The challenge with this is that these color channels are derived from the RGB values sent to each pixel. By displaying additional color channels, the colors that we are interested in projecting are modified and partially represented by the supplemental color channels. That is, rather than displaying three independent structured-light patterns in each of the color channels, the images are distorted and partially sent to multiple channels. Fortunately, when the projectors are operated in a "photo" mode, only the true RGB channels are used to display the image. Additionally, the color wheel of DLP® projectors are not divided into equal sections. This is due to a variety of factors including the spectrum of the lamp source and the selectivity of each of the color filters. The specifications of the color wheel's segmentation into color channels are not generally provided by original equipment manufacturers (OEMs). The makeup of the color wheel and the associated color processing are not shared to the public as OEMs consider this part of their competitive advantage. Additionally, in our case, since the color filters are removed from the projector, we have no initial notion of the color channel order or timing.

To learn the timing of the color channels, we use a photo diode. With a source voltage applied across two terminals of the diode, the voltage on a third terminal increases proportionally to the intensity of light projected onto the diode's surface. By using this diode, we learn the timing of each color channel in the projector.

To do this, an oscilloscope is synced to the 120 Hz color wheel rotation signal in one of the projectors. The photo diode is positioned in front of the projector, and solid color images, i.e red, green, blue, and white, are sent to the projector. For the primary colors at full brightness, we expect the photo diode to register a constant positive voltage during the corresponding color channel of

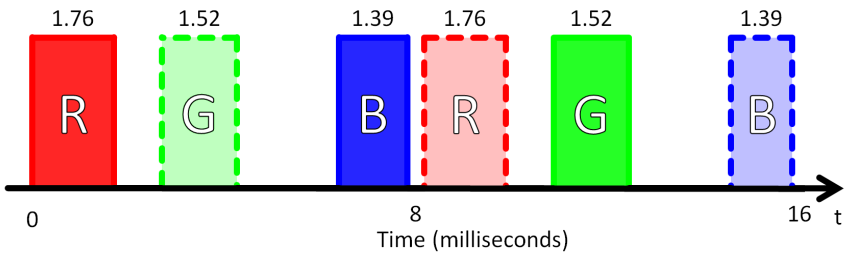


Figure 2.5: The duration of each color channel in a single video frame. The solid bordered color channels represent those that are captured during a single video frame.

the projector. By projecting each color, we are able to determine the instance and duration of each color channel relative to the 120 Hz rotation signal. Fig. 2.5 illustrates the timing for our projectors operating in "photo" mode.

Once the timing for the 60 Hz video signal and the individual color channels are known, the system can be operated. Fig. 2.6 shows the timing of the projectors and cameras in our system. As can be seen, only one projector illuminates the scene at any time. When a projector is on, it displays its red, green, and blue channels twice. The cameras that are at the same station as an illuminating projector capture images as the projector displays its patterns. The grayscale cameras capture an image for each of the red, green, and blue channels. The color camera integrates over a single set of the red, green, and blue channels to record an image for texture. The structured-light patterns appear as constant illumination by an observing camera that integrates over all color channels.

2.3.2 Microcontroller

As mentioned, the signal from the rotation encoder on the color wheel is used to identify the beginning of each 60 Hz frame. This signal serves as the reference point to synchronize the cameras to the projector. The signal is passed into a TTL logic buffering integrated circuit to convert the analog signal into a digital one. The digital signal is then fed into a microcontroller. Specifically, we use an OLIMEX LPC2103. The microcontroller is responsible for generating trigger signals to all cameras. The learned timing patterns in Fig. 2.4 are generated within the microcontroller program. The program waits for a rising edge on the input digital rotation signal and begins sending out triggers to all cameras. For the grayscale cameras, this means generating triggers during the red, green, and blue color channels within a single video frame. For the color cameras, the trigger occurs at the beginning of the frame and the exposure occurs over a complete red, green, and blue cycle, as shown in Fig. 2.6. During capture, the embedded program waits for a physical button press on the microcontroller to begin sending the trigger signals out to all the cameras.

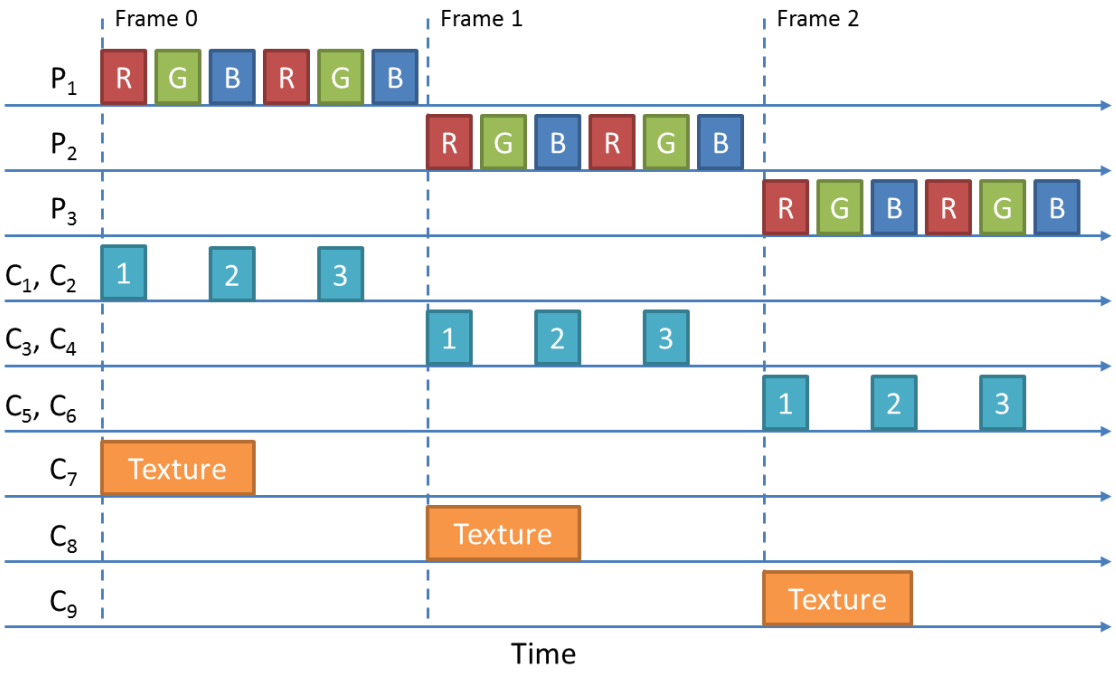


Figure 2.6: Timing of image projection and capture by devices in system.

2.3.3 Transmitting Trigger Signals

Since the trigger signals out of the microcontroller are only 0-3.3 V, they cannot be directly sent over the 10 m CAT5 cables to each of the stations. Therefore, digital-to-serial converter boards are used to convert the digital low-voltage signal to serial transmission voltage levels capable of being transmitted over a further distance. A board to step up the voltage is connected to the system microcontroller and a separate board is placed at each station to step the voltage back down to a level that can be directly fed in as an external trigger to the cameras.

2.4 Additional Hardware

In addition to the computing, capture, and projection devices, additional hardware described below is used to complete the system.

2.4.1 80/20 Aluminum Framing

Calibration is extremely important to generate consistent geometry from all of the stations. The precise intrinsic and extrinsic parameters must be known so the captured geometry can be properly generated and aligned. Since calibrating the system is a time-intensive process, it is important to have a sturdy and rigid platform to firmly hold the equipment in place once it is aligned. For our

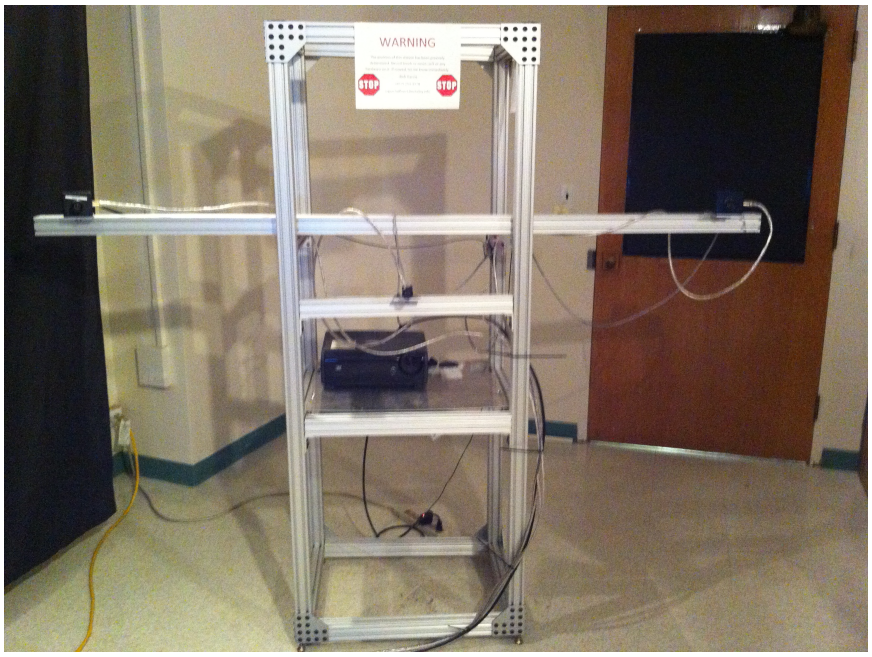


Figure 2.7: A single station in our multi-view structured-light system.

system, we use 80/20 aluminum extrusion to mount all of our cameras and projectors. The pre-cut aluminum pieces mount together using only a single hex wrench. The bulk and strength of the 80/20® ensures that once in place the hardware does not move. Each station weighs over 30 lbs, so they can be left in place without concerns about their unintentional movements.

The cameras are each mounted using brackets supporting the standard $\frac{1}{4}$" camera mounts found in tripods. Once fastened in position using hex bolts, the cameras are secure from minor bumping or misalignment. To mount the projector, a custom metal plate is fabricated. The plate is mounted to the 80/20 frame and the projector is mounted directly to the plate. Since projectors are often hung from ceilings, mounting sockets are available on the projector to secure it to the plate. A single station of the system is shown in Fig. 2.7

2.4.2 Cabling

The stations in the system are spread out to fully view the central capture scene from all directions. This layout causes some challenges because of the need to bring data to and from each station. Each station needs to receive trigger data for the cameras and video data for the projector. The trigger signals are sent over standard CAT5 cables. The video signal is sent over 30 ft. DVI cables. The station sends and receives data over IEEE 1394b cables to each camera. We chose 10 m cables to do this. Despite using long cables, we are within the maximum allowable distance for the DVI and 1394b protocols. Our serial level signals sent over CAT5 cables are able to travel this distance without any trouble. Any further length would have likely required us to add repeater hardware, which would be problematic for our synchronization.

2.5 Conclusion

The described system can synchronously drive patterns to all projectors and capture images from all cameras. The data gathering capacity of the system is used not only to capture the dynamic geometry of human subjects, but to also project patterns in the calibration process. Overall, we find that our system is able to consistently capture the data needed for further processing.

Chapter 3

Consistent Stereo-Assisted Phase Unwrapping Methods for Structured-Light Systems

In a stereo SL system capturing multiple poses of a dynamic scene, it is important for the phase unwrapped images of each camera at each time to be temporally and spatially consistent. In this chapter, we present two phase unwrapping methods in a stereo structured-light system to ensure viewpoint or temporal consistency. We refer to "viewpoint consistency" as unwrapping left and right camera phase images such that the corresponding scene points in each phase image have the same absolute phase. Similarly, we refer to "temporal consistency" as having temporally consecutive unwrapped phase images with similar phase values for all corresponding points between consecutive frames. The first method ensures viewpoint consistency when unwrapping images in the stereo SL system [38], and the second method ensures temporal consistency while unwrapping each camera over time [39].

In our first method, the phase images for both the left and right cameras in the stereo SL system are simultaneously unwrapped. Rather than unwrapping over the phase image of each camera individually, we unwrap by processing the pixels in the projector. There are several potential advantages to such a projector-domain phase unwrapping approach. First, operating on the projector pixels allows us to assign an absolute phase to corresponding pairs of pixels in the cameras. This ensures that the phase images are unwrapped consistently between camera views, and that the triangulated points from each camera are in agreement.

Second, in a projector-centric approach, the phase unwrapping algorithm only needs to be run once regardless of the number of cameras in the system. As such, it is likely to (a) be more computationally efficient, and (b) result in more consistent solutions with fewer discrepancies across cameras. Third, the computational complexity for a projector-based solution is proportional to the number of projector pixels rather than the number of camera pixels, as is the case in traditional camera-centric reconstruction. From a technological point of view, camera resolution is likely to increase at a much faster rate than projector resolution, projector-centric solutions are likely to be more computationally efficient.

In our second method, we unwrap temporally across multiple consecutive phase images from a single camera. Most existing SL systems for dynamic scenes process the captured data one frame at a time with no guarantee of temporal consistency in the unwrapped images. This could result in temporal discontinuities and artifacts when viewing the reconstructed point cloud over time. It is conceivable to both remove these temporal artifacts and improve phase unwrapping accuracy by unwrapping across time as well as space [8]. Thus, we propose merging stereo phase unwrapping with three-dimensional (3D) phase unwrapping to arrive at temporally consistent absolute phase volumes. By 3D, we refer to (X, Y, T) rather than the commonly used (X, Y, Z), where T represents time. Our proposed method in Section 3.4 is a variation of the 3D phase unwrapping algorithm proposed in [8]. Originally developed for MRI applications, the method assigns a quality measure to each pixel in the wrapped phase volume based on its local spatio-temporal second difference values. The edges between neighboring spatio-temporal pixels are also assigned quality measures based on the quality of the connected pixels. The phase volume is then unwrapped based on rank ordered edge qualities starting from the highest to lowest [8]. Since in our setup we assume a stereo-camera pair rather than a single camera, we develop a framework to integrate stereo phase unwrapping with the basic 3D phase unwrapping method in [8] in order to determine absolute phase. Besides providing temporal consistency, this approach is also effective in unwrapping scenes with large depth discontinuities or multiple spatially disjoint objects.

The outline of this chapter is as follows: Section 3.1 reviews the basic stereo phase unwrapping process. In Section 3.2, we present our proposed viewpoint-consistent phase unwrapping method. Section 3.3 provides an overview of the three-dimensional phase unwrapping method [8], and Section 3.4 presents our proposed temporally consistent phase unwrapping method, which is used in generating the results in Chapter 5. In Section 3.5, we describe the experimental setup along with results for both methods. Section 3.6 provides a comparison of our methods to related works. Section 3.7 discusses our choice of the two methods in capturing dynamic geometry, and Section 3.8 is conclusions.

3.1 Overview of Stereo Phase Unwrapping

In stereo phase unwrapping, a second camera is added to the traditional SL system made of a projector and a single camera. To maximize the camera coverage of the illuminated scene, the projector is positioned in between the two cameras, as shown in Fig. 3.1. Before processing captured data, the projector and the pair of cameras must all be calibrated [18].

During capture, the scene is illuminated by three consecutive phase-shifted sinusoidal images, as in [96]. After each camera captures the scene from its view, a wrapped phase image is computed as

$$\phi(x, y) = \tan^{-1}\left(\frac{\sqrt{3}(I_1(x, y) - I_3(x, y))}{2I_2(x, y) - I_1(x, y) - I_3(x, y)}\right), \tag{3.1}$$

where $I_i(x, y)$ represents the intensity of image i at image coordinates (x, y), and $\phi(x, y)$ represents the wrapped phase at (x, y). If a scene point is visible in both cameras, the phase measurement is

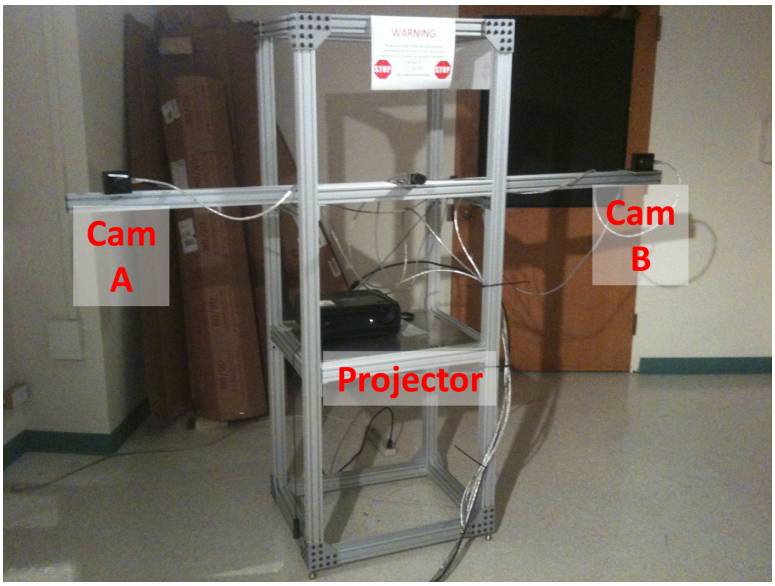


Figure 3.1: Configuration of stereo SL system.

independent of the viewing direction, except for saturated image regions corresponding to specular surfaces. Computing phase via (3.1) requires the projector to accurately project true sinusoidal patterns. Since the color-processing algorithms of the projector modify the intended projection patterns, we pre-distort the sinusoidal patterns so that the outgoing patterns are truly sinusoidal. Similar to [96, 51, 99, 91], we embed our three sinusoidal patterns into the RGB channels of a DLP® projector.

Even though the wrapped phase can be determined at each pixel of the camera, given M periods of sinusoids in the projection pattern, there is an M position ambiguity in each wrapped phase measurement [91]. To determine the absolute phase of a pixel, we search for the single absolute phase offset $2\pi m$ with $m \in \{0, 1, ..., M-1\}$, which must be added to its wrapped phase to obtain its absolute phase. We refer to m as the offset index. Since the M possible corresponding positions in the projector are known, the location of the M points can be triangulated, as shown in Fig. 3.2 [91]. These triangulated points all lie along the ray coming out of the camera, illustrated as camera A in Fig. 3.2. Fig. 3.3a shows an example of the wrapped phase image of camera A, from Fig. 3.2, with a pixel of interest P identified by a red dot. The extrinsic relationship between the stereo cameras in Fig. 3.2, as well as each camera's intrinsic parameters can be used to project the M 3D positions onto the second camera's image plane, as shown in Fig. 3.3b. By comparing the wrapped phase values at the M pixel locations to the first camera's phase value, the absolute phase of P can be estimated. In Fig. 3.3b, it is clear that point B is the corresponding match to the pixel P in Fig. 3.3a. Points A and C map to blank areas of the scene, and point D is projected outside of the image frame.

Even though this approach works for most pixels, in practice it results in occasional errors. Common causes are occlusions, calibration errors, motion errors, or noise in the calculated phase.

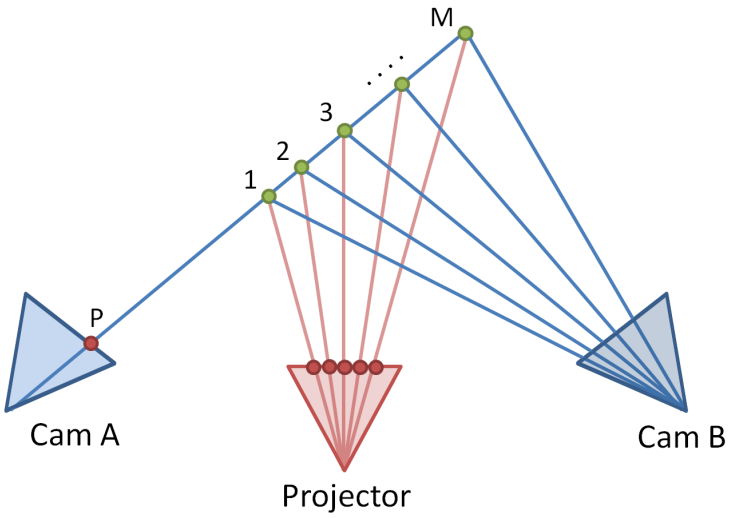
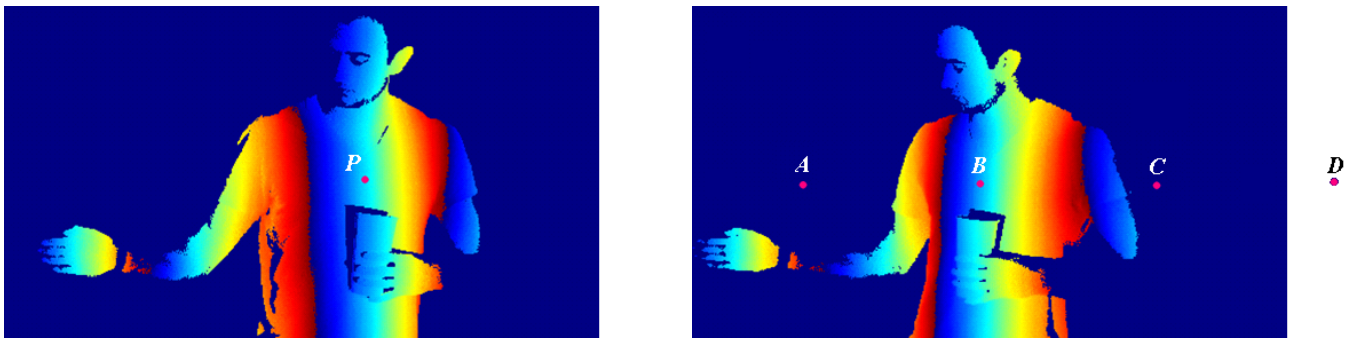
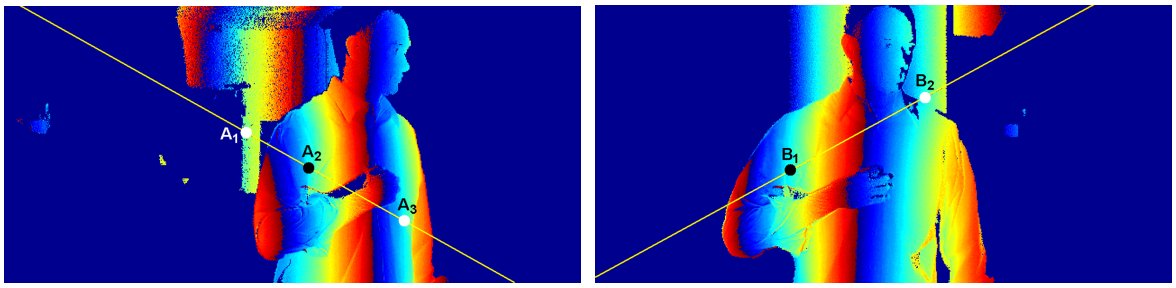


Figure 3.2: Triangulation of the M possible positions of pixel P in camera A.



(a) (b)

Figure 3.3: (a) Wrapped phase image of the left camera A with the pixel of interest identified by the red dot P; (b) wrapped phase image for the right camera B in a system with the M possible points projected onto the image.



(a) (b)

Figure 3.4: Wrapped phase images with epipolar line plotted. Each circle identifies a pixel with the same wrapped phase as the projector pixel; image from (a) camera A; (b) camera B.

In some cases, the phase of the point in camera A is closer to an invalid point than to the correct one. In [91], the stereo unwrapped points are segmented into continuous regions and loopy belief propagation is used to perform an energy minimization on a cost function to locally-smooth segmented regions.

3.2 Viewpoint-Consistent Phase Unwrapping

Our proposed viewpoint-consistent method consists of three steps. First, it is assumed that the cameras and projectors are calibrated with respect to each other. For each projector pixel, we determine its epipolar line in the two camera views, as shown in Fig. 3.4. By searching along the line in the images, we find correspondences between the camera views. The set of possible correspondences between the two views is treated as the set of labels for that pixel. Second, loopy belief propagation is applied to minimize an energy function that both quantifies the likelihood of each label and spatially smoothes a set of labels [30, 31]. Finally, the remaining points that are not solved via the stereo approach are unwrapped with respect to their already unwrapped neighboring points. In what follows, we describe each step in detail.

3.2.1 Determining Possible Correspondences

To solve for the absolute phase of both cameras, we process each pixel in the projected pattern. Given a pixel P in the projected image, the epipolar line corresponding to P in each camera view can be determined, illustrated as the lines in Figs. 3.4a and 3.4b. In each camera view, we identify the pixels along the epipolar line with the same phase as P, identified as $\{A_1, A_2, A_3\}$ and $\{B_1, B_2\}$ on the epipolar line in Figs. 3.4a and 3.4b, respectively. Then, the 3D position of each of these points is found by triangulating with point P, as illustrated in Fig. 3.5. All of the triangulated points lie along the ray out of the projector intersecting P.

Once the 3D position of each possible point is determined, the distances between all pairs of candidate points in camera A and B are calculated. For a pair of points corresponding to the true

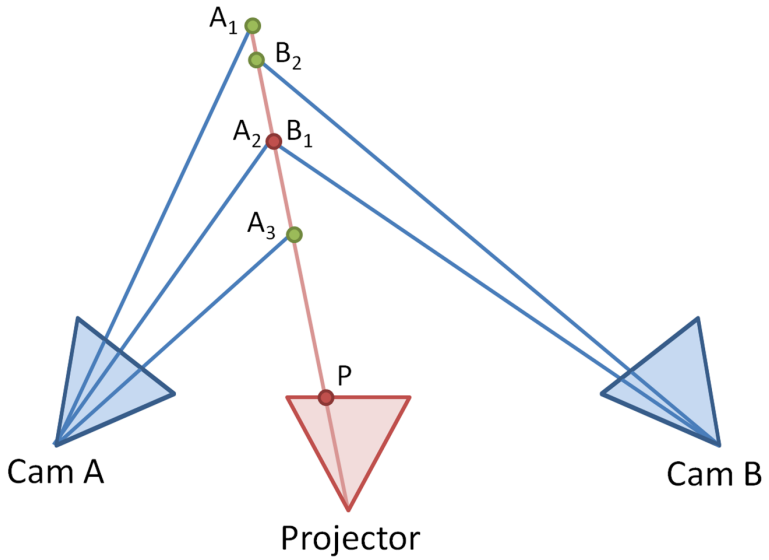


Figure 3.5: Illustration of the stereo-camera geometry used to identify correspondences across camera views.

correspondence, the triangulated 3D positions are likely to be near one another, as illustrated by the black dots in Fig. 3.4. Therefore, we can estimate the likelihood of a correspondence by the Euclidean distance between a pair of points.

3.2.2 Labeling via Loopy Belief Propagation

After the set of possible correspondences and their associated likelihoods are determined for each pixel in the projector, an energy minimization framework is applied to determine the best label for each projector pixel. We only include projector pixels in the graphical model for which there is at least one pair of correspondences with a distance smaller than a set threshold. If there are no possible correspondences within this threshold, the projector pixel is not likely viewable from both cameras. The set of labels for each projector pixel is all possible pairwise correspondences between the detected points in each camera view. Specifically, in Figs. 3.4a and 3.4b, the set of labels for pixel P are $\{(A_1, B_1), (A_1, B_2), (A_2, B_1), (A_2, B_2), (A_3, B_1), (A_3, B_2)\}$.

Similar to [91], we minimize a cost function consisting of a data and smoothing term:

$$E(f) = E_d(f) + E_s(f), \tag{3.2}$$

where f represents the labeling set for the pixels in the projector. The data component of the cost function comes from the likelihood of the correspondence as determined by the distance between the points in 3D space:

$$E_d(f) = \sum_{p_m \in F, p_m = \{A_m, B_m\}} \min(\beta, ||l_{3D}(A_m) - l_{3D}(B_m)||), \tag{3.3}$$

where p_m denotes the label of a pixel within the projector image F corresponding to a pixel A_m in camera A and B_m in camera B, β is a threshold to limit the penalty from occlusions, and $l_{3D}(A_m)$ denotes the 3D location of the labeled point A_m. The smoothing component is chosen to enforce a solution that minimizes large discontinuities within the scene. In most SL systems, the depth of the scene is assumed to vary slowly. This means that neighboring pixels within the camera image should have similar depths, and thus similar phase values. Since our method operates on the projector pixels, the projector pixels are assumed to vary smoothly. The basic intuition is that if two neighboring pixels in the projector illuminate the smooth scene surface, the two pixels in the camera image corresponding to these points should also be neighbors. This assumption fails when a scene discontinuity lies between the two examined pixels in the projector. Therefore, we enforce the following smoothness term:

$$E_s(f) = \sum_{\forall p_m, p_n \in F : p_m \in \mathcal{N}(p_n)} \min(\alpha, ||l_{2D}(A_m) - l_{2D}(A_n)||) + \min(\alpha, ||l_{2D}(B_m) - l_{2D}(B_n)||), \tag{3.4}$$

where p_m and p_n are neighboring points in the projector image, and A_m and B_m are the corresponding points for the pixel p_m in cameras A and B respectively. The function l_{2D} is the 2D image coordinates of the identified point, and $\mathcal{N}(p_n)$ is the set of four pixels that border p_n. The threshold α is set to minimize the penalty to occlusions.

To determine the label for the projector pixels that minimize (3.2), we apply loopy belief propagation [72]. The message passed between two neighboring pixels is constructed as

$$\begin{aligned} m^i_{p_m \to p_n}(f_n) = arg\,min_{f_m} & \min(\beta, ||l_{3D}(A_m) - l_{3D}(B_m)||) \\ & + \min(\alpha, ||l_{2D}(A_m) - l_{2D}(A_n)||) \\ & + \min(\alpha, ||l_{2D}(B_m) - l_{2D}(B_n)||) \\ & + \sum_{p_j \in \mathcal{N}(p_m) \setminus pn} m^{i-1}_{p_j \to p_m}(f_m), \end{aligned} \tag{3.5}$$

where f_n is the label assigned to pixel p_n and $m^i_{p_m \to p_n}(f_n)$ denotes the message passed from p_m to p_n in the i^{th} iteration. To minimize the energy cost function, messages are generated and passed to their neighboring pixels in each iteration. After a predetermined number of iterations is reached, the final label for each individual pixel is chosen by summing the final messages entering that pixel and choosing the label with the lowest cost. The final labeling is used to assign absolute phase offsets to the phase images in the two camera views. For each pixel in the projector, the energy minimization provides the most probable labeling. The final label of each projector pixel specifies the location where the projected pixel intersects the image plane of cameras A and B. Since the absolute phase for each point in the projected pattern is known, we can assign the identified image points with the proper absolute phase offset.

Noise in the captured images can lead to small errors in the decoded phase for each pixel. When searching for possible points of correspondence along the epipolar lines in the cameras, a small tolerance in the matched phase value is allowed to account for this noise. Because of this

tolerance and the small errors in the system calibration, it is possible for multiple projector pixels to map to the same pixel in the camera, or for a pixel in the camera to have multiple neighboring correspondences in the other camera. For these cases, we keep the match with the lowest final computed cost. These small errors rarely lead to errors in our estimated absolute phase offsets since we only keep the absolute phase offset that is determined from the match rather than the absolute phase value of the projector's pixel.

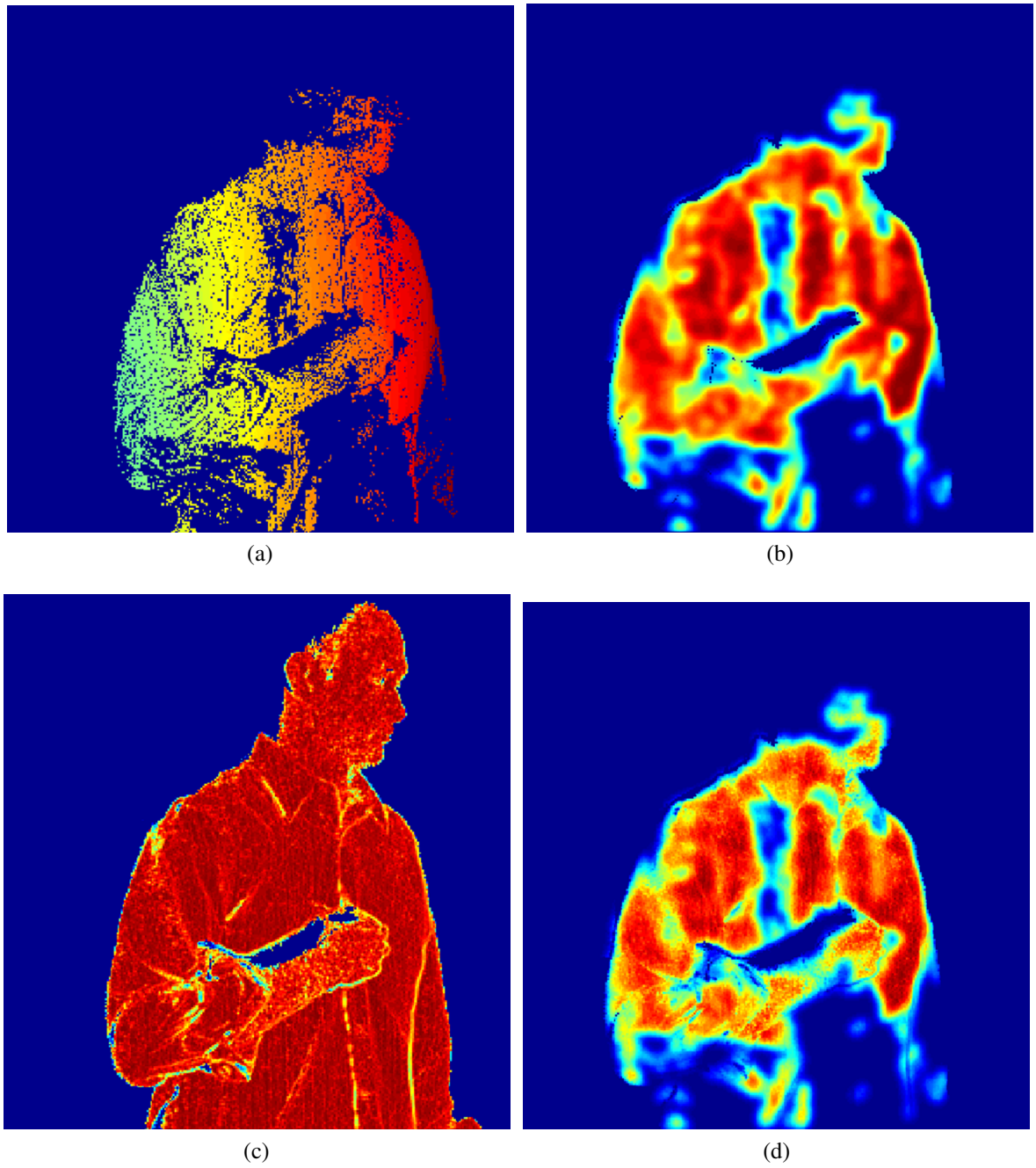
3.2.3 Filling in Missing Phases

Unlike the previous two steps where the processing is done in the projector domain, in this step, missing phase values in camera images are filled in. As mentioned previously, our graphical model only provides absolute phase values for projector pixels that likely have a correspondence. Since not all projector pixels can be labeled, not all the pixels in the camera images can be labeled either. Nevertheless, it is possible to determine the absolute phase of these remaining pixels by local spatial phase unwrapping with respect to the already unwrapped points.

Before unwrapping the remaining points, a quality map is derived for each camera's phase image. In two-dimensional spatial phase unwrapping, quality maps quantify the confidence in unwrapping a local region of an image and are often based on local rates of change in the phase [42]. Regions of the phase image with slowly varying phase are unwrapped first since they result in fewer errors. In our implementation, the quality map is generated by computing local derivatives in the original wrapped phase image and also by calculating the density of points with assigned phases via stereo unwrapping. Fig. 3.6a illustrates the camera points with phases from the stereo observations described in this section. Fig. 3.6b illustrates the local density of absolute pixels from Fig. 3.6a, Fig. 3.6c the local derivatives, and Fig. 3.6d the final quality map. The final quality map is generated by point wise multiplying the density image and the local derivative image. Once the quality map is generated, the points without an assigned absolute phase are unwrapped. Specifically, each pixel in the quality map is assigned a value from $0 - 1$, and the quality values are placed into bins that span $0 - 1$. Starting with the highest quality bin, we iterate through the non-unwrapped pixels in that bin and unwrap camera pixels that are neighbors to already absolute unwrapped pixels. After all pixels in a bin are unwrapped, the pixels in the next quality bin are unwrapped. A list of the remaining unwrapped pixels is maintained, and the unwrapping process continues until the list of remaining pixels is empty.

3.3 Overview of Three-Dimensional Phase Unwrapping

Multidimensional phase unwrapping has been an area of active research for many years, with a substantial body of work dedicated to unwrapping in two dimensions [42, 8]. Applications for these methods include synthetic aperture radar, medical imaging, and SL systems [42]. In recent years, many of the two dimensional techniques have been extended to three and higher dimensions [8].



(a) (b)

(c) (d)

Figure 3.6: Quality map components: (a) pixels unwrapped using the approach of Sections 3.2.1 and 3.2.2; (b) density of stereo-unwrapped points; (c) local derivative measure; (d) final quality map.

The basic idea behind most multi-dimensional phase unwrapping algorithms is to integrate along paths of neighboring phase values in order to determine the relative phase between all pixels. This is a trivial problem when the true phase difference between neighboring values is less than $|\pi|$ and noise does not push the resulting phase values outside this range. Since, in most applications, this assumption is invalid, phase unwrapping algorithms are designed to minimize the impact of these erroneous regions.

Most 3D phase unwrapping algorithms can be classified into one of three groups: 1) global-error minimization, 2) residue detection, and 3) quality guided methods [42]. Of these basic approaches, quality guided methods have proven to be both computationally efficient and robust [8]. The basic idea behind them is to quantify the likelihood that two neighboring pixels can be correctly unwrapped with respect to each other. An edge is defined to exist between all neighboring pixels and a measure of quality is assigned to each edge [8]. If two neighboring pixels have similar phase values, the quality of that edge is considered to be high. In contrast, two pixels with greater phase difference would have a lower quality edge. Although, basic gradient calculations between pixels have been used to generate quality maps [42], other methods, such as a second difference quality map, have been found to be more robust [8].

The second difference value for each pixel (i, j, k) in the phase volume can be computed as

$$SD_{ijk} = \sqrt{H_{ijk}^2 + V_{ijk}^2 + N_{ijk}^2 + \sum_{n=1}^{10} D_{n,ijk}^2}, \tag{3.6}$$

$$H_{ijk}^2 = \gamma[\phi_{i-1,j,k} - \phi_{i,j,k}] - \gamma[\phi_{i,j,k} - \phi_{i+1,j,k}], \tag{3.7}$$

$$V_{ijk}^2 = \gamma[\phi_{i,j-1,k} - \phi_{i,j,k}] - \gamma[\phi_{i,j,k} - \phi_{i,j+1,k}], \tag{3.8}$$

$$N_{ijk}^2 = \gamma[\phi_{i,j,k-1} - \phi_{i,j,k}] - \gamma[\phi_{i,j,k} - \phi_{i,j,k+1}], \tag{3.9}$$

where H, V, and N represent the horizontal, vertical, and normal, i.e., temporal, second differences via discrete Laplacians respectively. In addition, $\phi_{i,j,k}$ represents the wrapped phase at (i, j, k) and γ is a function that wraps its parameter to a value within $[-\pi, \pi)$. The ten diagonal second difference components $D_{n,ijk}^2$, $n \in \{1, ..., 10\}$ in (3.6) are computed through all of the diagonal paths in the $3\times3\times3$ volume centered on the pixel of interest. The squared second derivative identifies pixels where a discontinuity occurs. If there are no discontinuities, the phase increase should be approximately linear, resulting in a squared second difference value near zero; on the other hand, if the squared second difference value is large, there is a strong likelihood of a discontinuity. Using this quality map, it is possible to identify pixels with spatial or temporal discontinuities.

The quality of a pixel is inversely related to the sum of second difference values. Once a quality value is assigned to each pixel in the 3D phase volume, the quality of each edge is simply defined as the sum of the quality of each pixel connected to the edge [8]. The quality of edges determines the order in which the edges in the volume are unwrapped by sorting them from highest to lowest.

When an edge is evaluated, the pixels connected to the edge are unwrapped with respect to one another, forming links in a chain. Each chain has a pixel that is designated the lead pixel. When the first link of a chain is created, the lead pixel is defined as the pixel with the highest quality. As

each new pixel is added to a chain, the new pixel is unwrapped with respect to the connected pixel along the evaluated edge. Each newly added pixel is then given a phase value relative to the lead pixel in the chain. When two chains are merged, each pixel in the smaller chain is assigned a phase relative to the lead pixel in the larger chain [8]. Chains grow in connected 3D volumes and their actual shape is unimportant as long as the phase of each pixel in the chain is defined with respect to the lead pixel.

While this 3D algorithm is capable of accurately unwrapping the phases of all points within a volume, it does not necessarily result in absolute phase. Since each absolute phase is assigned to a column of pixels within the sinusoidal pattern, correct triangulation is only possible when the absolute phase is known; thus without a method to recover the absolute phase, the 3D unwrapping algorithm is useless in the SL system.

Additionally, the algorithm in [8] is only capable of unwrapping scenes with one contiguous, connected region. While this works in MRI, the application domain in which this algorithm was originally designed, it is not applicable to depth recovery for scenes with multiple disjoint objects as it could incorrectly assign relative phases to each object. This is because in [8] all edges between pixels with a wrapped phase are unwrapped with respect to one another regardless of the quality of the edges. For example, consider a scene with a ball traveling in front of a flat board. As the ball moves, the pixels that fall along the edge of the ball and the background receive low quality measures due to the motion and depth discontinuity. Therefore the algorithm [8] is likely to assign erroneous phases to those pixels. However, within the ball, as the local depth is not changing significantly, we would expect these pixels to have high quality edges. Unwrapping the background and ball independently prevents the two regions from being incorrectly unwrapped. As long as the absolute phase of each region can be determined, e.g., through stereo phase unwrapping, there is no need to unwrap across low quality edges.

The shortcomings of the phase unwrapping algorithm in [8] are a direct consequence of the fact that it has been developed for applications in which only a single observation exists for each phase value. In an application such as MRI, since each point is only measured or observed once there is no choice but to unwrap all pixels with respect to one another. However, in an SL system with a second camera to disambiguate the absolute phase of each pixel, unwrapping can be restricted to high-quality edges, i.e., those edges connecting pixels with similar phase values, thus reducing the error. In the following section, we exploit this to develop a method for combining stereo phase unwrapping with 3D phase unwrapping to overcome some of the shortcomings of [8].

3.4 Temporally Consistent Phase Unwrapping

In this section, we develop a probabilistic framework for stereo phase unwrapping in order to augment the functionality of the 3D phase unwrapping algorithm of [8] to make it capable of determining absolute phase.

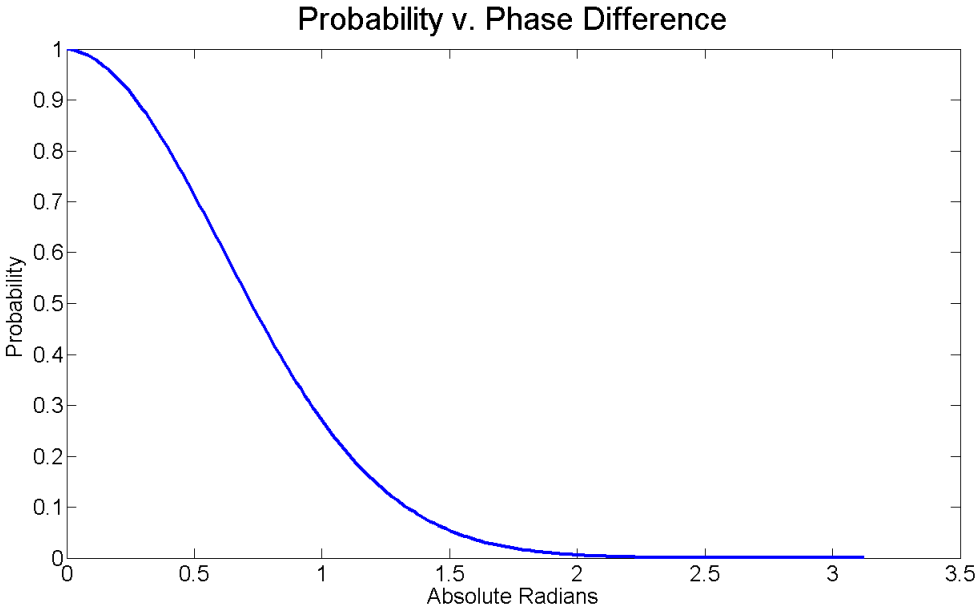


Figure 3.7: Converting absolute phase differences to probabilities.

3.4.1 Determining a Pixel's Absolute Phase Offset Probabilities

As mentioned earlier, the likelihood of a pixel having a specific absolute phase offset is determined by triangulating the M possible corresponding points between camera A and the projector, as shown in Fig. 3.2, projecting these M points onto the second camera's image, as shown in Fig. 3.3b, and comparing the phases of the M projected points to the phase of the original pixel P in camera A. With pixel's taking on wrapped phase values from $-\pi$ to π , if the original pixel P and one of the corresponding M projected pixels have a phase difference of 0, they are likely to be a correct match; likewise, if the pair differ by $|\pi|$ then they are unlikely to be a correct match[1]. One way to compute the likelihood for each of the M possible absolute phase offsets is to divide the phase difference of the corresponding pixels by π. However, this likelihood measure does not sufficiently penalize large phase differences. Rather, we propose using a Gaussian function to assign likelihoods to phase differences, as shown in Fig. 3.7.

In recovering the absolute phase of each pixel, not all of the M corresponding points necessarily project onto valid regions in the second image as shown earlier in Fig. 3.3. Some could fall on portions of the image where there are no wrapped phase values, as indicated by dots A and C in Fig. 3.3b, and some might fall outside the viewable region of the camera, as indicated by dot D in Fig. 3.3b. We assign σ_B and σ_E as the probabilities for 3D points projected outside the boundary and to an empty "phase-less" region of the image, respectively. Both σ_B and σ_E are set greater than zero to ensure we do not penalize non-overlapping views or occlusions between the cameras too

[1]When calculating the difference between the phase values, the wrapping function γ, from Equations (3.7)-(3.9), is used to ensure the difference is in the range $[-\pi, \pi)$

harshly during unwrapping. We note that some pixels near the edges of camera A's image may not be visible from the second view. For these cases, we assign a probability between σ_B and σ_E. Once the probabilities for all of the possible absolute phase offsets are determined, they are normalized to generate a probability distribution for the absolute phase offset of pixel P in camera A.

3.4.2 Using Pixel Probabilities to Build Chains

In [8], each time a pixel is connected to an existing chain, or when two chains are combined, the newly added pixels are unwrapped relative to their linked neighbors. The relative phase between all pixels in a chain is used to determine the phase difference between the lead pixel and each pixel in the chain. Since we only connect "high quality" pixels with reliable edges, all the pixels in a chain require the same phase offset to convert their relative unwrapped phase to the absolute phase[2]. To determine the absolute phase of the chain, we search for the single absolute phase offset $2\pi m$, where $m \in \{0, 1, ..., M-1\}$, which must be added to the chain's lead pixel's wrapped phase to obtain its absolute phase. We refer to m as the offset index for that chain. If the absolute phase of the lead pixel and the relative phase difference between the lead pixel and each pixel in the chain are known, then the absolute phase for each pixel in the chain can be determined.

The key to ensuring the pixels in the chain have the correct absolute phase is to use the phase probability distributions of each pixel in the chain to determine the absolute phase of the lead pixel. As explained shortly, the basic idea is to use the phase probability distribution of each newly added pixel in the chain to update the probability distribution of the offset index for the lead pixel of the chain. Once the chain is completed, the lead pixel offset with the highest likelihood is chosen to be the phase offset for the entire chain.

Similar to the unwrapping method in [8], we start by assigning quality values to all of the pixels and edges. The quality value of an edge is the sum of the quality of the two pixels connected to the edge. Once the quality of all edges has been determined, the pixels are ranked in quality from highest to lowest.

To start the unwrapping process, the most reliable edge is identified and the pixels connected to the edge are unwrapped with respect to each other to form a two pixel chain. We set the pixel with higher quality as the lead pixel of the chain. We need to combine the probability distributions of the absolute phase offset for these two pixels in order to determine the probability distribution for the absolute phase offset of the leading pixel, or equivalently the whole chain. One way to do this is to add the log probabilities of the two pixels. However, this can only be done after accounting for the phase difference between the two connected pixels.

To illustrate this, consider two possible chains, each with two elements. The first chain, shown in Fig. 3.8a, has a lead pixel with a wrapped phase of $\pi - 2\delta$ and a second pixel with a phase of $\pi - \delta$, where $0 < \delta << \pi$. Local unwrapping would imply that the two points are from the same sinusoidal period. This means that the log[3] probability distributions could be directly added

[2]In essence, we are assuming all the pixels in a chain are part of a contiguous object in a scene without depth discontinuities.

[3]Due to numerical stability issues, we opt to add the log of probabilities rather than multiplying them.

together to refine the offset index state of the lead pixel:

$$\log P(c_i = m) \propto \log P(p_j = m) + \log P(p_k = m), \tag{3.10}$$

where c_i is the offset index state of the lead pixel of the i^{th} chain given the influence from the connected pixels, p_j is the offset index state of the lead pixel and p_k is the offset index state of the second pixel, with offset index $m \in \{0, 1, ..., M-1\}$.

Now consider the second case, shown in Fig. 3.8b, where the lead pixel has a value of $\pi - \delta$ and the second pixel has a value of $-\pi + \delta$, where $0 < \delta << \pi$. For this case, the second pixel is unwrapped into a period that is outside the period corresponding to the lead pixel. Specifically, the second pixel has an absolute phase offset that is 2π greater than the lead pixel, or equivalently an offset index that is one greater than the lead pixel. Therefore, when we add the log probabilities to determine the absolute phase distribution for the whole chain, we must shift the individual pixel probability distributions, as shown in Fig. 3.8b:

$$\log P(c_i = m) \propto \log P(p_j = m) + \log P(p_k = m + 1). \tag{3.11}$$

The same procedure is used when adding a single pixel to an already formed chain. As each new pixel is added to the chain, the absolute phase offset distribution of the new pixel is first properly shifted to reflect the relative phase difference between the lead pixel and the new pixel; then the log probabilities for the new pixel are added to a running sum for the chain. In essence, this represents the contribution of the phase probability distribution of the new pixel to the overall phase offset of the chain it joins.

In summing log probabilities as pixels are added, we keep a count of the number of pixel probabilities added to each possible offset index. Due to the shifting, each possible offset index might have a different number of log probabilities summed together. For example, in Fig. 3.8b, the lead, left, pixel in the chain has two log probabilities summed together for $k_1 \in [0...M-2]$, shown in blue, but only one for $k_1 = M - 1$, shown in pink. We know the second pixel in the chain cannot have an offset index of 0, shown in green, since this would imply the lead pixel to have an offset index of -1. This is not possible since $m \in \{0, 1, ..., M-1\}$. To determine the final absolute phase offset for the lead pixel in the chain, the average log probability for each absolute phase offset is computed and the absolute phase offset with the maximum average log probability is selected. This absolute phase offset is then used to determine the absolute phase for all the pixels in the chain.

For the case where two chains are connected together, the smaller chain is unwrapped relative to the larger chain. That is, each pixel in the smaller chain is unwrapped relative to the lead pixel in the larger chain. In addition, the set of summed log probabilities for the smaller chain is shifted to reflect the relative phase offset between the lead pixels of the two chains. The shifted log probabilities are then added to the summed log probabilities of the larger chain. This is similar to the probability distribution shifting and summing explained in the two pixel chain examples in Equations (3.10) and (3.11) except that it is being carried out with respect to the lead pixels in the two chains rather than individual pixels.

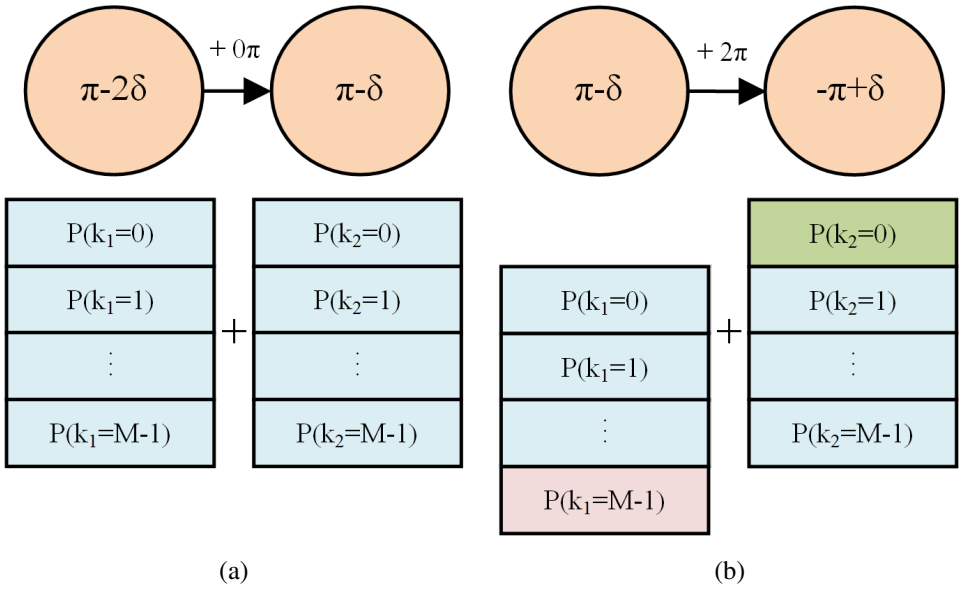


Figure 3.8: The addition of log probabilities for two pixels (a) in the same period and (b) from different periods.

3.4.3 Removing Edges

The advantage of using stereo phase unwrapping to anchor each chain is that it is no longer necessary to unwrap across all edges, which is done in [8]. This allows us to determine the absolute phase of spatially disjoint objects in the scene; in addition it prevents us from unwrapping across unreliable low quality edges that may have true phase differences greater than $|\pi|$. Local unwrapping fails when the true phase differences between neighboring pixels are greater than $|\pi|$.

By limiting unwrapping to edges with quality greater than a minimum threshold, we reduce the probability of unwrapping across spatial or temporal discontinuities. By adding the time dimension to unwrapping, small regions are unwrapped more accurately since more observations are merged over time.

In our implementation, we choose a minimum quality value to be met in order for the edge to be considered valid. Even though this results in several smaller chains at the end of the unwrapping process, the points within each chain are more likely to be correctly unwrapped. In general, erroneous unwrapping only occurs along the edges between discontinuous regions.

3.5 Experimental Setup and Results

To test the presented algorithms, we have developed an SL system similar to the one in [91], but without using the the color camera for texture capture. The system consists of two Point Grey Dragonfly Express® cameras with a resolution of 640×480 and 1394b ports. We use an Optoma

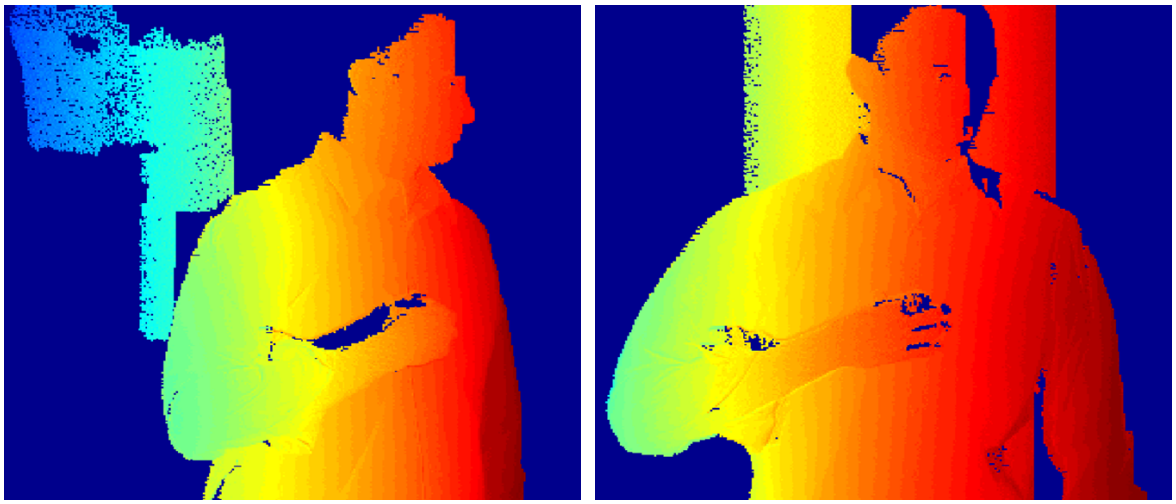
TX780 projector with a resolution of 1024×768, operating at 60 Hz. Similar to many other SL systems utilizing a DLP® projector, the color wheel has been removed to increase the rate at which grayscale patterns are projected to 180 Hz [51, 50].

To illustrate the effectiveness of our proposed algorithm, we have captured several data sets. Once the correspondences are found between the camera and projector, the depth of each pixel is determined via triangulation. We present results on the accuracy of the estimated absolute phase of the scene, or equivalently the accuracy of the reconstructed point clouds.

Our current implementations of the viewpoint and temporally consistent unwrapping algorithms do not operate in real time. Generating the stereo matching data for both the viewpoint and temporal unwrapping methods requires several points to be triangulated for each pixel in the phase volume. The independent nature of these calculations allows for the stereo matches to be computed using a highly parallelized GPU algorithm. For the viewpoint algorithm, the greatest bottleneck is in solving for the correspondences between camera views using loopy belief propagation. To speed up processing, a more efficient parallel loopy belief propagation algorithm could be used. Once the set of available correspondences are solved, unwrapping the remaining pixels is completed in linear time. For the temporal unwrapping method, the bottleneck is due to iterating through all the pixels in the phase volume. This step cannot be easily parallelized since the chains are built up throughout the entire phase unwrapping volume. In our non-optimized implementations, the unwrapping process requires on average less than 10 seconds per frame.

3.5.1 Viewpoint-Consistent Unwrapping Results

Figs. 3.9a and 3.9b show the unwrapped results using our proposed viewpoint-consistent method of Section 3.2 for the camera images shown in Figs. 3.4a and 3.4b, respectively. The algorithm is clearly effective in properly unwrapping the illustrated phase images. For comparison purposes, Fig. 3.10 shows the unwrapping results for Fig. 3.4a using our implementation of the approach described in [91], but without using the left to right consistency check. The consistency check is likely able to remove some outlying pixels, but would not correct the mislabeled segments encircled in Fig. 3.10. Clearly, the reconstruction quality of Fig. 3.9a is higher than that of Fig. 3.10. In a sample sequence of 700 frames, our method provides more accurate results than [91] in 80% of frames, and performance comparable to or better than [91] in 96% of frames. More specifically, in a sample sequence of the 700 phase images, unwrapping via [91] results in an average error of 3.2% of incorrect foreground pixels compared to 0.5% for our method. In our experiments, we have found that nearly all unwrapped images experience at least a few incorreclty unwrapped pixels. Unlike [91], in our experiments we do no compensate for scene motion. The amount of motion in our experiments has been deliberately limited in order to reduce the errors in decoding the phase value at each scene point. Despite our efforts, we do find that some of the decoded phase values are incorrect, especially along the edges of the scene. For these cases, we find that our algorithm is still able to unwrap the edge pixels assuming the error is not too large. A video comparing the two methods can be found on our website at [1]. The merged point cloud generated from the two unwrapped phase images in Figs. 3.9a and 3.9b is shown in Fig. 3.11. As seen, the merged point



(a) (b)

Figure 3.9: Final unwrapped images using our algorithm of Section 3.2 for (a) camera A, corresponding to Fig. 3.4a; (b) camera B, corresponding to Fig. 3.4b.

cloud is denser than either component; furthermore, the two components are reasonably aligned, and provide increased scene coverage by alleviating occlusion.

3.5.2 Temporally Consistent Unwrapping Results

We show our algorithm's ability to handle scenes with disjoint objects. Fig. 3.12a shows the unwrapped phase of a scene with disjoint objects using our proposed algorithm in Section 3.4. Fig. 3.12b illustrates how the algorithm in [8] fails to correctly unwrap the disconnected regions inside the ellipse. In the phase image, a cone shaped object is lifted off of a box. The cone and box objects are positioned at approximately the same depth, so they should have nearly the same unwrapped phase values as shown in Fig. 3.12a. Since there are no connections between the two regions, there is no way for the algorithm in [8] to determine the phase difference between the isolated segments. Our method handles this by using the added information from the stereo camera as long as the points are visible in both cameras. While existing SL approaches using stereo cameras enforce only spatial smoothness during the phase unwrapping process [91], our approach enforces both spatial and temporal smoothing. To characterize the advantage of temporal smoothing, we have compared our algorithm with the stereo phase unwrapping method presented in [91]. Since our method only determines the unwrapped phase of one camera at a time, to do a fair comparison we do not implement the left-to-right and right-to-left consistency check described in [91]. This type of consistency check could easily be applied to our method to provide the same benefits as the approach in [91]. In the loopy belief propagation implementation, we process 10 iterations before generating the final labeling for all segments. Figs. 3.13(a)-(b) and 3.14(a)-(b) show two successively unwrapped phase images resulting from the algorithm in [91] and our proposed al-

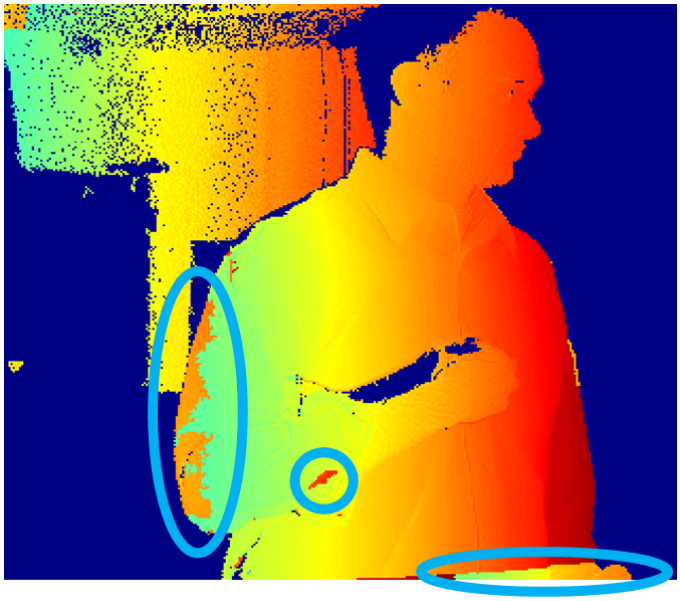
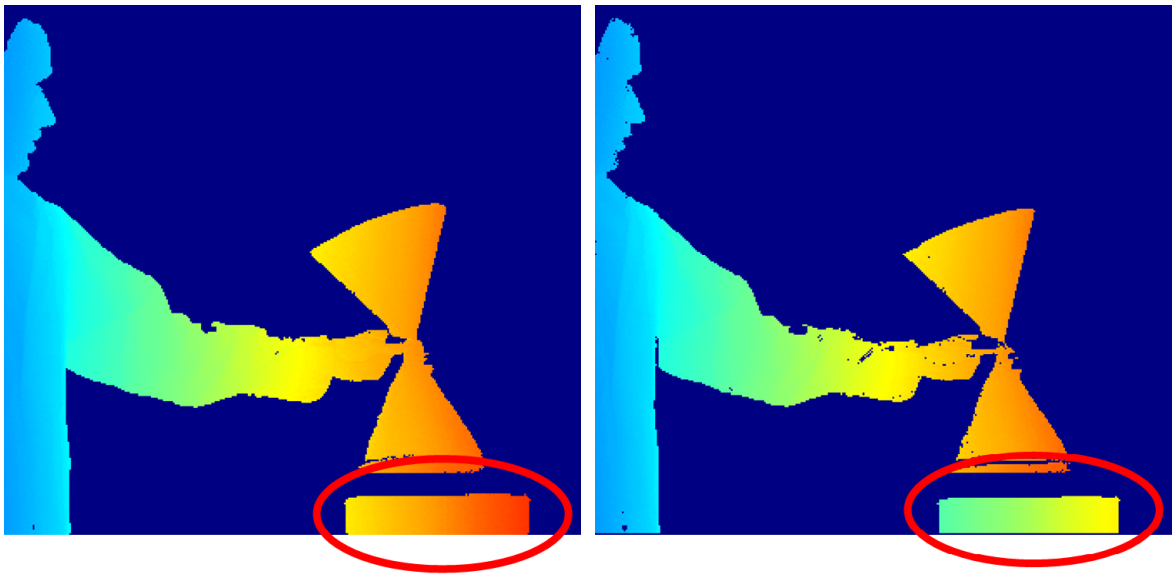


Figure 3.10: Unwrapped image for camera A, corresponding to Fig. 3.4a, using the method in [91]. Erroneous unwrapped regions are circled.

Figure 3.11: Merged camera A (red) and B (blue) points clouds resulting from unwrapped phases of Fig. 3.9.

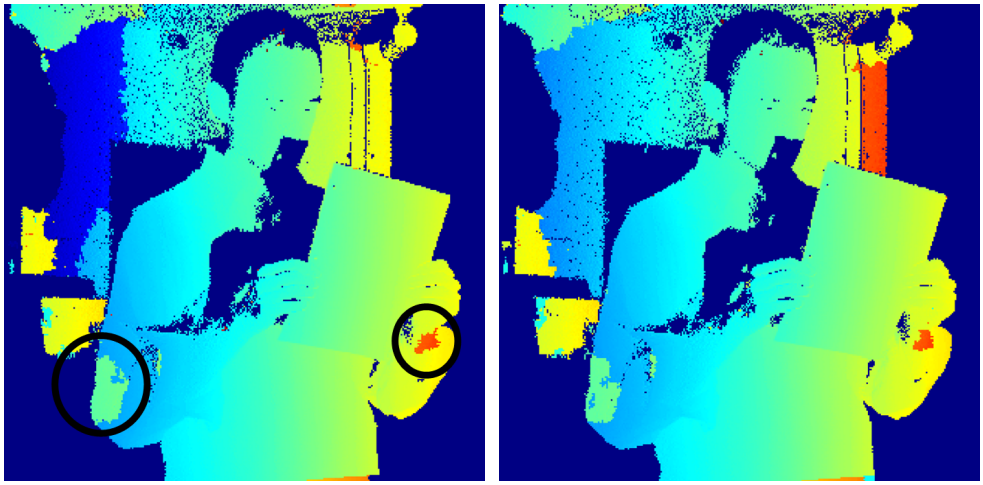


(a) (b)

Figure 3.12: (a) Correctly unwrapped scene using our proposed algorithm of Section 3.4; (b) incorrect unwrapping for the algorithm in [8].

gorithm, respectively. As seen, each image in Fig. 3.13(a)-(b) suffers from more artifacts than the corresponding one in Fig. 3.14(a)-(b). More importantly, the images in Fig. 3.14(a)-(b) are more temporally consistent than those in 3.13(a)-(b). The unwrapped phase difference between the successive images in Figs. 3.13(a)-(b) and 3.14(a)-(b) are shown in Figs. 3.13c and 3.14c, respectively. As seen, the areas with phase difference in Fig. 3.13c are significantly smaller and less noticeable than in 3.14c. Specifically, in Fig. 3.13c, 15.3% of pixels are not consistent over time as compared to 3.4% in Fig. 3.14c; some of the 3.4% corresponds to actual motion in the scene. Visual inspection of the two phase video sequences also confirms the temporal consistency of our approach as compared to spatial only smoothing which exhibits a significant amount of flicker. The comparison video can be found on our website at [2].

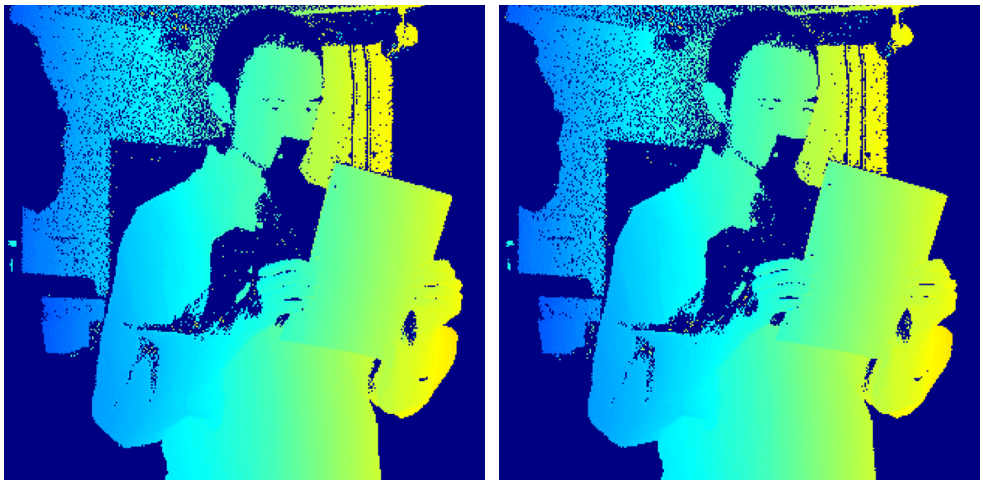
In [8], the edges are sorted and unwrapped sequentially according to the quality of the connection between pixels. Since the lower quality pixels are unwrapped at the end, any potential errors only affect the few pixels that follow in unwrapping order. In our setup, it is not necessary for every pixel to be unwrapped relative to all of its neighboring pixels since we combine the absolute phase positioning from the stereo camera. Therefore, our proposed algorithm eliminates the need to unwrap between pixels with low quality edges. Fig. 3.15 shows the same scene unwrapped with our proposed algorithm and that of [8]. In Fig. 3.15b resulting from [8], the black circled regions contain points that are unwrapped incorrectly. The phase in these regions changes rapidly both spatially and temporally due to the discontinuity between the two planes and the high motion of the plane. As seen, the low quality edges do not allow for accurate unwrapping.



(a) (b)

(c)

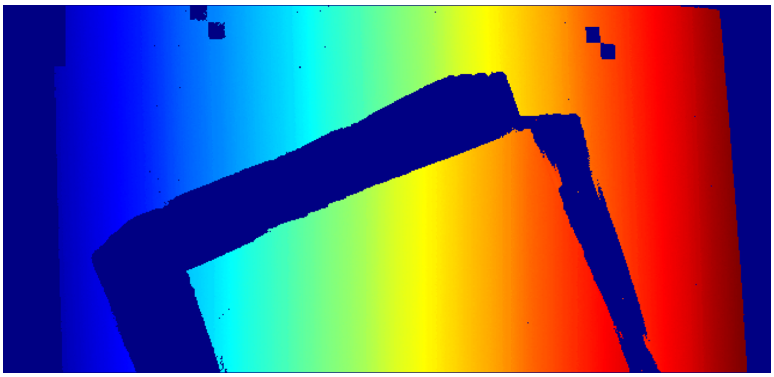
Figure 3.13: (a)-(b) Two successively unwrapped phase images using the method in [91]; (c) regions of phase difference in consecutive images.



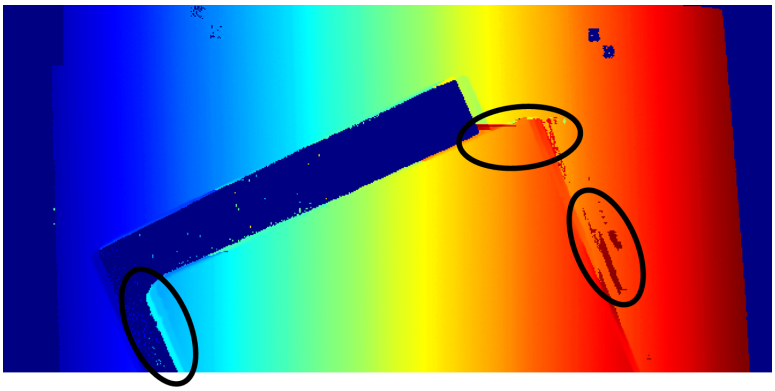
(a) (b)

(c)

Figure 3.14: (a)-(b) Two successively unwrapped phase images with our proposed algorithm of Section 3.4; (c) regions of phase difference in consecutive images.



(a)



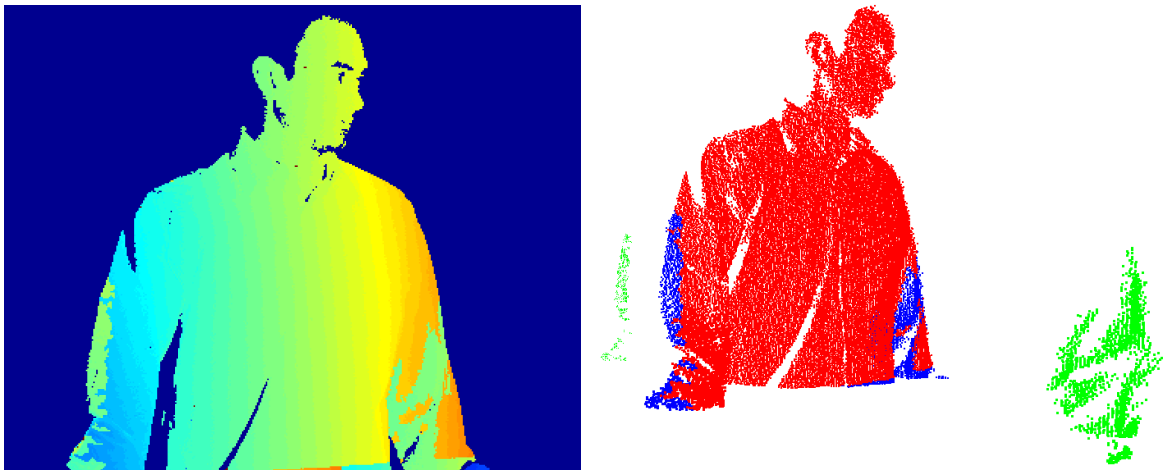
(b)

Figure 3.15: Front board is moving fast in front of back flat surface; unwrapping by (a) our proposed algorithm of Section 3.4; (b) the algorithm in [8].

Both our methods are effective for scenes in which the majority of the points in the scene can be viewed by both cameras. In situations where the two camera views are significantly different, neither of our stereo-assisted methods is able to correctly unwrap many points in the scene.

3.6 Discussion

Even though the main focus of this chapter has been on phase unwrapping, it is informative to examine the effect of unwrapping on the quality of the resulting 3D point cloud. The errors due to incorrect phase unwrapping are readily noticeable in the resulting 3D point clouds. Fig. 3.16a shows an incorrectly unwrapped phase image generated via [91] with errors in the arms and along the bottom of the image. The resulting 3D geometry for this image is shown in Fig. 3.16b. The red points represent the correctly unwrapped points, while the points in green represent the incorrectly unwrapped segments. Applying our viewpoint- consistent method of Section 3.2, we obtain blue points in Fig. 3.16b which are a result of correct unwrapping. As shown in this example, it is



(a) (b)

Figure 3.16: Example of 3D geometry from incorrectly unwrapped phase image generated via [91]: (a) incorrectly unwrapped phase image with errors in the arms and bottom of image; (b) resulting geometry; Points in green are due to incorrect unwrapping of some regions in (a); points in blue result from our viewpoint-consistent algorithm which correctly unwrapped those regions.

common to have unwrapping errors occur over local segments.

One of the fundamental causes of error in the stereo SL method is the false matches in the projected 3D points [91]. When the M points are projected onto the second camera view it is possible for one or more of the non-corresponding points to fall onto an image point that has a similar phase value to the one being searched for. The likelihood of this depends on the scene structure as well as the number of periods in the projection pattern. When the number of periods M in the projected pattern increases, so does the probability of a false match.

In our viewpoint-consistent unwrapping method, it is also possible to incorrectly choose the correspondence by relying on the stereo match data alone. Specifically, possible corresponding 3D locations from the left and right cameras all lie along the ray out of the projector that corresponds to the projector pixel being evaluated. In this setup, the correctness of points is based on the closeness in Euclidean space of the corresponding 3D points in the left and right view rather than closeness in phase. Switching the distance metric from a Euclidean distance to a phase difference does not necessarily guarantee better results. However, by enforcing the 3D and 2D cost functions in the energy minimization, our viewpoint-consistent method is able to generate accurate results.

The viewpoint-consistent method offers an advantage over [91] in that it determines the absolute phase of corresponding pixels in the two camera views, thus ensuring consistency in the final unwrapped phase between the two views. In [91], an estimate of the absolute phase for each pixel is fed into a loopy belief propagation algorithm. Despite using stereo information to determine the absolute phase estimate of all the pixels in the two camera views, there is no attempt to directly enforce the corresponding pixels in the two camera views to the same absolute phase value. For

example, a pixel in the left camera may find that its true corresponding pixel in the right camera has the lowest correspondence cost, and thus the absolute phase is correctly set for the left camera. However, the actual corresponding pixel in the right camera may not find the original pixel in the left to have the lowest cost. Although the loopy belief propagation step in [91] can clear up some of these errors, it does not ensure consistency across the two cameras.

In [91], the stereo matching information is used to select an initial offset for each pixel. It is acknowledged that the stereo information for a single pixel does not always indicate the correct absolute phase for a pixel. However, hard decisions are made with these stereo measurements and the resulting errors are later corrected by segmenting the initial unwrapped phase image and by applying an energy minimization on the mislabeled unwrapped image. In contrast, rather than selecting an individual offset for each pixel, our temporal method selects an offset for a group of pixels that are first locally unwrapped according to an edge quality measure. By restricting local unwrapping to edges with high measures of reliability, the correct relative phase between all pixels in a chain can be determined with high confidence. Once the relative phase of all pixels within the chain is known, only a single offset needs to be chosen in order to find the absolute phase for all pixels within the chain. The final probability distribution for the chain is generated by combining the "noisy" probability distributions of each pixel. In this detection/estimation problem, many random variables are combined to generate a new random variable with a distribution that has a lower likelihood of error.

In [91], the estimated absolute phase is segmented by combining pixels with a phase difference of less than $|\pi|$. Phase unwrapping based on local gradients has been shown to be less accurate than methods using other edge quality measures such as discrete Laplacians [8]. In addition, once segmentation is completed, the local smoothing component of the energy cost function in [91] only operates along the edges of segments. If incorrect labeling occurs within the interior of a segment, the local smoothing cost component cannot detect the error. In contrast, all edges between pixels in our temporal method are examined, and are only connected if the edge is reliable.

The most significant difference between our temporal method and existing phase unwrapping methods for SL is the enforcement of temporal consistency in the unwrapped phase maps [91]. In doing so, we take advantage of the inherent temporal correlation between successively captured phase images. If the capture rate of a system is high enough to ensure that the phase at a single point does not change by more than $|\pi|$ in successive frames[4], then the same spatial local smoothing assumptions during phase unwrapping can also be applied in the temporal domain. In our case, consistent labeling over time is ensured by building larger chains that span multiple frames. The offset for each of these larger chains is determined more accurately as long as only high quality edges are used during local unwrapping.

Finally, our temporal method improves the phase unwrapping results obtained by applying the technique in [8] to a structured-light system. The primary motivation of the phase unwrapping approach in [8] is magnetic resonance imaging applications. Unlike [8], by using the stereo observations from two cameras, our algorithm can determine absolute rather than relative phase values.

[4]For scenes with high speed motion, phase recovery at each pixel is erroneous if not compensated [91].

3.7 Selecting Between Methods for Capturing Dynamic Geometry

Both the viewpoint-consistent and temporally consistent approaches presented in this chapter are able to accurately unwrap phase images. However, when capturing dynamic geometry with our structured-light system, we must choose one approach to process the captured phase data. In applying both approaches to complex dynamic scenes, we find that the temporally consistent method is able to provide correctly unwrapped results more often than the viewpoint-consistent method. The underlying advantage of the temporally consistent method is in the high similarity between successive phase images. The extra information from taking into account the correlation of phase images over time is much greater than simply trying to use redundancy from two separate viewpoints of the scene in the viewpoint-consistent method. The temporally consistent method is able to generate unwrapped phase results that lead to accurate point clouds. Although not explored, if our viewpoint-consistent method was modified to also include temporal information, then the resulting method may offer improved results.

3.8 Conclusions

We have proposed two schemes for consistent phase unwrapping in a stereo SL system. The stereo cameras in our system not only aid in unwrapping phase, but also allow for reconstruction of a class of disjoint scenes not possible in the single camera case. Our viewpoint and temporally consistent phase unwrapping methods are more robust than simple stereo-assisted phase unwrapping alone. Enforcing viewpoint and temporal consistency results in fewer errors in unwrapped images and more accurate point clouds.

In future work, the basic temporal unwrapping idea could be combined with the viewpoint-consistent approach. Rather than running the energy minimization across the pixels of a single projected image, a larger graph could be constructed with multiple projector frames. At least one new data term would have to be added to enforce a smoothing in the labeling that occurs across time. Another improvement is to apply a short, sliding temporal window in order to keep the computational complexity manageable.

Chapter 4

Multi-Camera-Projector Calibration

In this chapter, we address the problem of robust calibration of multiple-camera-projector (MCP) systems. Our primary motivation is a multi-view structured-light system, as shown in Fig. 4.1, which captures the dynamic geometry of a scene from multiple perspectives. In particular, we are interested in camera-projector configurations where the sensors in the system surround a central observation volume. As seen in Fig. 4.1, the cameras and projectors are oriented so that their central axes of observation all roughly point towards the center of a capture volume. Thus, some of the sensors are facing each other rather than pointing in the same general direction. The key to joint calibration of a system with multiple cameras and projectors is to generate a set of correspondences across all devices. To accomplish this, we use a simple calibration target, namely a sheet stretched over a PVC pipe frame, as shown in Fig. 4.2. Binary patterns encoding the vertical and horizontal pixel locations of the projector are projected onto this target to establish camera-projector pixel correspondences across all cameras. The patterns that illuminate the sheet are clearly visible from both sides of the sheet making it possible to establish correspondences between cameras and projectors even when they are positioned in opposite facing directions. The sheet does not significantly scatter the projected light, resulting in the projected pattern appearing in focus from both sides of the calibration sheet. The key to our approach is to decode the projector-to-camera correspondences in order to generate camera-to-camera and projector-to-projector pixel correspondences. The final set of global correspondences consists of the projected coordinates of observed 3D points in each of the devices in the system. This set along with its corresponding set of 3D positions is input to a bundle adjustment framework along with a coarse initial estimate of each device's intrinsic and extrinsic measurements. The output is the refined intrinsic and extrinsic parameters for all the devices in an MCP system. Of existing work in the literature, our proposed approach comes closest to that of [33] except that the translucent sheet enables us to find correspondences among all devices, rather than only a subset of devices with overlapping views.

The outline of this chapter is as follows: Section 4.1 explains the pattern projection and decoding process. Section 4.2 presents strategy for generating correspondences across all sensors in the system. Section 4.3 includes results on calibrating an actual system, and Section 4.4 provides concluding remarks.

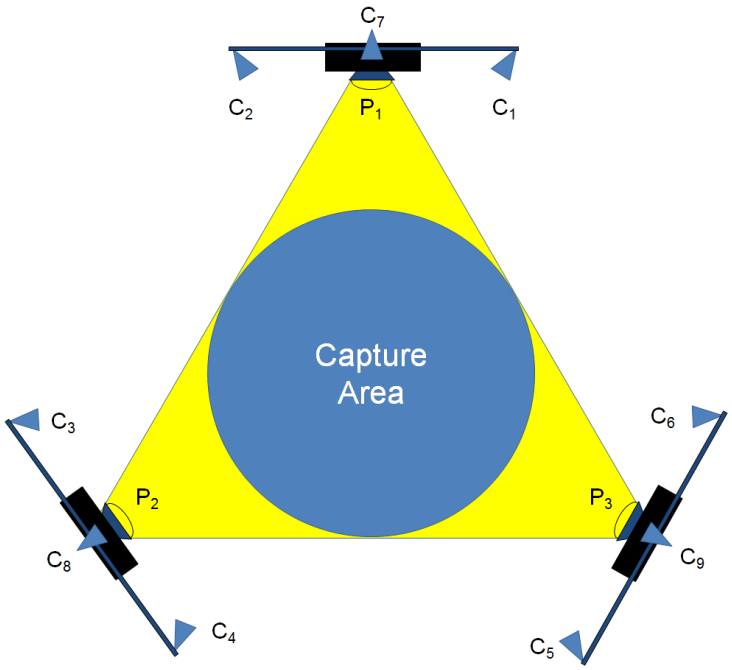


Figure 4.1: Layout of a multi-view structured-light system. Arrows represent the pairs of devices whose extrinsics are computed during initial calibration estimation.

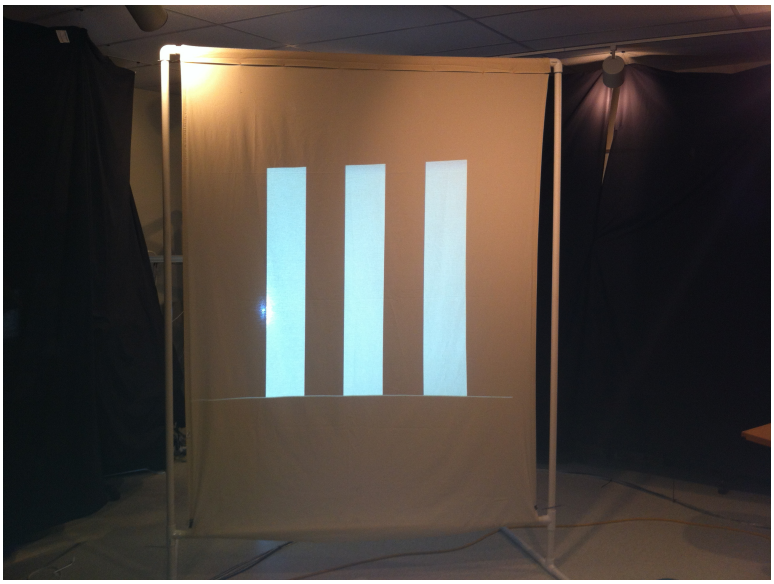


Figure 4.2: Calibration sheet used to increase overlapping view between cameras and projectors. The projected pattern on the sheet appears sharp and in focus from both sides of the sheet.

4.1 Pattern Projection and Capture

To establish global correspondences between all devices, first correspondences between each projector and all cameras are generated. This is done by encoding the horizontal and vertical position of each projector pixel with binary structured-light patterns. The overall calibration process requires multiple correspondence sets to be generated by placing the projection target in multiple positions. In doing so, correspondences are found at 3D locations that are spread throughout the capture volume. At each target position, each projector projects a set of vertical and horizontal binary-coded patterns which are captured by all the cameras. These images are used to establish global correspondences across all devices. During calibration, the target is positioned in the center of the capture area within the limited depth of field of the projectors. This ensures the projected patterns appear in focus and allows for accurate decoding.

The outline of this section is as follows: We begin by describing the binary patterns in Section 4.1.1. The process for decoding the binary patterns is explained in 4.1.2. Finally, the correspondence mappings between the cameras and projectors are presented in Section 4.1.3.

4.1.1 Pattern Projection

For a single camera and projector, pixel correspondences can be determined by projecting binary-coded patterns [75]. Most commonly, the patterns are generated by taking the index value of each column of pixels in the projector and representing it as as a binary number. A binary image is projected for each bit of the binary representation. The result is a set of striped patterns of decreasing width as shown in Fig. 4.3. Cameras observing a scene illuminated by the binary patterns can determine the projector column illuminating the scene point observed by each pixel by decoding the set of binary patterns. This creates a correspondence between each pixel in the camera and the columns of the projector. If a set of vertical and horizontal binary-coded patterns are projected, then pixel to pixel correspondences between the camera and projector are established. This process is repeated for all cameras observing the binary projected patterns. [1]

We opt to use a translucent, planar sheet as the projection target to calibrate all sensors. Translucency is needed so that even though the target is being projected from one side, the cameras viewing it from the opposite side can still view and decode the projected patterns. The projected patterns appear equally focused on both sides of the sheet, although with slightly lower brightness when viewed on the side not directly illuminated. Furthermore, to ensure proper decoding, the sheet must approximate a planar surface. As such, we have opted to frame it with PVC pipes as shown in Fig. 4.2. Using a translucent sheet as a projection target greatly increases the positions from which cameras can observe the projected patterns. In spite of this, cameras whose central axis are nearly perpendicular to the normal of the calibration sheet are unable to decode the projected patterns. To resolve this, we capture image sets with the target in different positions so as to obtain correspondences between most devices. Specifically, for the MCP system in Fig. 4.1, we place the sheet at the

[1] An alternative to direct binary coding for the binary patterns is Gray codes. We have empirically found direct encoding of the column indices to be sufficient since we only keep the decoded positions for pixels that are confidently decoded.

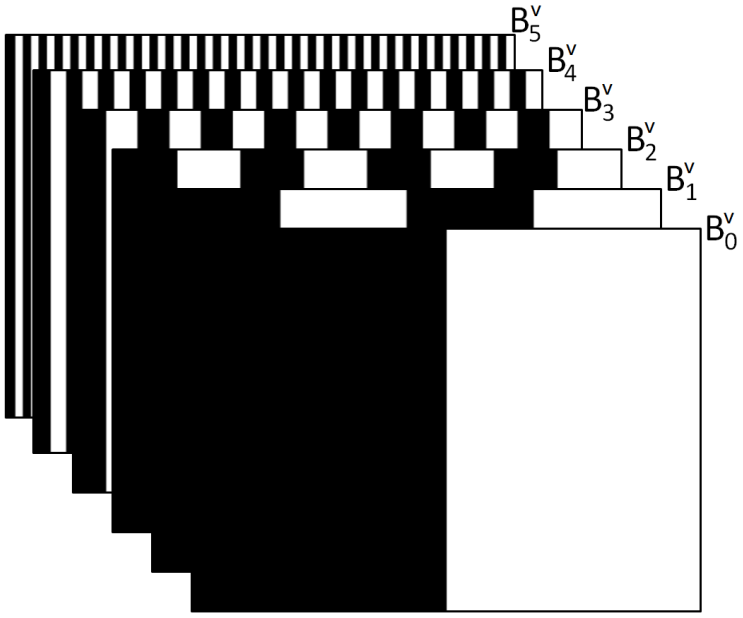


Figure 4.3: Vertical binary-coded projection patterns.

center of the capture area and orient it in multiple directions near the center of the capture area for each pattern projection and camera capture. In doing so, we are able to determine correspondences between a projector and all of the surrounding cameras. We use two sets of patterns to determine full pixel to pixel correspondences: a vertical set of binary codes, $B_n^V, n \in \{0, ..., K_V - 1\}$, and a horizontal one, $B_n^H, n \in \{0, ..., K_H - 1\}$, where K_V and K_H are related to the resolution of each projector, i.e. $K_V = \lceil \log_2(\text{projector width}) \rceil$ and $K_H = \lceil \log_2(\text{projector height}) \rceil$. In addition to the frames that represent the binary value, the inverse of each frame is also projected. As explained later, capturing a binary frame and its inverse makes the decoding process more robust and obviates the need to choose a global threshold to decode each frame.

4.1.2 Decoding

Once both sets of binary patterns are projected and captured by all the cameras, the images are decoded for a given target position. To decode each bit in the projected binary code, the image for each bit and its inverse representation are both used. We first compute the absolute difference between the two frames; the larger this difference the more reliable a binary labeling of 0 or 1. A small absolute difference occurs when camera pixels are not directly illuminated by the projected patterns or when the transition between black and white stripes in the projected binary images falls on a camera pixel. Only bits where the absolute difference is greater than an assigned minimum threshold are considered to be properly decoded. For such pixels, the bit is decoded as a 1 if the intensity from a frame is greater than its inverse, and as 0 when the inverse frame has a greater

intensity. [2]

After decoding all bits, we only keep the correspondences for pixels where all K_V and K_H bits of the vertical and horizontal patterns respectively are properly decoded. For cases where the projector resolution is higher than that of the camera, camera pixels can still be kept as correspondences even if the least significant bits are not decoded.[3]

4.1.3 Mapping

Assume a system consists of the set of cameras C, projectors P, and target plane positions T. For each camera $c \in C$ at a given plane position $t \in T$, correspondence matrices $Q^t_{x:c,p}$ and $Q^t_{y:c,p}$ are defined between the pixels of the camera and the x and y coordinates of each projector $p \in P$. The dimensions of $Q^t_{x:c,p}$ and $Q^t_{y:c,p}$ are both equal to the resolution of the camera c. Each element in the correspondence matrix holds the coordinates, either x or y, of its corresponding pixel in projector p. These correspondences are generated from the decoded binary patterns as described in Section 4.1.2. Jointly, $Q^t_{x:c,p}$ and $Q^t_{y:c,p}$ are referred to as $Q^t_{c,p}$. In the remainder of this chapter, we will drop the x and y notation for each correspondence matrix and merely refer to the union set. The correspondence matrices from the projector to the cameras are defined as $\bar{Q}^t_{p,c}$ [4] with the dimensions equal to the resolution of projector p. Each entry in these matrices holds the coordinates to the corresponding pixel in the camera c. We use $Q^t_{c,p}$ to populate correspondence matrix $\bar{Q}^t_{p,c}$. Specifically for each camera pixel in $Q^t_{c,p}$, its corresponding projector pixel in $\bar{Q}^t_{p,c}$ is populated with the location of the original camera pixel. Fig. 4.4 shows an example of $Q^t_{c,p}$ graphed as an image. The color in this picture represents the value of the corresponding coordinate in the projector. The figure illustrates how densely correspondences can be determined between devices. Also, the calibration target is large enough to allow correspondences to be established throughout most of the camera's field of view. The target is placed at multiple positions to ensure not only correspondences across all devices, but also correspondences for all projector pixels. Mask matrices $M^t_{c,p}, \bar{M}^t_{p,c}$ are generated to indicate which pixels in $Q^t_{c,p}$ and $\bar{Q}^t_{p,c}$ have a correspondence. The sets of correspondence matrices and masks are generated for all $|T| \times |C| \times |P| \times 2$ combinations of $t \in T$, $c \in C$, and $p \in P$.

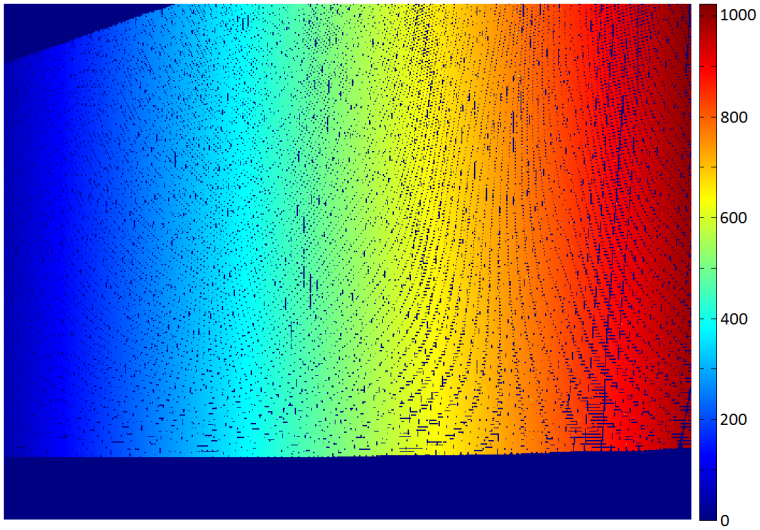
4.2 Full Correspondence Generation

So far, the sets of camera-projector correspondences for each target orientation have been determined. The goal is to use the individual pairwise correspondence sets to find global correspondences across all, or nearly all, devices. For each projector, the projector pixels with correspon-

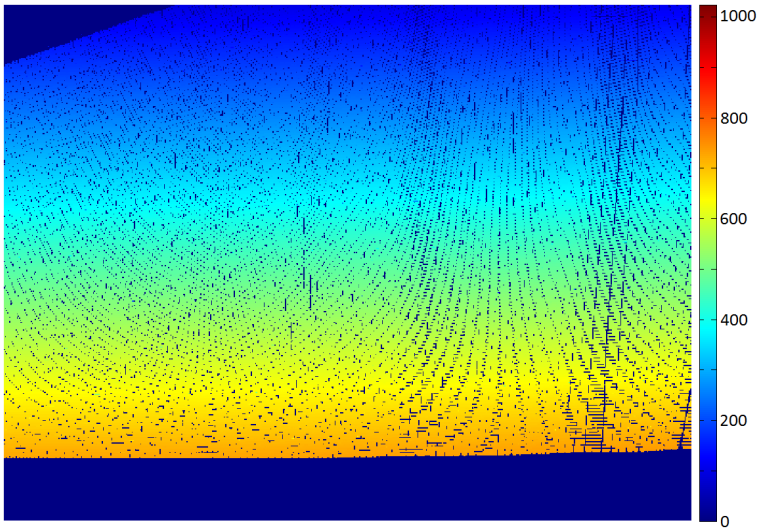
[2] In our camera-projector configuration, the light sources of the projectors are not visible in the captured camera images. For the configurations where this does not hold, the lamp needs to be masked out and camera gain needs to be adjusted accordingly.

[3] The projector-camera resolution mismatch for generating correspondences makes the decoding of the least significant bits difficult [84]. It is possible to use additional sparse high frequency patterns to obtain more accurate correspondences.

[4] Throughout this chapter, we use the bar for symbols related to the projector.



(a)



(b)

Figure 4.4: Mapping of correspondences from camera c to (a) x-coordinates $Q^t_{x:c,p}$ and (b) y-coordinates $Q^t_{y:c,p}$ of projector p.

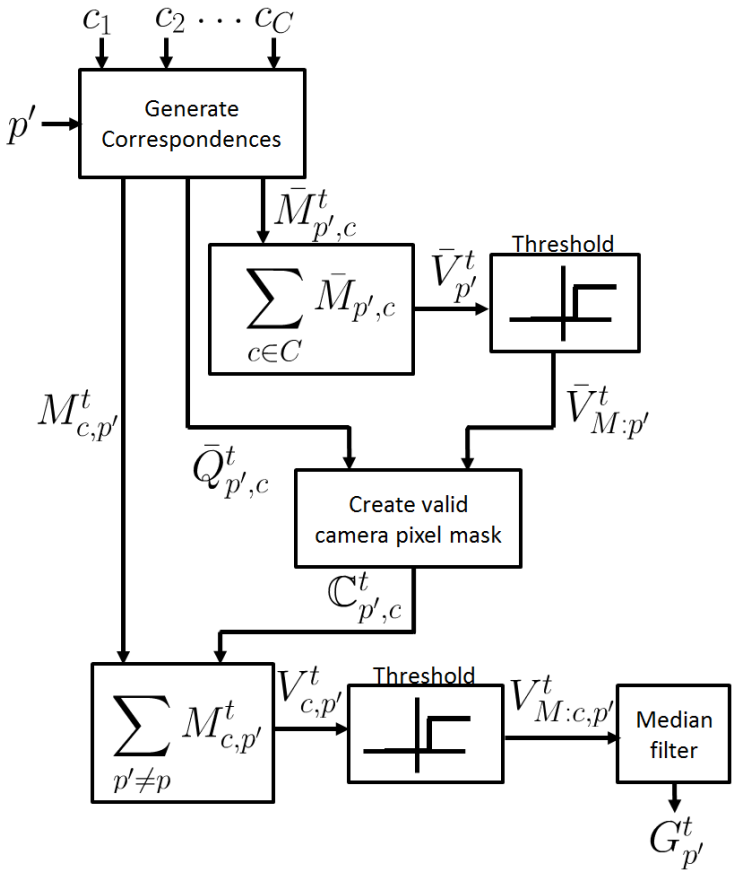


Figure 4.5: Block diagram of the processing steps for the first projector p'.

dences in multiple cameras are used to identify a set of potential global correspondences. For each of the projector pixels in this set, the corresponding pixels in each of the cameras are identified. These camera pixels together with the camera-to-projector correspondence matrices are used to find potential correspondences to the other projectors in the system. A detailed explanation of the above process follows.

To start, we use the set of pairwise correspondences and masks from a single target position t, $Q^t_{c,p}, M^t_{c,p}, \bar{Q}^t_{p,c}, \bar{M}^t_{p,c}, \forall c \in C, p \in P$ to generate a set of global correspondences $G^t = \{G^t_x, G^t_y\}$. Global correspondences are those seen across many devices in the system. The matrix G^t_x has $|P| + |C|$ rows; its number of columns is equal to the number of global correspondences found for target position t. The global correspondence sets for each target position are concatenated to create a final set of global correspondences $G = \bigcup_{t \in T} G^t$. The matrix G has $|P|+|C|$ rows and the number of columns is equal to the sum total of correspondence points found in all target positions.

Fig. 4.5 shows the block diagram of the steps to generate G^t for the first projector p'. The processing for the remaining projectors in the system is more or less identical to the first one except for a minor detail to be described shortly.

4.2.1 Correspondences Using Visibility Matrices

We start by examining the projector-to-camera correspondences for the projector p', denoted by $\bar{Q}^t_{p',c}$, $c \in C$. The set of correspondence masks, $\bar{M}^t_{p',c}, c \in C$, are element-wise summed together, $\sum_{c \in C} \bar{M}^t_{p',c}$ to obtain the projector visibility matrix, denoted by $\bar{V}^t_{p'}$, for projector p'. The dimensions of $\bar{V}^t_{p'}$ are equal to the resolution of projector p'. The values in the projector visibility matrix represents the number of cameras that are able to view each pixel of the examined projector p'. Only projector pixels that are visible to enough cameras, i.e. larger than C_{min}, are considered as candidate global correspondences. These pixels are used to generate the visible projector pixel mask for projector p' and target t, denoted by $\bar{V}^t_{M:p'}$. Intuitively, the projector pixels in $\bar{V}^t_{M:p'}$ have correspondences in a sufficiently large number of cameras.

The visible projector pixel mask $\bar{V}^t_{M:p'}$ along with the projector-to-camera correspondences $\bar{Q}^t_{p',c}$ are used to identify the pixels in each camera, $\mathbb{C}^t_{p',c}$, that correspond to the pixels in $\bar{V}^t_{M:p'}$. Intuitively, the camera pixels in $\mathbb{C}^t_{p',c}$ identify the correspondences to those projector p' pixels that are "visible" in a sufficiently large number of cameras.Next, we determine the number of projectors that can see each camera pixel in $\mathbb{C}^t_{p',c}$ denoted by matrix $V^t_{c,p'}$. The matrix $V^t_{c,p'}$ is defined by setting the $(i,j)^{th}$ value to $\sum_{p'' \neq p'} M^t_{c,p''}(i,j)$ for positions where $\mathbb{C}^t_{p',c}(i,j)$ is non-zero. Next, we determine the subset of camera pixels in $V^t_{c,p'}$ that are visible in a sufficiently large number of projectors, i.e. in more than P_{min} projectors. We denote this set by matrix $V^t_{M:c,p'}$ and compute it for every camera. Intuitively the pixels in $V^t_{M:c,p'}$ represent pixels in camera c that are sufficiently visible and can therefore be used to calibrate the projector extrinsically with respect to the other devices.

In most cases, if a correspondence between projectors exists, multiple cameras are able to observe it. Sometimes small errors in the camera-to-projector pairwise correspondences can occur due to the mismatch in resolution of cameras and projectors. The camera redundancy in these projector-to-projector correspondences can be used to remove decoding errors. To do so, we stack the set of redundant observations from multiple cameras to find correspondences between projector p' and p''. For each set of redundant observations corresponding to a single projector-to-projector correspondence, the median x and y coordinates are used as the actual correspondence location and stored in matrix $G^t_{p'}$ along with correspondence coordinates for the cameras.

4.2.2 Global Correspondence Matrix

We now proceed to find correspondences $G^t_{p''}$ for the second projector p''. A matrix $V^t_{M:p''}$ for a second projector p'' is generated in the same way as $V^t_{M:p'}$ for the first projector p' and its coordinates are compared against those in $G^t_{p'}$ from the first projector p'. Duplicate points are removed and the processing is continued using the same method as for p'. We remove the redundant points so as to ensure that each global correspondence equally contributes to the final calibration solution. Note that generating G^t_p for each individual projector is only necessary when P_{min} is less than the number of projectors. Otherwise, if it is equal to the number of projectors, it is sufficient to only perform the processing on a single projector. In practice, the value chosen for P_{min} is dependent on the layout of projectors in the MCP system. If all projectors illuminate a common area, then P_{min} should be set to the number of projectors in the system. If the projectors do not all

share a common observation region, then P_{min} should be set to the smallest number of projectors illuminating any one portion of the capture region.

Next, the global correspondence points generated from each projector are concatenated into a single matrix to generate the total set of correspondences for the plane position t, $G^t = \{G_1^t, G_2^t, \ldots, G_N^t\}$. The process for plane t is repeated for all the other plane positions, and a final set of correspondences is generated by concatenating the correspondence matrices from each target, $G = \{G^1, G^2, \ldots, G^T\}$. Matrix G represents the correspondence matrix between every sensor in the system. Before this data is applied to a bundle adjustment (BA) framework, we need to provide two other pieces of information as initial conditions to the BA: 1) a coarse estimate of the calibration parameters and 2) an estimate of the 3D positions of the correspondence points.

4.2.3 Bundle Adjustment

The coarse intrinsic calibration estimate of each camera is generated via Bouguet's camera calibration toolbox [18]. Additionally, we use the stereo calibration feature of the toolbox to estimate the extrinsic relationships between neighboring pairs of cameras. We refer to the process of solving for this extrinsic relationship as "pairwise" calibration. Enough extrinsic pairwise estimates are generated to convert each camera to a defined world coordinate system, which is conveniently chosen to be the same as the coordinate system of one of the cameras. To generate the coarse intrinsic and extrinsic calibration for the projector, we use the Projector-Camera Calibration Toolbox [29] which is built on top of Bouguet's toolbox [18].

The 3D points provided to the bundle adjustment process are generated by triangulating the existing corresponding points from a pair of cameras. If all correspondences are not visible from a single camera pair, the remaining points are triangulated with another camera pair and then transformed into the world coordinate system. The cameras used to triangulate the 3D points should have a similar field of view. It is convenient to define the world coordinate system with respect to one of the cameras in the pair. The estimates for 3D position, sensor intrinsics, sensor extrinsics, and the correspondence information are input into a bundle adjustment framework [67] to refine sensor intrinsics and extrinsics. The output of the BA provides us the intrinsics of each device, including radial distortion parameters, as well as the extrinsic parameters.

4.3 Results

The proposed calibration method is demonstrated on a multi-view structured-light system consisting of three stations surrounding a central capture area. Each station is equipped with two grayscale cameras, i.e Point Grey Dragonfly Express 640×480, a color camera, i.e. Point Grey Flea 2 640×480, and a video projector, i.e. Optoma TX780 1024×768, as illustrated in Fig. 4.6. Each camera and projector is pairwise calibrated with respect to a single camera in each station to be used as initial conditions in BA. In Fig. 4.1, the pairwise calibrations within each station are shown as dashed arrows, and those between stations are shown as solid arrows. In total, nine cameras and three projectors are calibrated. The calibration plane is positioned in eight distinct orientations at the center of the capture area in order to generate the global correspondence points,

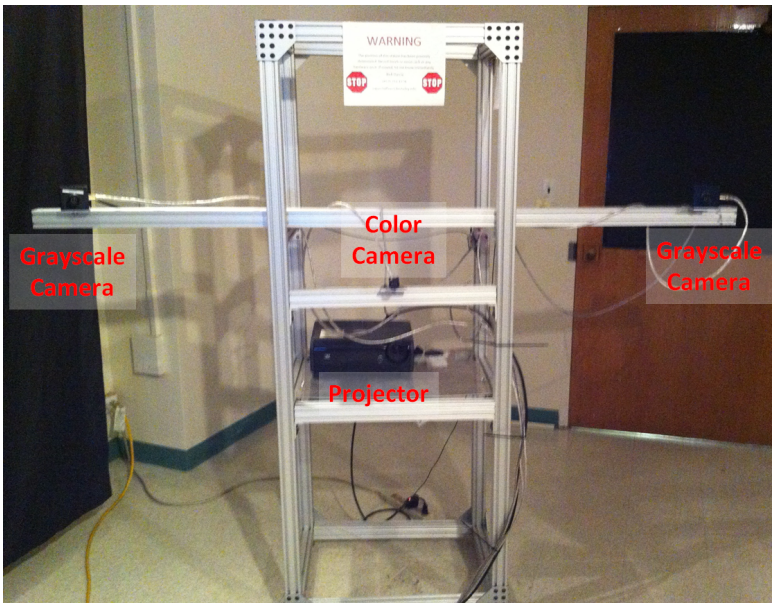


Figure 4.6: Single station from our multi-view structured-light system.

as described in Section 4.2, to be used in bundle adjustment. C_{min} and P_{min} are set to 8 and 3, respectively. All three projectors are able to clearly see the projection target positioned in the central capture area, so P_{min} is set to three. We opted to set C_{min} to 8 rather than 9 to accommodate the cases where a camera may not be able to decode the projected pixels due to the large angle between the central axis of the camera and the normal of the calibration target. In total, 17,262 points are used for calibration. Fig. 4.7 compares the reprojection error of our proposed calibration method against pairwise calibration. As seen, the reprojection error of sensors is considerably smaller for our method than pairwise calibration. Fig. 4.8a illustrates the projection of 3D correspondence points onto the coordinate frame of camera 7. The misalignment between the projected points and the actual point locations are especially visible on the left hand side of the image. Fig. 4.8b shows an improved alignment with the new calibration method.

MCP calibration greatly affects the quality of 3D reconstruction in the multi-view structured-light system of Fig. 4.1. We compare the accuracy of pairwise calibration with our proposed method on a spherical test object of radius 31 cm by applying phase shifted sinusoidal patterns for 3D reconstruction. Partial reconstruction from each of the six grayscale cameras C_1 through C_6 in Fig. 4.1 are merged and a least square fit to the spherical point cloud is obtained. Average distance and standard deviation of the points in the point cloud to the sphere are then used as two accuracy metrics to compare the two methods. The average distance of the proposed method and pairwise calibration are 2.7 mm and 11.0 mm respectively. The standard deviation of the proposed method and pairwise calibration are 2.0 mm and 5.9 mm respectively. The results indicate a closer fit to the sphere and hence superior accuracy of the proposed calibration method. Both calibration methods generate an estimate of the sphere's radius that is within the margin of error, i.e. 1.6 mm, of the manually measured radius. Fig. 4.9 shows slices of the reconstructed sphere along x,y, and z

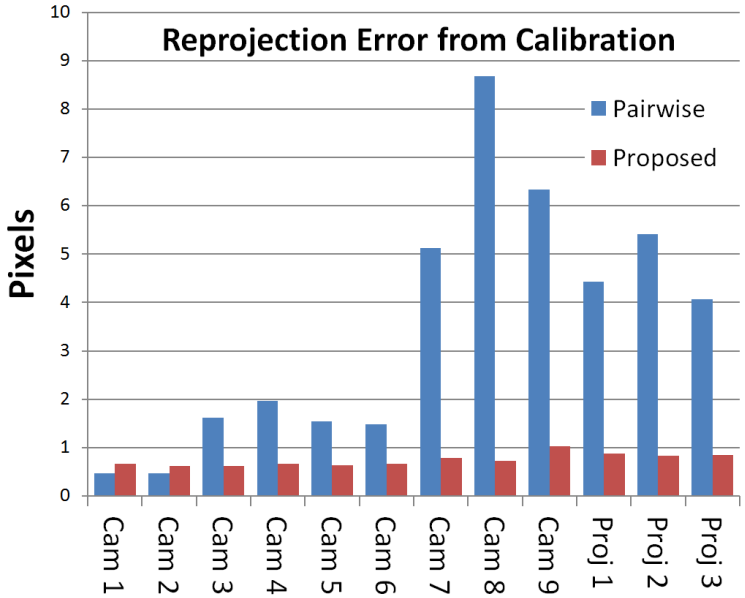
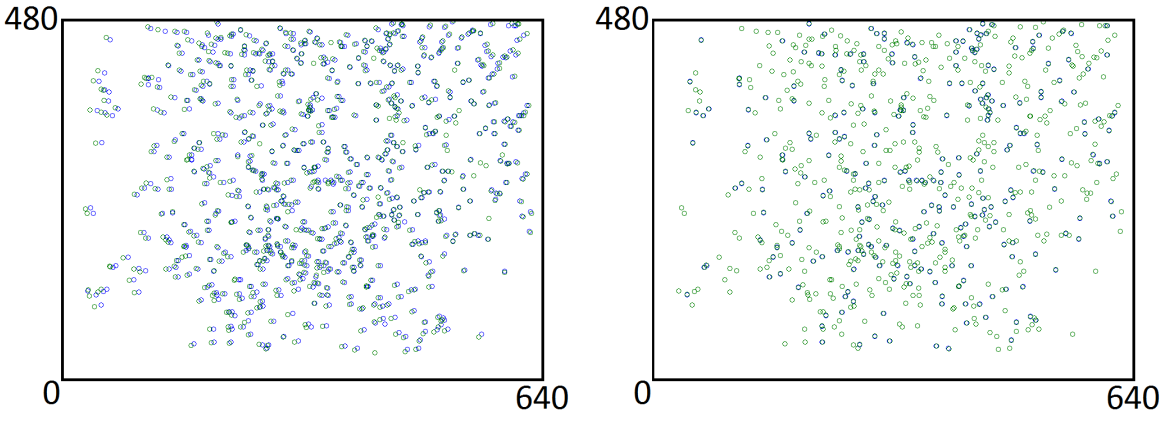


Figure 4.7: Comparison of the reprojection error using the proposed calibration method vs. pairwise calibration.



(a) (b)

Figure 4.8: Reprojection error using (a) pairwise calibration; (b) proposed method. Image locations are shown in green and the projected image locations in blue.

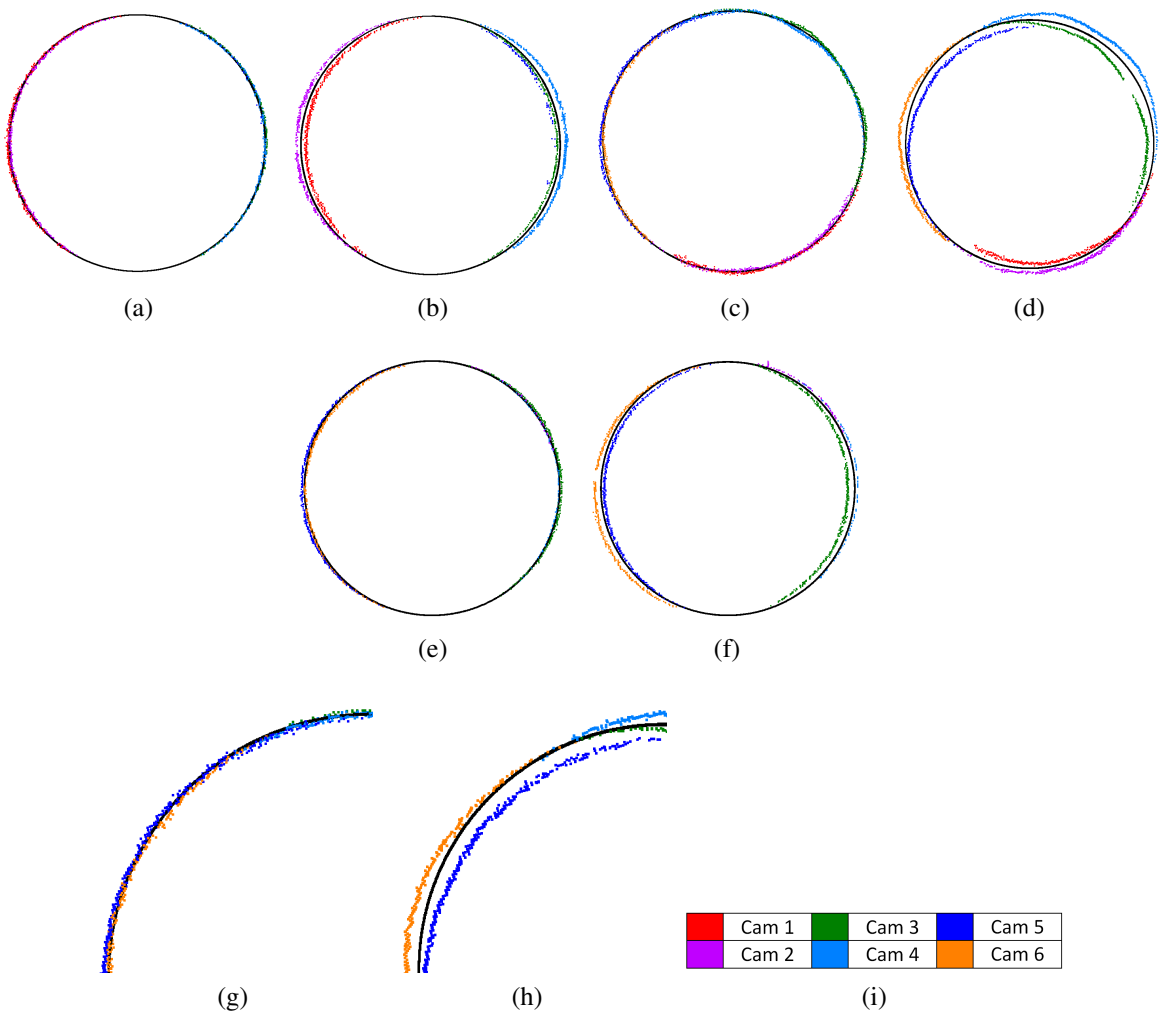
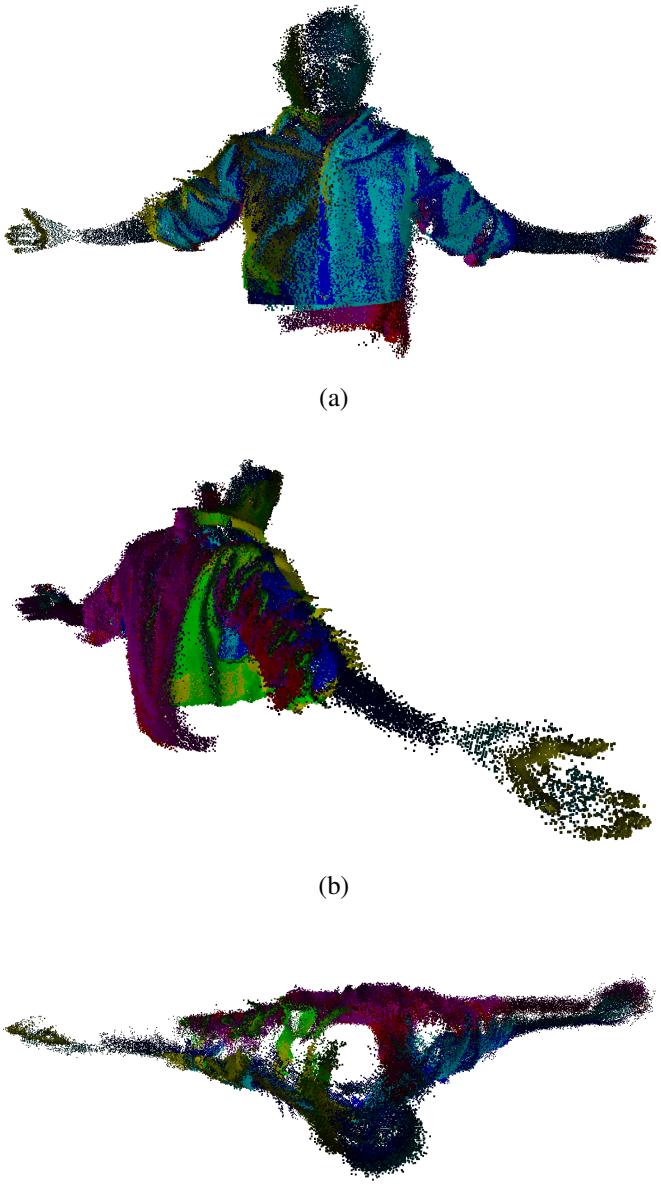


Figure 4.9: Projection of sphere for the proposed method along (a) x-axis, (c) y-axis, (e) z-axis, and for pairwise calibration for (b) x-axis, (d) y-axis, (f) z-axis; (g) zoomed in portion of (a); (h) zoomed in portion of (b). The points for each camera are assigned a unique color (i).

axes for both calibration methods. The double surfacing is clearly visible in Figs. 4.9b, 4.9d, 4.9f and 4.9h.

Fig. 4.10 shows three views of merged point clouds for a moving person generated by the sinusoidal phase shifted, three-view structured-light system shown in Fig. 4.1, calibrated with our proposed method. The resulting point cloud from each camera is shown in a different color. As seen, the resulting point clouds from all six cameras are well aligned.



(a)

(b)

(c)

Figure 4.10: Point cloud of moving person generated from a sinusoidal phase shifting three-view structured-light system of Fig. 4.1 calibrated with our proposed method: (a) front, (b) back, and (c) top view; the partial point clouds for each of the six cameras in three stations are shown in 6 colors: red, green, blue, cyan, magenta, yellow.

4.4 Conclusions

We have proposed a method for calibrating an MCP system by generating correspondences across all sensors in the system. Aside from generating correspondences, our calibration target opens up the possibility of generating correspondences between projectors. The method is effective even in environments where the viewpoints of all sensors surround a central area and do not necessarily share a field of view.

Chapter 5

Dynamic Deforming Geometry

In this chapter, we present a markerless motion capture system consisting of multiple structured-light subsystems to capture geometry of a human subject from surrounding views. As with any multi-view system, since the entire shape of the human subject is not always visible from any one capture device, we opt to use a template to represent the shape and pose of the human subject. However, unlike existing approaches which require additional scanning hardware or complex template initialization processes, we use the same motion capture system to generate the template. Specifically, from an initial scan of the human subject in a occlusion free pose, we create a customized template for that subject as shown in Fig. 5.1b. In contrast to existing methods that use 2D images, we capture 3D geometry using three structured-light systems. Specifically, with two cameras per structured-light system as shown in Fig. 5.2, we generate 3D scans from six views as shown in Fig. 5.1a, deform the template, shown in Fig. 5.1b, and fit the target scan as shown in Fig. 5.1c. The problem of fitting a template to the human subject is greatly simplified if constrained by 360-degree geometry from three stations.

There are existing methods for reconstructing dynamic geometry from 3D geometry scans. Dou et al.[28] use eight Kinects surrounding a human subject to build a model of a dynamic figure. Due to sensor noise, they cannot directly create a template in a single frame; rather, they learn it over time. Our system is most similar to [46] which captures two opposing views of the human subject and stitches the views together, leaving significant artifacts around the seams.

The outline of this chapter is as follows. In Section 5.1, we present the specifics of the system architecture and hardware used to capture 3D partial reconstructions of the human subject. Section 5.2 describes how we generate and deform the template. In Section 5.3, we explain the sequential deformation method used to fit a template to partial geometry scans. Section 5.4 describes texture mapping for the template. Sections 5.5 and 5.6 present an example of processing real captured human motion and conclusions, respectively.

(a) (b) (c)

Figure 5.1: Using (a) captured partial scans and (b) a template of a human subject, we generate (c) the time varying geometry of a human subject.

5.1 System Setup and Data Capture

Our proposed system consists of three distinct structured-light stations, shown in Fig. 5.2, surrounding a central capture volume with the devices located on each station pointing towards the center of the capture volume. Each station is equipped with one DLP® projector, two grayscale cameras, and one color camera in a configuration similar to [91]. To prevent interference between the three projectors at each structured-light station, we apply structured-light patterns to one projector at a time in a round-robin fashion. In other words, each station takes turns illuminating the human subject in a time-multiplexed sequence. The system is controlled by two separate desktop PCs, one for driving structured-light patterns to the three projectors, and a second one for streaming in images from all nine cameras. A microcontroller is used to extract timing signals from the projectors to generate trigger signals for all of the cameras. An external PCIe expansion chassis is added to the camera PC to provide enough Firewire cards to connect all nine cameras to the single PC.

The resolution of the projectors is 1024×768, and the resolution of the cameras is 640×480. At each station, the two grayscale cameras are used with the local projector to capture geometry, and the color camera is used to capture texture. As described in Chapter 2 and [96], the projectors are modified to only project in grayscale by removing the color wheels. DLP® projectors generate color images by sequentially projecting the individual red, green, and blue color channels of a video source. The removal of the color wheel effectively modifies the projector from a 60 Hz color projector to a 180 Hz grayscale projector. Similar to [96], we use three phase-shifted sinusoidal patterns to reconstruct depth. Each pattern is stored into one of the three separate color channels of an image and sent to the projector. The three sinusoidal patterns are individually captured as they illuminate the human subject and the phase of each pixel in the camera is computed [96]. While the phase of a camera pixel determines which points in the projector potentially illuminate it, due to the periodic nature of the sinusoidal patterns, correspondences between the cameras and

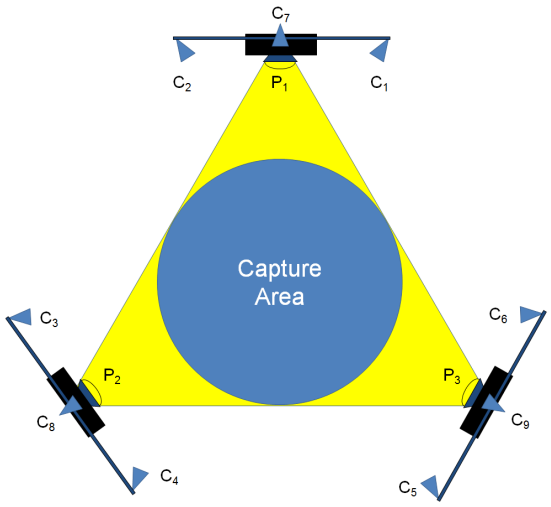


Figure 5.2: Top-down view of the three structured-light stations in the system. Geometry cameras are labeled $C_1 - C_6$. Color cameras are $C_7 - C_9$. Projectors are $P_1 - P_3$.

projectors are not uniquely known by the phase alone. By using the temporal phase unwrapping method from Section 3.4, the phase images can be unwrapped over time and space to ensure spatiotemporally consistent point clouds at each station. Specifically, we combine a quality guided phase unwrapping approach with absolute phase estimates from the stereo cameras to solve for the absolute phase of connected regions. Temporal unwrapping with stereo information significantly improves the consistency of unwrapped results, which in turn leads to consistency of the point clouds [36].

If multiple projectors simultaneously illuminate the same point on the human subject, neither the phases of projectors can be accurately estimated, nor can the correspondence between camera and projector pixels. Since the projectors natively operate at a 60 Hz rate, the time-multiplexing between the three stations lowers the capture rate from each station to 20 frames per second. Clearly faster projectors allow for capturing human subjects with faster motion. To implement this interleaving process, the video signals sent to each projector are all synchronized to each other. Specifically, we use two NVIDIA® Quadro® 5000 graphics cards with an additional G-Sync® daughter card in the display PC to synchronize all video source signals. Triggering signals sent to the capture cameras are derived from the timing of the synchronized projectors. All timing signals are generated using an external microcontroller. Similar to [96], the grayscale cameras capture the individual structured-light patterns at approximately 180 Hz, and the color cameras capture at 60 Hz. The exposure of the color cameras is chosen to last for the entire duration of projection of the three sinusoidal patterns; thus the integration of the three phase-shifted sinusoidal patterns appears as a constant illumination to the color cameras.

Each grayscale camera projector pair is capable of capturing a portion of the human subject's geometry from its own viewpoint. Since our system is designed to capture the full 360-degree view of the human subject, the individual views need to be aligned into a a single coordinate frame, thus

requiring an accurate calibration of all cameras and projectors. We use the multi-camera-projector calibration approach in Chapter 4 and [37] to accomplish this. We use a translucent target placed at various positions within the center of the capture volume as a calibration target. Each projector projects striped binary patterns in the vertical and horizontal directions. The observing cameras in the system can decode the striped binary patterns to determine pixel-level correspondences between all devices. The correspondences along with estimates of the 3D locations of the correspondence points are fed into a bundle adjustment optimization method to calibrate all the devices.

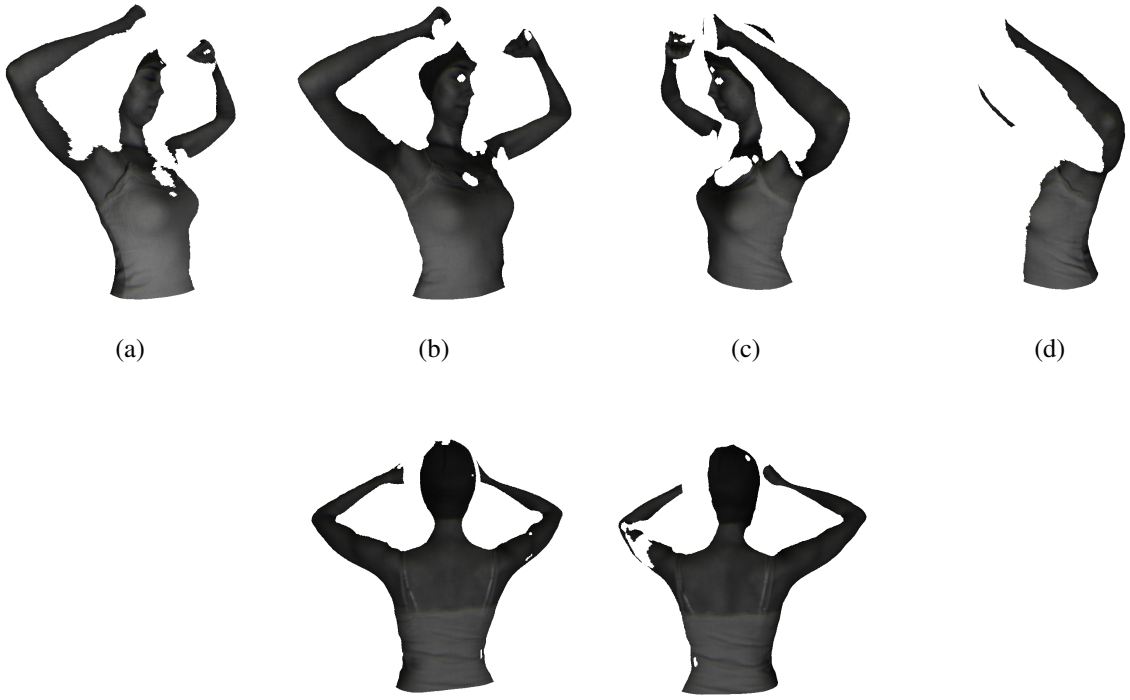
Even though the evolving pose of the human subject can be observed frame by frame after point cloud alignment, the limited number of capture devices and stations surrounding the human subject causes holes during reconstruction. This is especially true for regions that are not illuminated by the projector, such as tops of heads and regions that are occluded by other parts of the body, such as a portion of a torso occluded by an arm. Additionally, regions with low reflectivity and regions with shadows also create holes. Fig. 5.3 illustrates a single frame captured by each of the six grayscale cameras of the system in Fig. 5.2. There are several holes in each of the views which need to be filled out in the fused 3D model. This motivates the use of templates in our system.

5.2 Template

Even though visual hulls can be used to estimate geometry for many poses of the human subject, for scenes with significant self-occlusions, it is not always possible to generate a watertight representation of the entire subject's surface. As such, *a priori* information on the structure of the human subject can help to properly fill in these regions. In this chapter, we use a template to create a complete representation of each frame. However, unlike existing methods [9], we use the same system configuration that captures the dynamic geometry to also capture the template, obviating the need for a separate system such as a high-quality laser scanner.

To create a template, the human subject is captured in a pose where the majority of the his or her surface can be observed by the cameras and projectors so as to limit the self-occlusions. Fig. 5.4 shows the captured geometry with the human subject in the template pose. As seen, the majority of the geometry is captured, except for regions on the tops and bottoms of the arms, the top of the head, and the bottom of the torso, which are not observable by any of the capture devices. These regions are not illuminated by the projector, and therefore their 3D shape cannot be captured. We use Poisson surface reconstruction [58] to fit a smooth meshed point cloud over the existing geometry. As seen in Fig. 5.5, this generates a hole-free representative template of the human subject.

Even though the template provides an *a priori* reference of the true shape of the human subject, as a mesh alone, it lacks the structure to effectively fit the fused captured geometry from the three stations. In poses with significant occlusions, a reference to the overall structure of the template is useful to prevent unnatural mesh deformations. This is especially true when little geometry for the human subject's arms is captured as they pass in front of the body. In these poses, often only the outside of the arm is visible to the capturing devices. The lack of geometry may cause



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

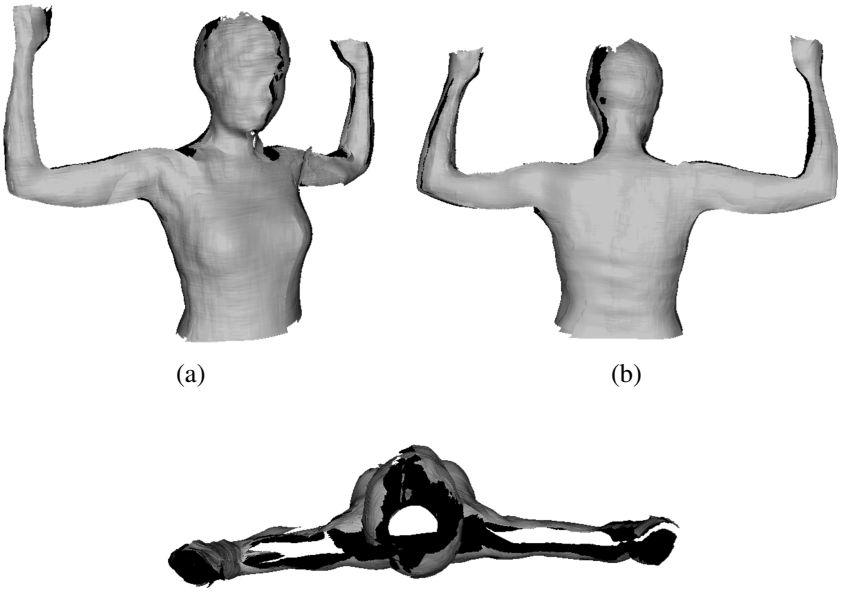
(e) (f)

Figure 5.3: Partial meshed point clouds of a human subject generated from unwrapped phase images of a single frame. Views captured from each of the six grayscale cameras in the system: (a) C_1, (b) C_2, (c) C_3, (d) C_4, (e) C_5, and (f) C_6 as shown in Fig 5.2.

bending in rigid regions of the arm since there are limited correspondence points to enforce a natural deformation. As is done in traditional motion capture, we choose to fit a skeleton rig to the template to control the way its geometry is deformed over time [70].

The skeleton rigging, or skeleton, is a set of artificial bones set within the template to emulate natural pose deformations of the template mesh. The position and movement of the bones control the movement of the vertices in the template mesh. For our system, which only captures a waist-up view of the human subject, we opt to use 12 bones within our skeleton, as shown in Fig. 5.5. [1] Each bone in the skeleton has a single lead joint that controls its movement and position. The joint-bone pairs are labeled in Fig. 5.5. The position of each vertex in the template mesh is determined by the positions and orientations of nearby bones. The influence of each bone on each vertex's position is specified by the bone weight map matrix $\mathbf{W}$ with dimensions of $[N \times J]$, where N is the number of vertices in the template mesh and J is the number of bones in the template. Each entry in the matrix $\mathbf{W}(i, j)$ takes on a real value from zero to one, specifying the influence of each bone j on

[1] The limited field of view of each of the cameras and projectors prevents us from capturing the entire height of the human subject.



(a) (b)

(c)

Figure 5.4: Partial geometry used to generate a template. Views from (a) front, (b) back, and (c) top.

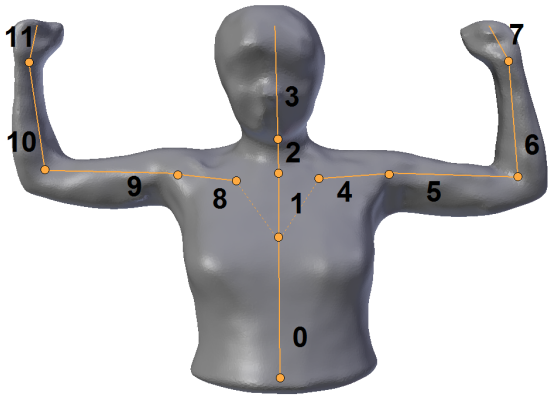


Figure 5.5: Rigged template of a human subject generated by Poisson surface reconstruction. The bones and joints are numbered and indicated in orange.

the position of each vertex i by taking into account a vertex's relative distance to each bone. The vertices are influenced by one to three bones. Each row of influence from all bones to a single vertex is normalized to sum to one.

We deform the template by using dual quaternion skinning [56] to modify the skeleton pose so as to fit the fused geometry from three stations. In contrast to the commonly used linear blend skinning, dual quaternion skinning provides more natural deformation, especially under large twisting motions. We solve for the bone positions that deform our mesh to best align the template to the fused surfaces captured in each frame.

As is standard with skeleton-rigged deformation, rotating a branch around a joint not only rotates the connected bone, but also all subsequently connected bones. Specifically, the skeleton of our template is connected in a tree structure with joints serving as the nodes of the tree and the bones as the paths between nodes. We can think of the set of bones that follow from a joint in the skeleton to a terminal bone in the tree as a branch. The largest branch starts from the joint for bone 0 in Fig. 5.5 and includes all bones in the skeleton. The smallest branches are single bones 3, 7, and 11 as shown in Fig. 5.5. Anytime a bone is transformed around its joint, the subsequent bones along the branch need to be updated as well. For example, in Fig. 5.5, if bone 4 is rotated, then bones 5, 6, and 7 are moved to keep their pose relative to bone 4. We refer to the branch that starts at a bone j as branch j. A branch weight map $\bar{\mathbf{W}}$ can be calculated from the bone weight map:

$$\bar{\mathbf{W}}(i,b) = \sum_{j \in B(b)} \mathbf{W}(i,j), \tag{5.1}$$

where b is the current branch and $B(b)$ is the set of bones in a branch. We use branch weights when calculating how well the vertices in a given branch fit the target frame geometry.

For each bone, we specify a range of motion under which it can be rotated relative to its parent bone. A parent bone p_j is defined as the bone connected to bone j that is higher in the skeleton tree and connected to the lead joint of bone j. For example, in Fig. 5.5, bone 5 is the parent of bone 6. We define a local right-handed coordinate system for each bone in the original pose P_0 of the template, shown in Fig. 5.5, that places each joint at the origin and the bone out of that joint on the y-axis as shown in Fig. 5.6. The constraint set for each joint is generated by inspecting each joint in the template under varying amounts of rotation and selecting a rotation range for the joint relative to its parent that falls within the physical range of motion for a human subject.

Specifically, the constraints set for each joint are defined as minimum and maximum limits of rotation on each of the x, y, and z-axes relative to its parent joint. The rotation constraints can be applied directly only when a joint rotates about a single coordinate axis. Since in practice, this is not often the case, we need to compute a different constraint representation based on the values from the individual axis rotation constraints. To do this, we represent the constraint for each joint in two parts. The first component $\mathbf{R}_{\text{joint}}$ is a rotation matrix that represents the portion of the joint rotation that occurs around an axis lying in the x-z plane of the joint's local coordinate system as shown in Fig. 5.6. This rotation moves the bone from the y-axis to its new position shown in red in Fig. 5.6. The second component of the joint rotation, represented by the rotation matrix $\mathbf{R}_{\text{bone}}$, rotates around the new axis of the bone shown in red in Fig. 5.6. Thus, the total transformation

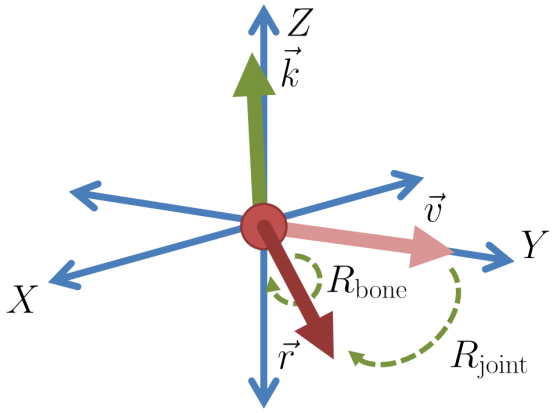


Figure 5.6: An example of the decomposition of $\mathbf{R}_{\text{local}}$ into $\mathbf{R}_{\text{joint}}$ and $\mathbf{R}_{\text{bone}}$. The original pose of a bone in P_0 always lies along the *y*-axis with unit vector $\vec{v}$. The axis of rotation used to rotate between $\vec{v}$ and final bone position $\vec{r}$ is $\vec{k}$.

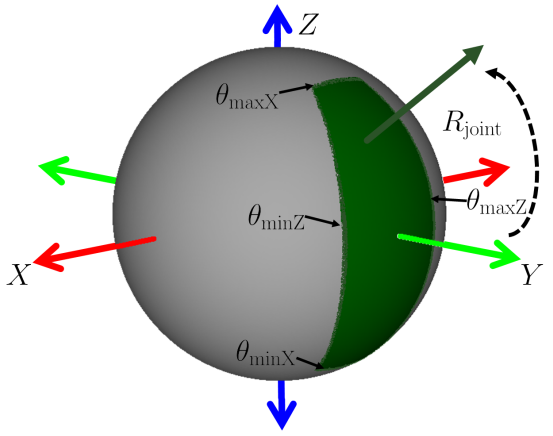
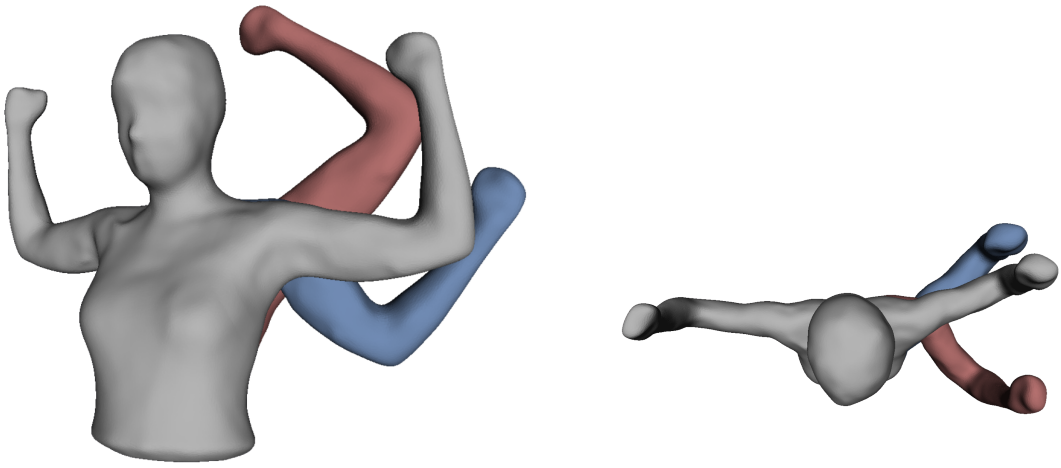


Figure 5.7: Illustration of a constrained region for a skeleton joint. The bone out of a joint is always aligned with the *y*-axis in the original template pose. The green region shows valid new positions into which $\mathbf{R}_{\text{joint}}$ can rotate the bone. The four sides of the valid region are constrained by the rotation limits around the *x* and *z*-axes.



(a) (b)

Figure 5.8: An example shoulder joint rotated (a) along the x-axis and (b) along the z-axis of the local coordinate frame. The figure in gray is the original pose. The red model represents a positive rotation around the axis and blue represents a negative rotation.

$\mathbf{R}_{\text{local}}$ is given by:

$$\mathbf{R}_{\text{local}} = \mathbf{R}_{\text{bone}}\mathbf{R}_{\text{joint}}. \tag{5.2}$$

We refer to this total transformation as $\mathbf{R}_{\text{local}}$ since it denotes the transformation of the joint with respect to its own local coordinate frame. Since the rotation $\mathbf{R}_{\text{joint}}$ rotates around an axis that lies in the x-z plane, no component of $\mathbf{R}_{\text{joint}}$ rotates around the y-axis. Therefore, the rotation matrix $\mathbf{R}_{\text{joint}}$ is constrained by the individual minimum and maximum constraints for the x-axis and z-axis. As seen in Fig. 5.7, the valid range in which $\mathbf{R}_{\text{joint}}$ can be defined is constrained by the regions that fall within the minimum and maximum angles of rotation for the x-axis, namely $\theta_{\text{min}X}$ and $\theta_{\text{max}X}$, and the z-axis, namely $\theta_{\text{min}Z}$ and $\theta_{\text{max}Z}$. Once the bone has been put in position by $\mathbf{R}_{\text{joint}}$, we constrain the amount of rotation around the bone by using the angle limits for the y-axis, since rotating around the y-axis in the original pose is the same as rotating around the bone axis in the pose after $\mathbf{R}_{\text{joint}}$ is applied. We refer to the minimum and maximum limits of rotation around the bone axis as θ_{minBone} and θ_{maxBone} respectively.

We choose this representation for joint rotations because it is intuitive to constrain. For example, for the joint at the wrist, we expect a limited range of rotation in the x-z plane and zero rotation around the bone axis. Even with correct alignment of the skeleton to existing geometry, joint constraints are important. For regions with significant missing geometry due to occlusion, for example, the constraints keep the skeleton in a reasonable pose until the true geometry can be observed again. Table 5.1 shows a sample set of motion constraints for the 12 joints in our template. To generate the values in the table, the template is deformed at each joint along each of the x, y, and z-axes, is visualized over a range of deformation angles from $[-\pi, \pi)$, and the subset of this

	Rotation Limits (radians)					
Joint	$\theta_{\text{min}X}$	$\theta_{\text{max}X}$	$\theta_{\text{min}Z}$	$\theta_{\text{max}Z}$	θ_{minBone}	θ_{maxBone}
0	-3.14	3.14	-3.14	3.14	-3.14	3.14
1	-0.4	0.4	-0.4	0.4	-0.4	0.4
2	-0.6	0.4	-0.4	0.4	-0.8	0.8
3	-0.4	0.4	-0.4	0.4	-1.4	1.4
4	0.0	0.4	-0.2	0.2	-0.4	0.4
5	-1.0	1.2	-0.5	1.4	-2.8	0.4
6	-0.8	0.8	-1.4	1.1	-0.8	1.6
7	-0.4	0.4	-0.4	0.4	0.0	0.0
8	-0.1	0.4	-0.2	0.2	-0.4	0.4
9	-1.0	1.2	-1.4	0.4	-0.4	2.8
10	-1.0	1.5	-1.2	1.0	-0.4	1.2
11	-0.2	0.4	-0.4	0.8	0.0	0.0

Table 5.1: A sample set of constraints used for the template.

range of angles that appears as a physically plausible limit is selected. The table is populated by iteratively examining each joint over the three possible axes. Fig. 5.8 presents sample poses of a shoulder joint rotated about the local x-axis and z-axis.

5.3 Processing a Sequence of Frames

In this section, we describe our method for fitting a template to captured 3D scans. Reconstructing the dynamic shape of a human subject occurs in several nested iterative processes. At the top level, we deform the template to each fused 3D scan resulting from one round of three stations capturing the scene. In doing so, we iteratively update the orientation of each bone in the template to bring it into alignment with the 3D captured geometry. We repeat this process for each consecutive new set of data.

The human subject is captured from only one station at a time at a rate of 60 Hz. At each station, two cameras generate point clouds representing their views of the scene. Each of these two points clouds are meshed into surfaces, which in turn allows for normals to be estimated for each vertex. We merge the six 3D surfaces generated in three consecutive time steps, two from each station, into a partial surface reconstruction of the scene. We refer to the merged partial surfaces captured at time t as a frame F_t. The merged surfaces are not re-meshed to create a single surface; rather, the partial reconstruction is represented as a set of multiple disconnected surfaces. In the process of fitting the template to the partial reconstructions, the surface connectivity of the partial scans is less important than the normal and location of each vertex in the partial surface since these are used for finding correspondences.

To capture the overall movement of the human subject over time, the template is sequentially

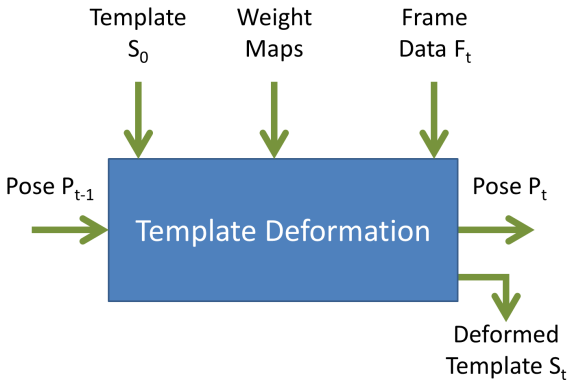


Figure 5.9: The basic input and output of the iterative mesh deformation method.

fit to each frame as shown in Fig. 5.9. The captured data at time t, denoted by F_t, is processed to compute the pose of the human subject at time t, denoted by P_t. We refer to the template in its original pose, as shown in Fig. 5.5, as S_0 and the corresponding original skeleton pose itself as P_0. The final skeleton pose that aligns S_0 with frame F_t is referred to as P_t. As seen in Fig. 5.9, given the template S_0, the captured frame F_t, the weight maps $\mathbf{W}$ and $\bar{\mathbf{W}}$, and the pose P_{t-1} at time $t-1$, we deform the template S_0 to fit the target frame F_t and generate pose P_t and deformed template S_t as output. The steps for template deformation are shown in the flow diagram of Fig. 5.10. At a high level, the template deformation takes place in three main steps: initialization, processing, and finalizing. During initialization, all data needed for processing is loaded and the template is deformed to the pose from the previous frame. Within the processing step, the template is evaluated to determine whether it closely fits the target frame. If it does not, the branch fitting loop is executed. During branch fitting, the poses of branches within the skeleton are iteratively updated to align to the target geometry. After each pass through branch fitting, the fit between the source template and the target frame is reevaluated. Once a good fit is found, the processing step is concluded and the finalizing step saves the pose and final template for processing the next frame. We now explain each step in more detail.

During initialization, the template S_0, vertex weight map $\mathbf{W}$, branch weight map $\bar{\mathbf{W}}$, skeleton pose from the previous time step P_{t-1}, and frame F_t are all loaded. The pose of each bone is represented as a rotation matrix plus translation vector. The pose of a skeleton P_t specifies the transformation of each bone from the original template pose P_0 to its position and orientation at time t, and it is represented by the set of rotation and translation vectors for each bone. While initializing for time instance t, the template S_0 is deformed according to the skeleton pose P_{t-1} that was aligned to frame F_{t-1}. Since the frames are processed sequentially, pose P_{t-1} is the best initial estimate for target frame F_t. The bone weight map $\mathbf{W}$ assigning the influence of each bone on each template vertex is needed to properly deform the template. The vertices of the template are updated to match the skeleton pose using dual quaternion skinning. We refer to this updated template as S_t^{curr} since it is the current best estimate for aligning with F_t.

This deformed template is used as input to the "processing" portion of our algorithm shown

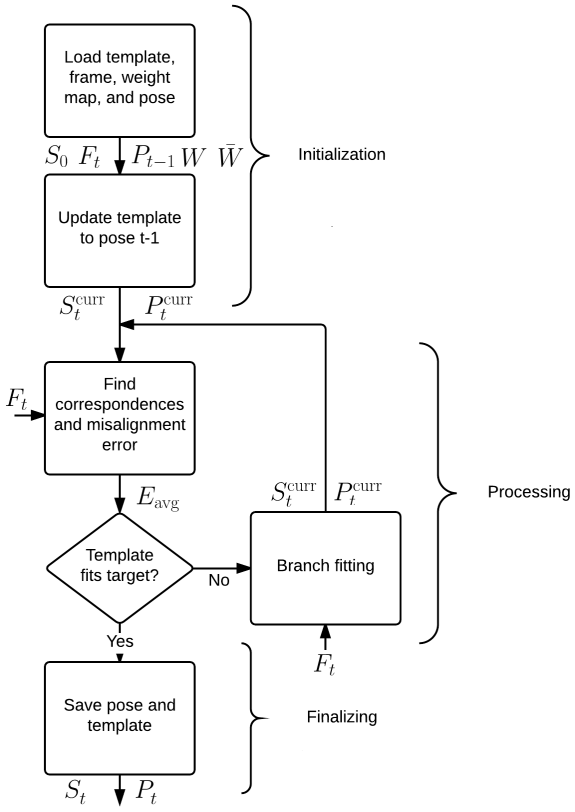


Figure 5.10: Processing steps for the template deformation block shown in Fig. 5.9.

in Fig. 5.10. Correspondences from template S_t^{curr} to the frame F_t are generated, and the average distance between all correspondence points is found. Specifically, a correspondence for each vertex in frame F_t is found by locating the closest vertex in template S_t^{curr} that lies in the direction of the frame vertex's normal. To prevent erroneous correspondences, the maximum distance is kept below a preset threshold. The average misalignment distance over all correspondence pairs is calculated and represented as E_{avg}. If this average error is small, then template S_t^{curr} is well aligned with the frame F_t and the finalizing step can start. If S_t^{curr} it not yet well aligned with F_t, then the branch fitting process must be executed. In the branch fitting process to be described in Section 5.3.1, the pose of each branch of the skeleton P_t^{curr} is updated to bring template S_t^{curr} into alignment with the frame F_t. For each frame, the branch fitting process is executed multiple times until a reasonable fit is found.

In the finalizing step, the pose of the skeleton P_t^{curr} that transforms the source template S_0 into alignment with the frame F_t is stored along with the deformed template S_t^{curr}. The deformed template itself is not used during subsequent processing, but its pose is used during the initialization step of the following frame. The deformed template is the main output of our system corresponding to the dynamic geometry of the human subject. The heart of our approach for template deformation lies within the branch fitting process to be described next.

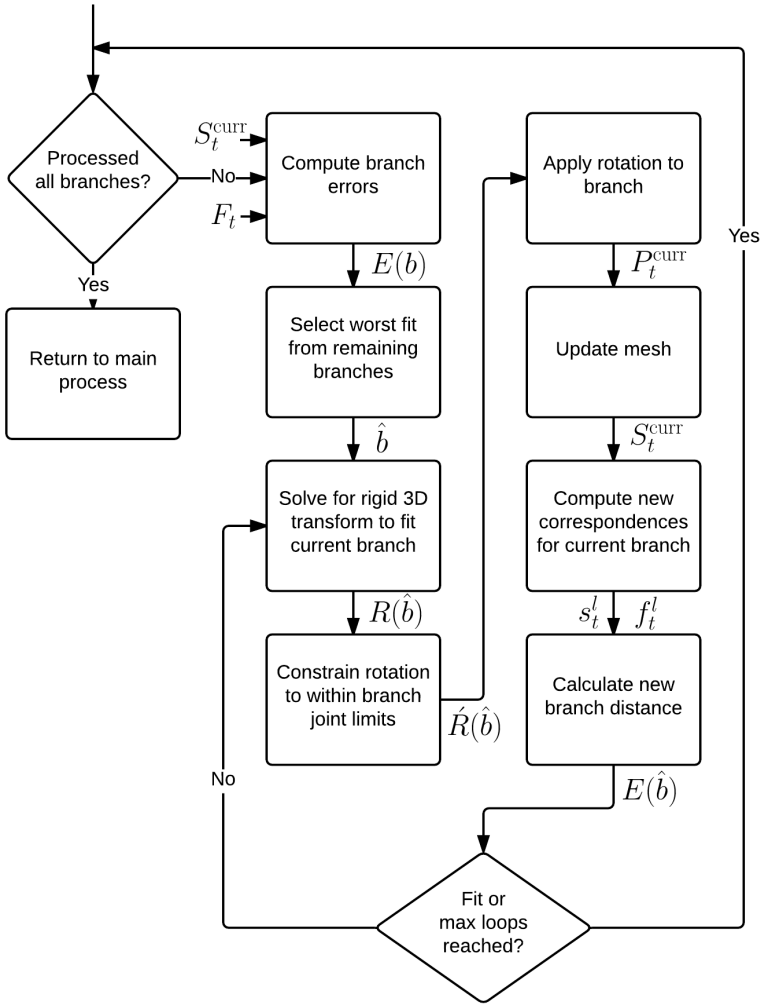


Figure 5.11: Processing steps for the branch fitting block showed in 5.10.

5.3.1 Branch Fitting

The branch fitting process is illustrated in Fig. 5.11. At a high level, the pose of each branch in the template skeleton is updated to bring the source template into alignment with the target frame. This process is run until each branch in the skeleton has been updated. The order in which the branches are processed is determined by the misalignment error for each branch, starting with the worst fit branch and continuing in order of decreasing error.

The branch fitting process begins by checking whether all the branches have been processed. If they have not, the error for each branch is calculated. To do this, correspondences between the template S_t^{curr} and captured frame F_t are found as described earlier. We denote the correspondence i as a vertex s_t^i in the template S_t^{curr} and a vertex f_t^i in frame F_t. The Euclidean distances between each corresponding vertex pair in the source and target $d(s_t^i, f_t^i)$ are computed. Using these values,

the average misalignment of correspondences of each branch b is

$$E(b) = \frac{\sum_{i \in N} d(s_t^i, f_t^i) \bar{\mathbf{W}}(i, b)}{\sum_{i \in N} \bar{\mathbf{W}}(i, b)}. \tag{5.3}$$

Each distance term is weighted by the branch weight map $\bar{\mathbf{W}}(i, b)$, which represents the total influence of branch b on the vertex position s_t^i. We select the most misaligned branch $\hat{b}$ and correct it before any other branch:

$$\hat{b} = \arg\min_{b \in \mathbb{B}} E(b), \tag{5.4}$$

where $\mathbb{B}$ is the set of possible branches.

Once the branch is selected, the transformation to align the correspondences between the template S_t^{curr} and frame F_t^{curr} in the branch is estimated by solving for a rotation and translation. Specifically, we use a standard 3D rigid transform to align the correspondences for vertices in branch $\hat{b}$ [15]. The 3D correspondence pair for vertex l that is fed into the transform is weighted as

$$w_{t,l} = \begin{cases} d(s_t^l, f_t^l) n_t^{s,l\mathsf{T}} n_t^{f,l} & \text{if } n_t^{s,l\mathsf{T}} n_t^{f,l} > 0, \\ 0 & \text{if } n_t^{s,l\mathsf{T}} n_t^{f,l} \leq 0, \end{cases} \tag{5.5}$$

where $n_t^{s,l}$ and $n_t^{f,l}$ are the normals of the vertices in the target and source, respectively. A correspondence weight is calculated for each vertex l in the set of vertices in branch $\hat{b}$. The correspondence weights are proportional to the distance between s_t^l and f_t^l in a correspondence pair. Correspondences with a large distance between the source and target vertices occur in regions where the source template is most misaligned with respect to the target frame, i.e. the regions that are most important to fit. To ensure correct correspondences, we penalize those without similar normals. Specifically, if the normals are more than 90 degrees apart, then they are likely matching the incorrect side of a surface and are thus ignored. Before computing the rotation to align the correspondences, the vertices of the source and target are translated such that the origin of the world coordinate frame is centered at the lead joint of branch $\hat{b}$. The resulting rotation matrix $\mathbf{R}(\hat{b})$ and translation vector $\vec{T}(\hat{b})$ computed from the 3D transformation represent the update transformation of the branch in the world coordinate system centered at the lead joint of branch $\hat{b}$ [15]. Fig. 5.12 illustrates the forearm of a human template, shown in gray, being fit to target geometry, shown as red dots in the figure. By applying the rotation $\mathbf{R}(\hat{b})$, the forearm of the template is aligned with the target geometry. The rotation $\mathbf{R}(\hat{b})$ is applied in the world coordinate frame centered at the lead joint of branch $\hat{b}$. When fitting all branches other than the root branch, we discard the translation vector $\vec{T}(\hat{b})$ since the position of a branch's lead joint is determined by the parent bone connected to it. For example, in Fig. 5.5, the position of the lead joint of bone 7 is determined by the position and orientation of bone 6.

Once the update rotation $\mathbf{R}(\hat{b})$ is estimated, the next step is to ensure that the rotation is within the constrained limit for the branch. This is described in detail in Section 5.3.2. Constraining the update rotation of a branch results in a new rotation matrix $\acute{\mathbf{R}}(\hat{b})$. With the new rotation, the

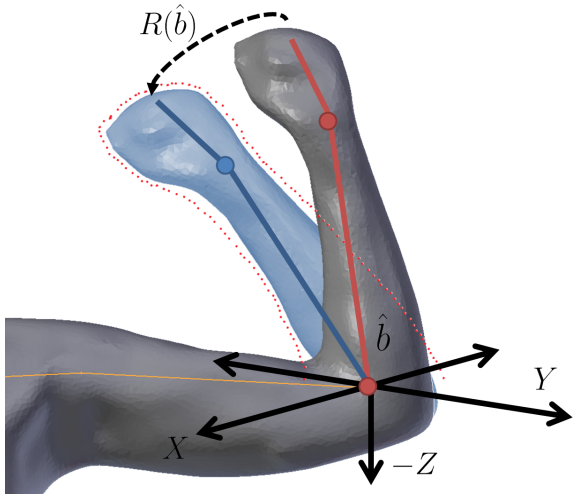


Figure 5.12: An example of branch fitting. The forearm branch of a template is rotated by $\mathbf{R}(\hat{b})$ to align with points in the target geometry shown as red dots. The new template pose after applying $\mathbf{R}(\hat{b})$ is shown in blue. The rotation occurs in the world coordinate frame with the origin positioned at the lead joint of the rotating branch.

position of branch $\hat{b}$ is updated in P_t^{curr} and the template S_t^{curr} is updated according to this new pose. As done in the initialization, the template S_t^{curr} is deformed using dual quaternion skinning.

Given the new pose for the template S_t^{curr}, we recompute the correspondences for the updated branch $\hat{b}$, the one just rotated, and compute the average misalignment error for the branch using Equation 5.3. The process is repeated until the average misalignment error for $\hat{b}$ drops below a certain threshold for a preset number of iterations. Once this happens, we return to the beginning of the branch fitting process and check to see whether all branches have been fit. If they have, we exit the branch fitting process, otherwise, the next branch is processed. After all branches are processed, we exit the branch fitting process.

5.3.2 Constraining a Branch Rotation

At the time the update rotation $\mathbf{R}(\hat{b})$ of branch $\hat{b}$ is calculated in Fig. 5.11, the skeleton is in pose P_t^{curr} and the template S_t^{curr} reflects this pose. We need to apply $\mathbf{R}(\hat{b})$ to rotate branch $\hat{b}$ to better align the template mesh S_t^{curr} with the target frame F_t. The pose P_t^{curr} is represented by a set of rotation matrices $\mathbf{R}_t^{\text{curr}}(b)$ for each branch b that specify the amount of rotation the branch undergoes in the process of moving from the original template pose P_0 to the current pose P_t^{curr}. The rotations $\mathbf{R}_t^{\text{curr}}(b)$ are all represented in the world coordinate frame since this is the coordinate frame of the template S_t^{curr} and target F_t. On the other hand, to test whether a branch $\hat{b}$ satisfies its specified constraints, its rotation must be represented in the local coordinate frame of branch $\hat{b}$ in pose P_0 since the constraints of each branch are defined in this local coordinate frame. Therefore, to apply any constraint in the rotation of bones, we must reconcile these two coordinate systems.

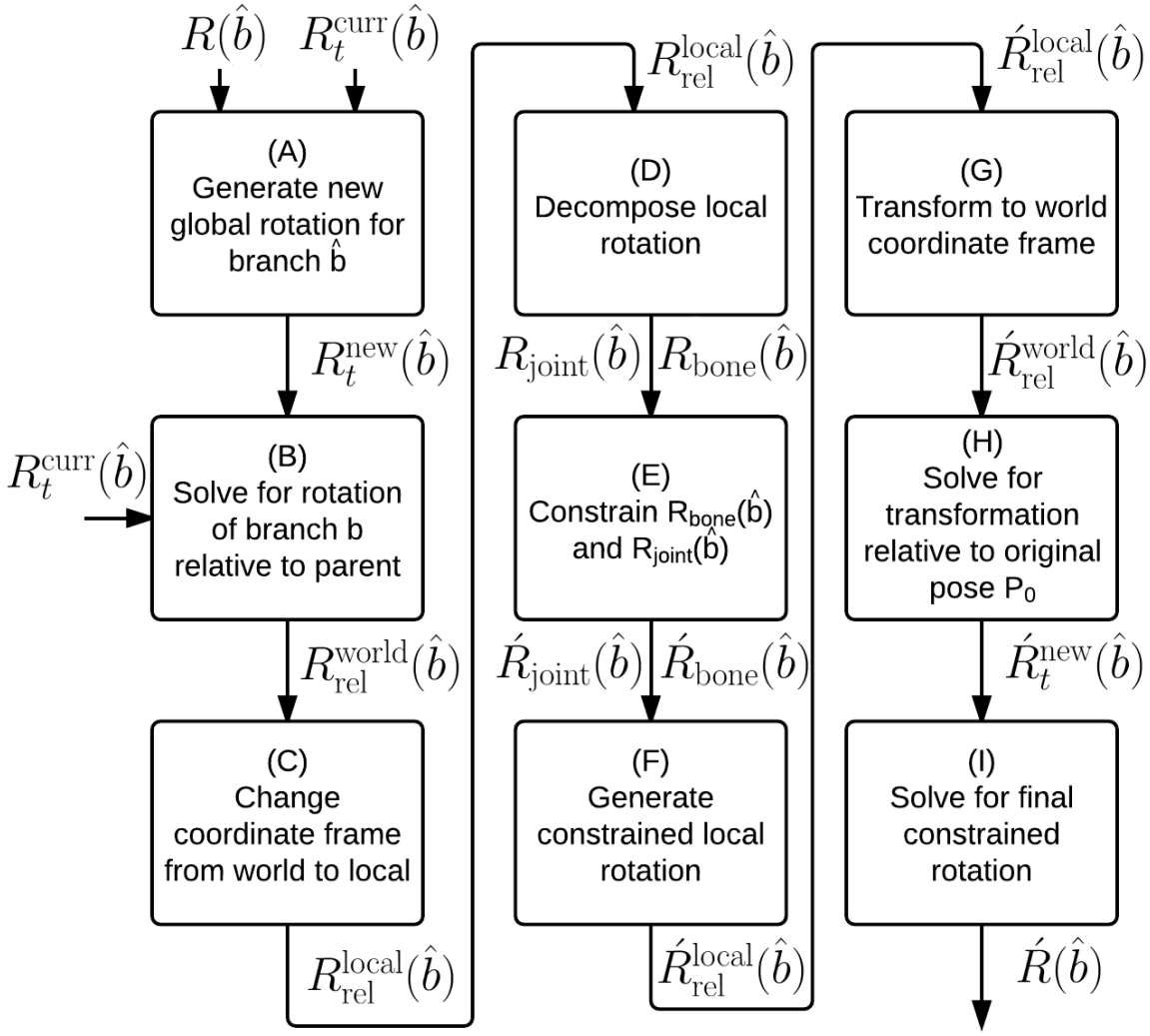


Figure 5.13: Processing steps for the constraining a branch rotation block shown in 5.11.

The steps for constraining a branch rotation are illustrated in Fig. 5.13 and can be summarized as follows. In step A, we concatenate incremental rotation $\mathbf{R}(\hat{b})$ from Fig. 5.11 with the current rotation $\mathbf{R}_t^{\text{curr}}$ of the skeleton to determine the new rotation $\mathbf{R}_t^{\text{new}}(\hat{b})$ of branch $\hat{b}$ in the world coordinate system with respect to its pose in P_0. Then, in step B, we solve for the rotation of branch $\hat{b}$ relative to its parent branch $\hat{p}$ and represent it as $\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b})$. This is because the constraints of each branch are defined relative to the orientation of their parent branch. Since the relative rotation must be represented in the local coordinate frame where the constraints are defined, in step C, the coordinate frame of the relative transformation $\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b})$ is converted from the world coordinate system to the local coordinate frame of branch $\hat{b}$ in pose P_0 denoted by $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$. In step D, the local rotation is decomposed into the two rotation components $\mathbf{R}_{\text{bone}}(\hat{b})$ and $\mathbf{R}_{\text{joint}}(\hat{b})$ presented in Section 5.2. In step E, we constrain the two rotation components to meet the branch constraints

resulting in $\acute{\mathbf{R}}_{\text{bone}}(\hat{b})$ and $\acute{\mathbf{R}}_{\text{joint}}(\hat{b})$. From here, the inverse of steps A–D are visited in reverse order. First in step F, the constrained local relative rotation $\acute{\mathbf{R}}_{\text{rel}}^{\text{local}}(\hat{b})$ is generated by composing $\acute{\mathbf{R}}_{\text{bone}}(\hat{b})$ and $\acute{\mathbf{R}}_{\text{joint}}$ and then, in step G, it is transformed to the world coordinate system $\acute{\mathbf{R}}_{\text{rel}}^{\text{world}}(\hat{b})$. In step H, the transformation is computed relative to the original pose P_0 rather than to the parent branch to generate $\acute{\mathbf{R}}_t^{\text{new}}(\hat{b})$, and, in step I, we remove the $\mathbf{R}_t^{\text{curr}}$ from $\acute{\mathbf{R}}_t^{\text{new}}(\hat{b})$ to obtain the final constrained rotation $\acute{\mathbf{R}}(\hat{b})$. We now explain each step in more detail.

The rotation $\mathbf{R}_t^{\text{curr}}(\hat{b})$ represents the rotation that branch $\hat{b}$ has undergone in moving from pose P_0 to pose P_t^{curr} in the world coordinate system. The update rotation $\mathbf{R}(\hat{b})$ in Fig. 5.11 is the incremental rotation we apply in addition to $\mathbf{R}_t^{\text{curr}}(\hat{b})$ to better align the vertices of branch $\hat{b}$ in S_t^{curr} with F_t. In step A, the total rotation $\mathbf{R}_t^{\text{new}}(\hat{b})$ for branch $\hat{b}$ is the concatenation of these two rotations as follows:

$$\mathbf{R}_t^{\text{new}}(\hat{b}) = \mathbf{R}(\hat{b})\mathbf{R}_t^{\text{curr}}(\hat{b}). \tag{5.6}$$

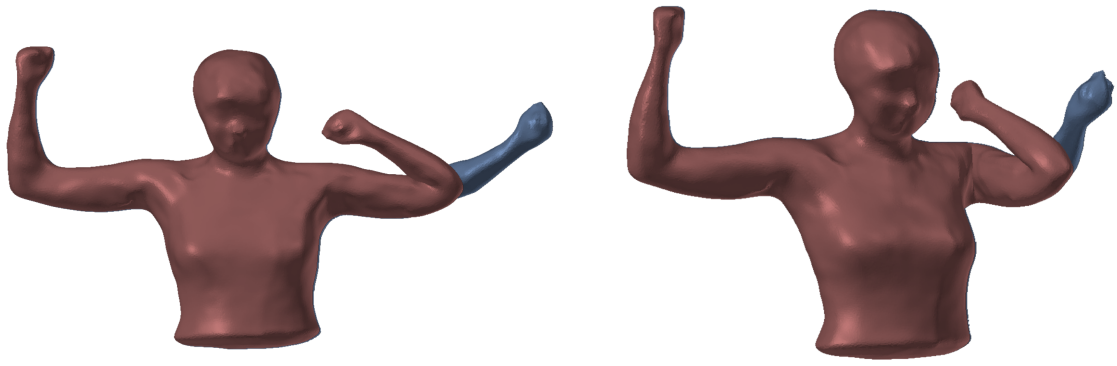
The rotation $\mathbf{R}_t^{\text{new}}(\hat{b})$ for branch $\hat{b}$ is represented in the world coordinate system. It describes the rotation of the branch in the same coordinate system as the template S_t^{curr} and target frame F_t. The rotation $\mathbf{R}_t^{\text{new}}(\hat{b})$ specifies how much branch $\hat{b}$ is being rotated from its orientation in pose P_0. Rather than represent the rotation of the branch with respect to its orientation in pose P_0, we would like to capture the rotation of branch $\hat{b}$ relative to its parent branch $\hat{p}$. This is because the constraints defined for each branch are represented as the amount of rotation a branch can undergo relative to its parent branch. The need for this representation is illustrated in Fig. 5.14. For a human posed with the upper arm extending out to the side, there is a set of rotations in the world coordinate system that describes the possible movement of the forearm, as shown in Fig. 5.14a. If the upper arm is extended to the front of the body as shown in Fig. 5.14b, the range of possible rotations for the forearm in the world coordinate system would now be different. However, the relative rotation of the forearm with respect to the upper arm would be the same in these two situations. We are interested in calculating this relative rotation since our constraints for a branch are defined in this relative manner. As such, in step B, the relative rotation $\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b})$ of branch $\hat{b}$ to the parent branch $\hat{p}$ is computed as

$$\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b}) = \mathbf{R}_t^{\text{new}}(\hat{b})\mathbf{R}_t^{\text{curr}}(\hat{p})^{-1}. \tag{5.7}$$

To illustrate the method of relative rotations, we present an example where branch $\hat{b} = 6$ and $\hat{p} = 5$ as labeled in Fig. 5.5. If a template takes on pose P_t the corresponding rotation matrices to generate this pose from P_0 are represented as $\mathbf{R}_t(b),\ \forall\ b \in \mathbb{B}$. When the template is in pose P_0 as shown in Fig. 5.15a, $\mathbf{R}_{t=0}(\hat{b}) = \mathbf{R}_{t=0}(\hat{p}) = \mathbf{I}$, the identity matrix. In Fig. 5.15b, we depict a rotation about the branch $\hat{p}$ at time $t = 1$. In this pose, both branches 5 and 6 are rotated from their orientation in pose P_0 by a rotation of $\mathbf{R}_1$, that is $\mathbf{R}_{t=0}(\hat{p}) \neq \mathbf{R}_{t=1}(\hat{p})$, but $\mathbf{R}_{t=1}(\hat{p}) = \mathbf{R}_{t=1}(\hat{b}) = \mathbf{R}_1$ since both branches have undergone the same rotation. If now the branch $\hat{b}$ is rotated at time $t = 2$ by a rotation of $\mathbf{R_2}$, as shown in Fig. 5.15c, then $\mathbf{R}_{t=1}(\hat{p}) = \mathbf{R}_{t=2}(\hat{p}) = \mathbf{R}_1$, but $\mathbf{R}_{t=2}(\hat{b}) \neq \mathbf{R}_{t=2}(\hat{p})$. Obviously $\mathbf{R}_{t=2}(\hat{b}) = \mathbf{R}_2\mathbf{R}_1$ and $\mathbf{R}_{t=2}(\hat{p}) = \mathbf{R}_1$. The rotation of branch $\hat{b}$ relative to $\hat{p}$ for pose P_2 can be determined by using Equation 5.7:

$$\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b}) = \mathbf{R}_{t=2}(\hat{b})\mathbf{R}_{t=2}(\hat{p})^{-1} = \mathbf{R}_2\mathbf{R}_1\mathbf{R}_1^{-1} = \mathbf{R}_2. \tag{5.8}$$

Thus, we are able to isolate the rotation of branch $\hat{b}$ relative to its parent branch $\hat{p}$ by using Equation



(a) (b)

Figure 5.14: An example range of rotation for an elbow joint is shown with the upper arm positioned (a) towards the side of the body and (b) towards the front of the body. Note that range of motion for the elbow is the same regardless of the position of the upper arm.

5.7. Additionally, since all rotations $\mathbf{R}_t(\hat{b})$ are defined in the world coordinate system, the relative rotation $\mathbf{R}_{\text{rel}}^{\text{world}}$ is also represented in the world coordinate system.

In step C, we convert the relative rotation $\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b})$ from the world coordinate frame to the local coordinate frame of branch $\hat{b}$ in pose P_0. The coordinate transformation between a branch $\hat{b}$ in the local coordinate frame of pose P_0 and the world coordinate frame is represented as $\mathbf{R}_{l\to w}(\hat{b})$. We use this coordinate transformation to convert the relative rotation from world coordinates to local coordinates:

$$\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b}) = \mathbf{R}_{l\to w}(\hat{b})^{-1}\mathbf{R}_{\text{rel}}^{\text{world}}(\hat{b})\mathbf{R}_{l\to w}(\hat{b}). \tag{5.9}$$

The relative rotation is now represented in the correct local coordinate frame, but it needs to be decomposed into a different representation before the constraints can be tested. In step D, we decompose $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$ into $\mathbf{R}_{\text{joint}}(\hat{b})$ and $\mathbf{R}_{\text{bone}}(\hat{b})$, as in Section 5.2, to match the same form in which the branch constraints are defined. The constraints for $\mathbf{R}_{\text{joint}}(\hat{b})$ are specified as allowable ranges of rotation around the x-axis and z-axis as shown in Fig. 5.7. The constraint for $\mathbf{R}_{\text{bone}}(\hat{b})$ is specified as an allowable range of rotation around the axis of the bone, as shown in Fig. 5.6. The rotation $\mathbf{R}_{\text{joint}}(\hat{b})$ captures the new position to which a bone would be moved to if rotated by $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$. The rotation $\mathbf{R}_{\text{bone}}(\hat{b})$ accounts for any rotations around this bone caused by $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$. Deriving $\mathbf{R}_{\text{joint}}(\hat{b})$ and $\mathbf{R}_{\text{bone}}(\hat{b})$ from $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$ is performed in two steps. First, we solve for $\mathbf{R}_{\text{joint}}(\hat{b})$ and then generate $\mathbf{R}_{\text{bone}}(\hat{b})$ by using $\mathbf{R}_{\text{joint}}(\hat{b})$ and $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$.

The rotation $\mathbf{R}_{\text{joint}}(\hat{b})$ captures the movement of the bone to a new location caused by $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$. In the local coordinate frame of each branch in original pose P_0, the bone out of each joint lies along the y-axis as shown in Fig. 5.6. We can determine the new position of the bone by transforming a y-direction unit vector by $\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b})$ as follows:

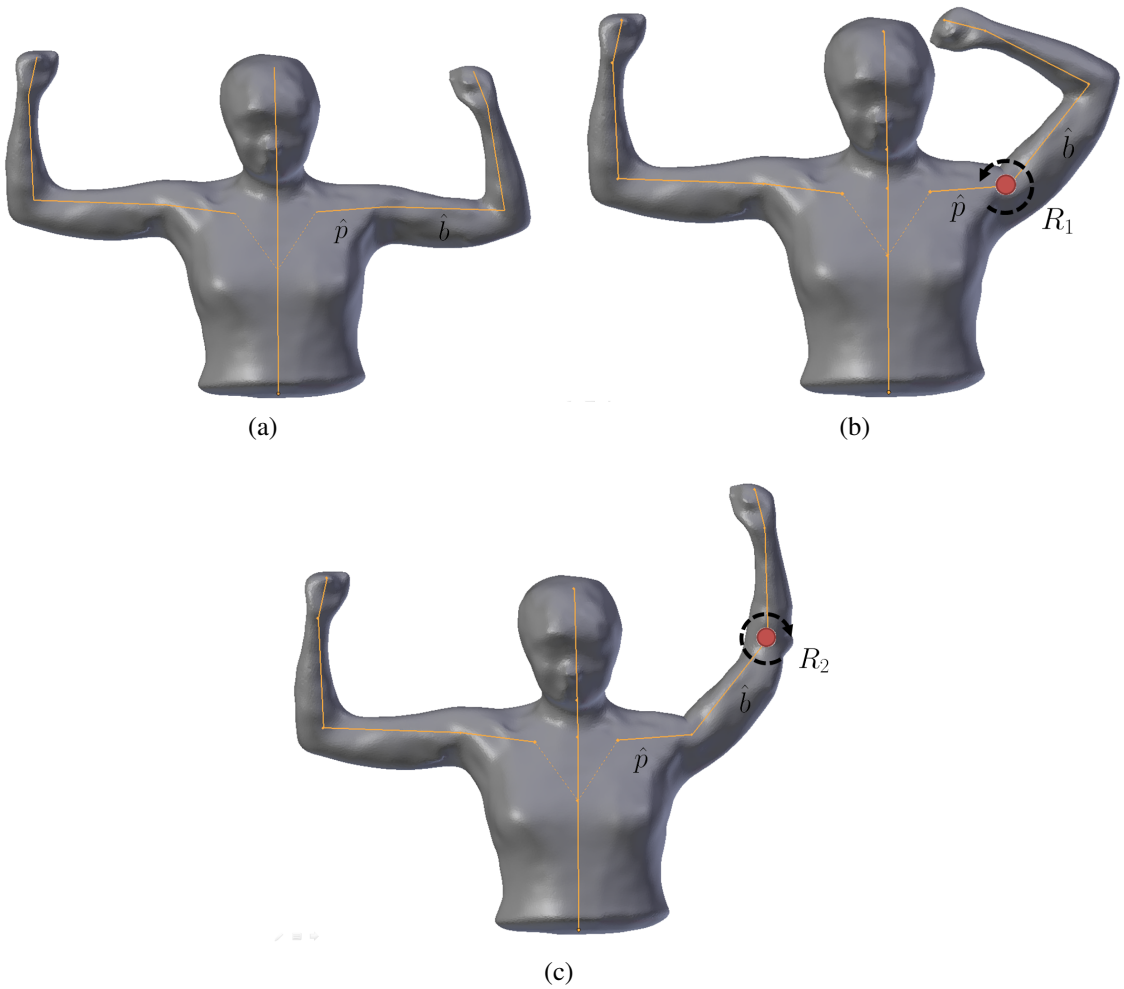


Figure 5.15: Illustration of the template in three consecutive poses: (a) the template fit to pose P_0, (b) pose P_1 where branch $\hat{p} = 5$ is rotated, and (c) pose P_2 where branch $\hat{b} = 6$ is rotated.

$$\vec{r} = \left[\mathbf{R}_{\text{rel}}^{\text{local}}(\hat{b}).\right] \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{5.10}$$

The vector $\vec{r}$ accounts for the new location of the bone, but it does not represent the amount of rotation around the axis of the bone. We wish to solve for the rotation needed to transform the original pose of the bone in P_0, a unit-length vector on the y-axis, to the new position $\vec{r}$ without any rotation around the y-axis. The rotation matrix $\mathbf{R}_{\text{joint}}(\hat{b})$ can be represented in an axis-angle representation, where $\vec{k}$ is the axis of rotation as shown in Fig. 5.6 and θ_{joint} is the angle of rotation about the axis. The axis of rotation for $\mathbf{R}_{\text{joint}}(\hat{b})$ must lie in the x-z plane of the joint's local coordinate system to effectively separate the rotation about the bone axis in $\mathbf{R}_{\text{bone}}(\hat{b})$. To trans-

form a y-axis unit vector to align with vector $\vec{r} = [r_x r_y r_z]^\intercal$, the axis of rotation $\vec{k}$ is defined as $[r_z\ 0\ -r_x]^\intercal / |[r_z\ 0\ -r_x]|$. The vector $\vec{k}$ is perpendicular to $\vec{r}$ and lies in the x-z plane as shown in Fig. 5.6. By rotating a y-axis unit vector around $\vec{k}$, the y-axis unit vector can be aligned with $\vec{r}$. We set the original direction of the bone vector to $\vec{v} = [0\ 1\ 0]^\intercal$, and we use the Rodrigues' rotation formula to solve for θ_{joint} [60]:

$$\vec{r} = \vec{v}\cos\theta_{\text{joint}} + (\vec{k} \times \vec{v})\sin\theta_{\text{joint}} + \vec{k}(\vec{k} \cdot \vec{v})(1 - \cos\theta_{\text{joint}}). \tag{5.11}$$

Given the rotation vector is $\vec{k}$ and the angle of rotation is θ_{joint}, we can solve for the rotation matrix $\mathbf{R}_{\text{joint}}(\hat{b})$ using the matrix representation of the Rodrigues' formula [60]:

$$\mathbf{R}_{\text{joint}}(\hat{b}) = \mathbf{I} + (sin\theta_{\text{joint}})\mathbf{K} + (1 - cos\theta_{\text{joint}})\mathbf{K}^2, \tag{5.12}$$

where $\mathbf{K}$ is the cross-product matrix for $\vec{k}$ and $\mathbf{I}$ is the 3×3 identity matrix. Given $\mathbf{R}^{\text{local}}_{\text{rel}}(\hat{b})$ and $\mathbf{R}_{\text{joint}}(\hat{b})$, we can solve for $\mathbf{R}_{\text{bone}}(\hat{b})$ using Equation 5.2:

$$\mathbf{R}_{\text{bone}}(\hat{b}) = \mathbf{R}^{\text{local}}_{\text{rel}}(\hat{b})\mathbf{R}^{-1}_{\text{joint}}(\hat{b}) \tag{5.13}$$

At this point we have converted the local rotation $\mathbf{R}^{\text{local}}_{\text{rel}}(\hat{b})$ into the proper form of our constraints $\mathbf{R}_{\text{bone}}(\hat{b})$ and $\mathbf{R}_{\text{joint}}(\hat{b})$. In step E, the individual rotations are compared against the branch constraints in Table 5.1 and modified to fit as necessary. First, we check whether the position of $\vec{r}$ falls within a valid rotation range for the x and z axes. The angle between the y-axis and vector $\vec{r}$ is used to calculate θ_X and θ_Z, the rotation of $\vec{R}$ around the x-axis and z-axis, respectively. As shown in Fig. 5.7, there is a range of valid positions into which the branch can be rotated, shown in green in the figure. Specifically, we must ensure that $\theta_{\text{min}X} < \theta_X < \theta_{\text{max}X}$ and $\theta_{\text{min}Z} < \theta_Z < \theta_{\text{max}Z}$. If $\vec{r}$ falls outside of the constrained region, it is moved into a valid position. Using Rodrigues' formula in Equation 5.11, the value of θ_{joint} is updated to a new angle $\acute{\theta}_{\text{joint}}$ to move $\vec{r}$ into a valid new position $\acute{\vec{r}}$. Applying Equation 5.12, the new angle $\acute{\theta}_{\text{joint}}$ and $\vec{k}$ are used to generate $\acute{\mathbf{R}}_{\text{joint}}(\hat{b})$.

Next, we convert $\mathbf{R}_{\text{bone}}(\hat{b})$ into its axis-angle representation. By definition, $\mathbf{R}_{\text{bone}}(\hat{b})$ must rotate around the axis of the bone. We determine the angle of rotation θ_{bone} for $\mathbf{R}_{\text{bone}}(\hat{b})$ and ensure it meets the constraints $\theta_{\text{minBone}} < \theta_{\text{bone}} < \theta_{\text{maxBone}}$. The rotation θ_{bone} is found using the matrix to axis-angle conversion for rotation matrices[60]:

$$\theta_{\text{bone}} = \arccos\left(\frac{\text{trace}(\mathbf{R}_{\text{bone}}(\hat{b})) - 1}{2}\right). \tag{5.14}$$

If θ_{bone} does not meet the set constraints, it is modified to the closer value of θ_{minBone} and $\theta_{\text{max}Bone}$. The constrained rotation is updated as $\acute{\theta}_{\text{bone}}$. Using $\acute{\theta}_{\text{bone}}$ and $\acute{\vec{r}}$, the new rotation around the bone, $\acute{\mathbf{R}}_{bone}(\hat{b})$, is computed using Equation 5.12.

The final four steps of Fig. 5.13, steps F–I, are the inverse of steps A–D in reverse order. They are executed to convert the constrained rotation back to the proper coordinate frame and representation. In step F, we compose $\acute{\mathbf{R}}_{\text{bone}}(\hat{b})$ and $\acute{\mathbf{R}}_{\text{joint}}(\hat{b})$ into one rotation to generate local rotation:

$$\acute{\mathbf{R}}^{\text{local}}_{\text{rel}}(\hat{b}) = \acute{\mathbf{R}}_{\text{bone}}(\hat{b})\acute{\mathbf{R}}_{\text{joint}}(\hat{b}). \tag{5.15}$$

In step G, $\acute{\mathbf{R}}_{\text{rel}}^{\text{local}}(\hat{b})$ which is in the constrained relative rotation in the local coordinate system is transformed to the world coordinate system:

$$\acute{\mathbf{R}}_{\text{rel}}^{\text{world}}(\hat{b}) = \mathbf{R}_{l\to w}(\hat{b})\acute{\mathbf{R}}_{\text{rel}}^{\text{local}}(\hat{b})\mathbf{R}_{l\to w}(\hat{b})^{-1}. \tag{5.16}$$

$\acute{\mathbf{R}}_{\text{rel}}^{\text{world}}(\hat{b})$ represents the constrained relative rotation of branch $\hat{b}$ with respect to its parent. In step H, $\acute{\mathbf{R}}_{\text{rel}}^{\text{world}}(\hat{b})$ is modified so that it defines the rotation of $\hat{b}$ with respect to the original pose of the branch in pose P_0:

$$\acute{\mathbf{R}}_{t}^{\text{new}}(\hat{b}) = \acute{\mathbf{R}}_{\text{rel}}^{\text{world}}(\hat{b})\mathbf{R}_{t}^{\text{curr}}(\hat{p}). \tag{5.17}$$

Finally, in step I, the rotation of the branch due to the pose $\mathbf{R}_{t}^{\text{curr}}(\hat{b})$ is removed to isolate $\acute{\mathbf{R}}(\hat{\mathbf{b}})$:

$$\acute{\mathbf{R}}(\hat{b}) = \acute{\mathbf{R}}_{t}^{\text{new}}(\hat{b})\mathbf{R}_{t}^{\text{curr}}(\hat{b})^{-1}. \tag{5.18}$$

Once $\acute{\mathbf{R}}(\hat{b})$ is calculated, the constrained rotation is returned to the calling branch fitting process in Fig 5.11.

5.4 Texturing the Template

In addition to capturing the realistic movement of the human subjects, we are also interested in capturing the colors of the human subject's clothes and skin. This is done by using the three color cameras located on the three system stations. Since the connectivity and number of vertices in the template does not change as it is deformed in each frame, each vertex should correspond to the same location on the human subject throughout the captured sequence. For example, the vertex located on the nose of the human subject should correspond to the nose in all frames of data.

We opt to texture the human subject primarily from three color images captured at the same time as the frame used to generate the template, specifically at time $t = 0$. These three images are aligned with the template in its original pose and expose a maximum amount of the subject's surface to the color cameras. With the three color cameras located on each structured-light station, the majority of the surface of the template can be textured as shown in Fig. 5.16a. To texture, we project the vertices of the template S_0 onto the color images and each vertex is assigned the color of the pixel it is projected onto.

Template regions that are not visible to the texture cameras at time $t = 0$ are shown in black in Fig. 5.16a. They make up regions such as the top of the head and top and bottom of the arms. We provide an estimate of the colors for these vertices by using the remaining color images captured during each target frame. Specifically, the deformed template generated for each target frame is textured using the color images captured at the same time as the target frame. Each deformed mesh is textured in the same manner as the original template. However, we do not expect as many of the vertices to be textured since most frames have more occlusions than the original template pose.

Once the deformed templates for all frames are textured, we analyze the color vector assigned to each vertex in each frame of the sequence. In particular, we focus on the vertices that are not directly assigned colors from color images of time $t = 0$. Even though some colors may be incorrect in individual frames, we can use the entire color vector for a vertex to estimate a representative

(a) (b) (c)

Figure 5.16: The template is shown textured: (a) using only color images from the template frame, (b) using median colors mapped to each template vertex, and (c) using a blend of textures (a) and (b).

color. To do this, the vector of colors for a single vertex is sorted in order of increasing intensity in the HSV color space. We use this color space rather than RGB so as to sort according to a single value per vertex. By selecting the color with the median intensity for each vertex, we obtain a reasonable estimate of the color for all vertices in the mesh. Fig. 5.16b shows the median color for each vertex in the template. It does not have as much detail as the direct projection of texture from images as in Fig. 5.16a, but it still represents the human subject's texture well.

The final texture for the template is generated by texturing the majority of the template with the color images from the template frame and by filling the empty regions with the median selected colors. The seams between the two regions are manually blended to improve the overall texture quality, as shown in Fig. 5.16c. In the event that a vertex of the mesh is never viewed by a color camera, the vertex will remain black. To correct these cases, the vertex color is copied from the nearest neighboring pixels with a computed texture. However, in the capture sequence presented in Section 5.5, each vertex is viewable by the observing cameras in at least a few frames, so we do not need to perform this texture copy process.

We have empirically found that directly applying the texture for frame t using the corresponding color images at time t results in unnatural, displeasing artifacts compared to the proposed texturing method. This is because the geometry in each frame is from a deformed version of the template; as such, any slight misalignment in the estimate of the pose for the template can result in misaligned texture. This can especially be true in regions where local surface deformations are not captured by the deformed template.

5.5 Results

We test our deformation method on a 50-second animation sequence consisting of 992 frames. Starting from the grayscale images that capture the human subject under structured light, a sequence of phase images are calculated for each camera. These phase images are unwrapped as in Section 3.4 and a sequence of point clouds are generated from each camera. The point clouds from each of the six geometry capturing cameras are filtered, meshed, and transformed into a single world coordinate frame. The result is a sequence of partial scans that we use to deform our template. During processing, we set parameters to iterate through all branches four times per frame. Additionally, the rigid transformation of each branch is estimated at most four times before proceeding to the other branches.

Fig. 5.17 shows the transformation of each branch over time in the local coordinate frame. The branches are labeled as in Fig. 5.5 and the rotation angles θ_X, θ_Z, and θ_{bone} of each branch are relative to the branch's orientation in pose P_0. Specifically, the angles for a bone b are derived from the rotation matrices $\acute{\mathbf{R}}_{\text{joint}}(b)$ and $\acute{\mathbf{R}}_{\text{bone}}(b)$ computed from the final pose P_t of each frame shown in Fig. 5.9. The sequence of captured geometry starts with the human subject in pose P_0, which is reflected in Fig. 5.17 as the θ_X, θ_Z, and θ_{bone} angles for each branch set to zero. Over time, the angles drift from this pose. The angles for the wrists, i.e. branches 7 and 11, are not illustrated because we did not allow for rotation around these joints in the final reconstruction. Limited geometry captured for the hands in most frames makes it difficult to constrain their movement consistently. As seen, there is a significant variation in the amount of rotation for each branch. The branches in the arms, specifically branches 5, 6, 9, and 10, exhibit the greatest variation in orientation. Sample poses shown in Fig. 5.18 illustrate the large range of motion that the arms exhibit during the the captured sequence.

The rotation of the root branch 0 around the bone axis θ_X is plotted in Fig. 5.19. Since the root branch does not have a parent branch, the rotation around the axis of the root bone is plotted relative to the world coordinate system. We show the pose of the template at approximate rotation angles of (a) 0, (b) $-\pi/2$, (c) π, and (d) $\pi/2$ in Fig. 5.20. As seen, the orientation of this angle controls the rotation of the template pose.

The effect of the joint constraints for the shoulder of the left arm, i.e. branch 5, is visualized in Fig. 5.22. As shown, the angles of each axis of rotation obey the specified constraints throughout the entire captured sequence. Although an angle may be constrained over multiple consecutive frames, the rotation of the angle resumes normally once the angle is within the constrained range. We find constraints to be most useful for branches not visible in a given pose. Additionally, some constraints are useful in ensuring that rotations occur at the correct joint. For example, the collar bones 4 and 8 in Fig. 5.5, are constrained in many frames to ensure that rotation of the arms are handled by the shoulder joints rather than the collarbones. This matches the anatomy of actual human skeletons which have a limited range of motion in collar bones compared to shoulders. We have empirically observed the majority of 992 to frames have at least one constrained joint. Fig. 5.21 shows the histogram of the number of constrained joints for all 992 frames. As seen, the majority of frames have only one or two joints constrained. This implies the iterative updates to the skeleton pose in successive frames are tracking to reasonable positions. Some of the constrained

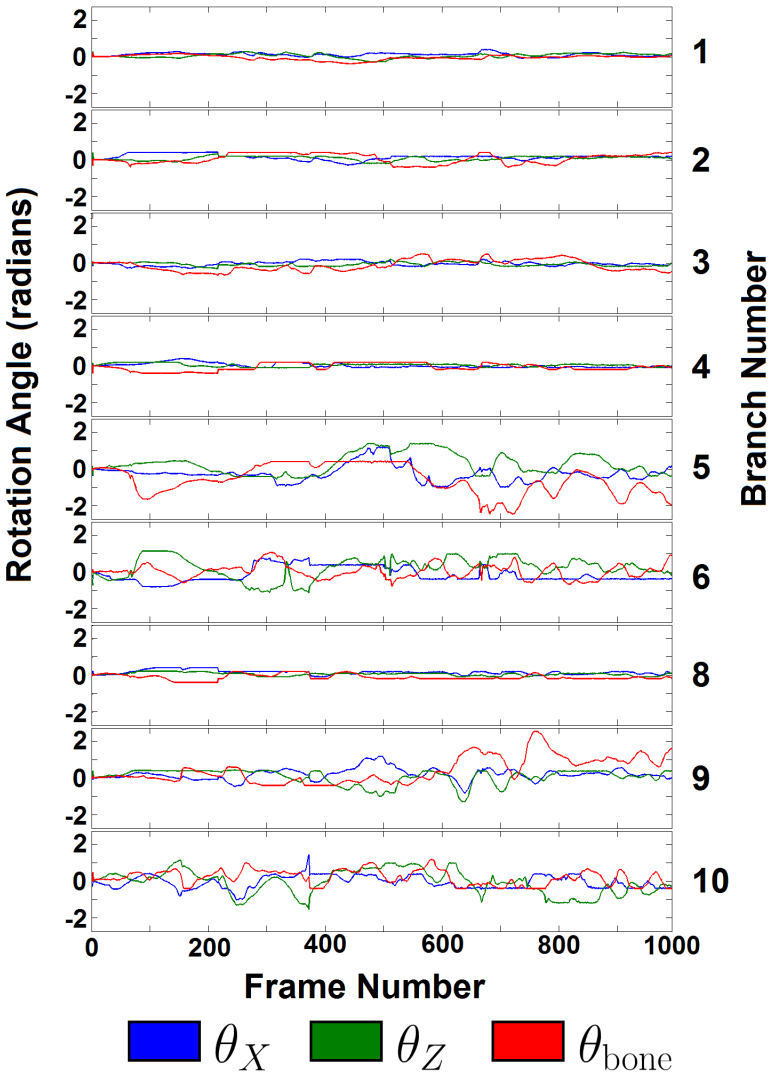


Figure 5.17: Rotation angles for branches in Fig. 5.5. For each branch, the sequence of values for θ_X, θ_Z, and θ_{bone} are plotted over the frame indices.

Figure 5.18: The template is deformed to fit the partial scans in a variety of poses.

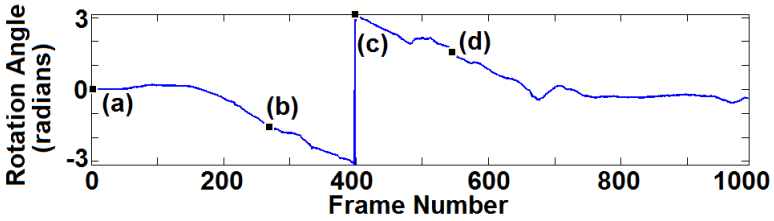
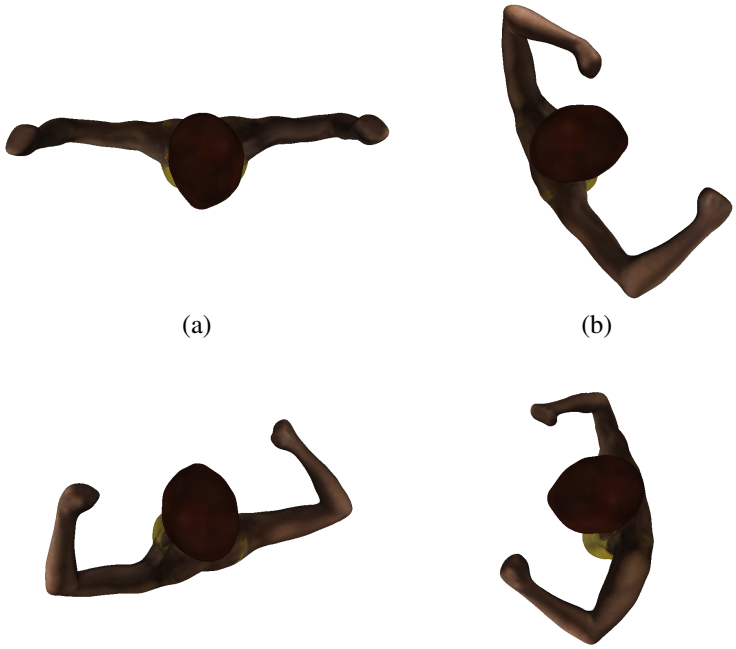


Figure 5.19: Rotation of root branch relative to world coordinate system. Rotation angles (a) 0, (b) θ_X, (c) θ_Z, and (d) θ_{bone} are illustrated in Fig. 5.20.

joints can be attributed to intentional restriction of some joints, as in the example with the collar bone above. We only consider the ten joints that were not constrained, excluding the two wrist joints.

As shown in Fig. 5.18, our deformation method is able to successfully capture the human subject in a variety of poses. Since we capture geometry from multiple sides of the human subject, the human subject is able to rotate and change the direction her body faces while still having the template successfully deform to match the geometry. The human subject must keep her fists closed during capture due to the limited resolution of the cameras in the system. The deformation processing of each frame takes less than two minutes with non-optimized prototype code. We expect this to be significantly improved by using the GPU.

The rendered view of the reconstructed sequence is shown in [3] and [4]. Specifically, [3] shows a side-by-side comparison of a video of the human subject and the same sequence reconstructed using our proposed approach. Also, [4] shows a 360-degree virtual view of the subject. As seen, our method is able to both accurately capture the geometry of the human subject and apply realistic texture. In [3], we can clearly observe the reconstructed geometry matching the motion of the human subject. In [4], the video shows that the human is accurately captured on all sides of the



(a) (b)

(c) (d)

Figure 5.20: Top view of deformed template for frames labeled in Fig. 5.19:(a) 0, (b) $-\pi/2$, (c) π, and (d) $\pi/2$.

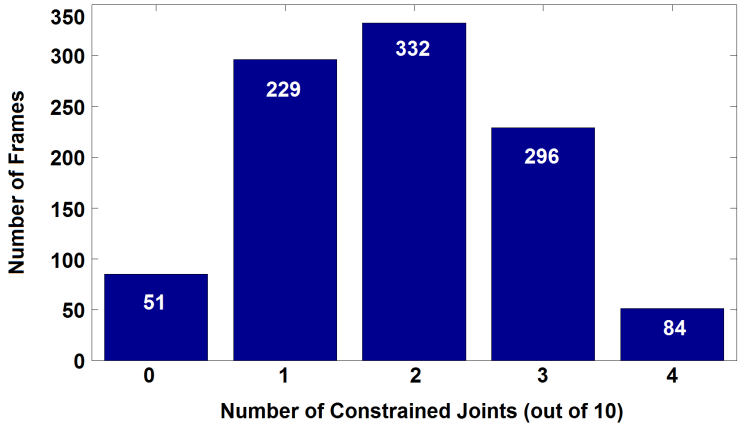


Figure 5.21: Histogram of the number of constrained joints for all 992 frames.

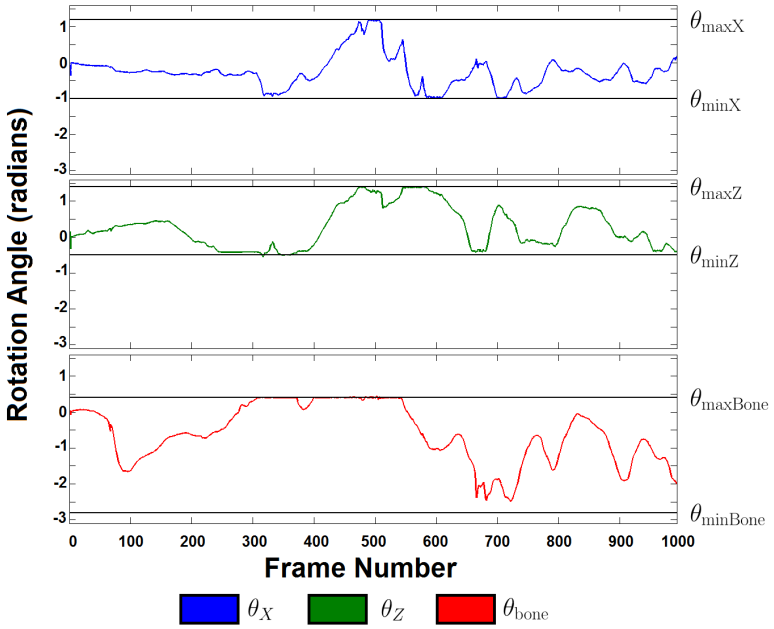


Figure 5.22: Variation of θ_X, θ_Z, and θ_{bone} for branch 5 over captured sequence with angle limits plotted for each axis of rotation.

human subject.

5.6 Conclusions

We have presented a multi-view structured-light system capable of performing markerless motion capture for human subjects. With rich captured 3D data rather than 2D video streams, we are able to easily deform a template to match partial scans of the human subject. Unlike existing methods, we can easily create templates directly from our system without having to rely on a special scanning process or needing high quality laser scans.

In future work, we plan to explore improve the quality and level of details of geometry resulting from the system. This could be done by testing other configurations of structured-light patterns or changing projector and camera hardware for models with higher speed and improved resolutions. This would allow us to also integrate local deformation methods to capture the subtle details of the human subject's geometry [63]. Additionally, we would like to decrease processing time by offloading some computing onto the GPU and by streamlining the 3D geometry generation and deformation steps.

Chapter 6

Conclusion and Future Work

In this dissertation, we presented a multi-view structured-light system for markerless motion capture of human subjects. This is the first interference free system consisting of multiple structured-light stations operating. Our design ensures strict timing between all cameras and projectors. With precise control over device timing, we can ensure that only a single projector illuminates the human subject at any time, thus preventing interference between individual structured light stations.

Additionally, our system is able to directly generate a watertight template of human subjects using the partial scans resulting from multiple structured-light stations. Many markerless motion capture systems deform a template of the human subject over time to capture human motion. They generally require additional 3D scanning hardware to capture the initial template shape of the human subject. In our system, partial scans of the human subject from each structured-light station are merged into a single coordinate frame. If the subject is positioned in a pose that has limited occluded surfaces from the perspective of the structured light stations, a watertight mesh can be fit over the partially observed subject. Our template provides an accurate representation of the shape and texture of the human subject.

Finally, a distinct advantage of our system over other markerless systems is in directly measurement of the shape of the human subject's surface. Specifically, existing multi-camera markerless motion capture systems do not directly capture the surface shape of the human subject. They often resort to estimating the human subject's pose by using visual hulls. Direct capture the subject's surface allows for accurate correspondences between the template and surface scans, leading to a deformation method which consistently and accurately captures human motion.

In this dissertation, we presented the architecture and algorithms for a human motion capture using multi-view structured light. The methods and algorithms presented within this dissertation would not have been possible without the full development of the system hardware and software that enabled pattern projection and image capture. The synchronization of the individual structured-light stations gave us the ability to bring multiple structured-light stations together in a single system to capture dynamic human motion.

In addition to the system harware, our phase unwrapping methods allow us to generate dynamic reconstructions of a human performer. Whereas most phase unwrapping methods for sinusoidal structured-light systems attempt to unwrap phase images individually, our methods for unwrapping

over time and by using multiple viewpoints provide final reconstructed geometry that is temporally consistent. The algorithms we present in phase unwrapping enable each station in our structured-light system to accurately and consistently capture geometry.

Accurately combining the multiple views of our captured geometry was made possible through our multi-camera-projector calibration method. With our improved calibration approach, we were able to simultaneously calibrate all cameras and projectors. In contrast to other approaches that calibrate projectors individually, separate from system cameras, our method is able to calibrate all devices together which better distributes any residual error across all devices. The end result is a calibration method that precisely aligns the dynamic geometry we are able to capture from each one of our stations.

Finally, we are able to arrive at a final hole-free reconstruction of a human subject's performance. By generating a template directly from our system, we are able to fill in regions of the human subject's surface that cannot be viewed by the devices of our system. We present an approach to deform a template over time to fit our partial observations of the scene. By using color images captured from our stations, we texture our template to arrive at an accurate human motion capture result.

There are a few directions of work that would be worth pursuing to improve the results of our system even further. First, improving the resolution of all cameras and projectors in the system would allow us to capture more detailed surfaces of our human performer. Additionally, it would be interesting to see how much further the accuracy of the surface reconstructions could be improved through a combination of modifying our structured-light patterns and changing the placement of geometry capture cameras at each station. With sinusoidal patterns that use a greater number of periods across the pattern, we might be able to improve the level of detail in the surface reconstructions that are captured. However, the benefit would need to be balanced with the potential for less tolerance for motion in the scene. Additionally, greater detail in the captured geometry could be merged with the our template deformation methods to capture local surface deformation rather than only high level deformation from skeleton manipulation. Finally, exploring approaches that combine stereo reconstruction techniques while our projectors illuminate the scene could improve our systems performance in difficult scene regions such as portions of the scene that are dark or highly textured. Merging stereo techniques with structured light could offer the best combination when trying to capture the detailed geometry of a dynamic human subject.

Bibliography

[1] URL: http://www-video.eecs.berkeley.edu/research/4D_SL/viewpointConsistentUnwrap.wmv.

[2] URL: http://www-video.eecs.berkeley.edu/research/4D_SL/temporalConsistentUnwrap.wmv.

[3] URL: http://www-video.eecs.berkeley.edu/research/4D_SL/reference.mp4.

[4] URL: http://www-video.eecs.berkeley.edu/research/4D_SL/rotate.mp4.

[5] H. S. Abdul-Rahman, M. Arevalillo-Herráez, M. Gdeisat, D. Burton, M. Lalor, F. Lilley, C. Moore, D. Sheltraw, and M. Qudeisat. "Robust three-dimensional best-path phase-unwrapping algorithm that avoids singularity loops". In: *Appl. Opt.* 48.23 (Aug. 2009), pp. 4582–4596.

[6] H. S. Abdul-Rahman, M. Gdeisat, D. Burton, and M. Lalor. "Fast three-dimensional phase-unwrapping algorithm based on sorting by reliability following a non-continuous path". In: *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*. Ed. by W. Osten, C. Gorecki, & E. L. Novak. Vol. 5856. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. June 2005, pp. 32–40.

[7] H. S. Abdul-Rahman, M. A. Gdeisat, D. R. Burton, M. J. Lalor, F. Lilley, and A. Abid. "Three-dimensional Fourier Fringe Analysis". In: *Optics and Lasers in Engineering* 46.6 (2008), pp. 446–455. ISSN: 0143-8166.

[8] H. S. Abdul-Rahman, Munther A. Gdeisat, David R. Burton, Michael J. Lalor, Francis Lilley, and Christopher J. Moore. "Fast and robust three-dimensional best path phase unwrapping algorithm". In: *Appl. Opt.* 46.26 (Sept. 2007), pp. 6623–6635.

[9] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. "Performance Capture from Sparse Multi-view Video". In: *ACM Trans. Graph.* 27.3 (Aug. 2008), 98:1–98:10. ISSN: 0730-0301.

[10] M. Arevalillo-Herráez, D. R. Burton, and M. J. Lalor. "Clustering-based robust three-dimensional phase unwrapping algorithm". In: *Appl. Opt.* 49.10 (Apr. 2010), pp. 1780–1788.

[11] M. Arevalillo-Herráez, M. A. Gdeisat, and D. R. Burton. "Hybrid robust and fast algorithm for three-dimensional phase unwrapping". In: *Appl. Opt.* 48.32 (Nov. 2009), pp. 6313–6323.

[12] S. Audet and M. Okutomi. “A user-friendly method to geometrically calibrate projector-camera systems”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. June 2009, pp. 47–54.

[13] L. Ballan and G. M. Cortelazzo. “Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes”. In: *3D Data Procesing, Visualization, and Transmission (3DPVT), Atlanta, GA, USA* 37 (2008).

[14] D. Bergmann. *New approach for automatic surface reconstruction with coded light*. Ed. by Toni F. Schenk. San Diego, CA, USA, 1995.

[15] Paul J. Besl and Neil D. McKay. “Method for registration of 3-D shapes”. In: *Proc. SPIE*. Vol. 1611. 1992, pp. 586–606.

[16] F. Blais. “Review of 20 years of range sensor development”. In: *Journal of Electronic Imaging* 13.1 (2004), pp. 63–76.

[17] D. J. Bone. “Fourier fringe analysis: The two-dimensional phase unwrapping problem”. In: *Appl. Opt.* 30.25 (Sept. 1991), pp. 3627–3632.

[18] J.-Y. Bouguet. *Camera Calibration Toolbox for MATLAB*. 2004. URL: http://www.vision.caltech.edu/bouguetj/calib_doc/.

[19] X. Chen, J. Xi, Y. Jin, and J. Sun. “Accurate calibration for a camera-projector measurement system based on structured light projection”. In: *Optics and Lasers in Engineering* 47.3–4 (2009), pp. 310–319. ISSN: 0143-8166.

[20] M. Constantini, F. Malvarosa, and F. Minati. “A general formulation for robust and efficient integration of finite differences and phase unwrapping on sparse multidimensional domains”. In: *Proc. ESA Fringe Workshop*. Frascati, Italy, Nov. 2009.

[21] S. Corazza, L. Mndermann, A.M. Chaudhari, T. Demattio, C. Cobelli, and T. P. Andriacchi. “A Markerless Motion Capture System to Study Musculoskeletal Biomechanics: Visual Hull and Simulated Annealing Approach”. In: *Annals of Biomedical Engineering* 34.6 (2006), pp. 1019–1029. ISSN: 0090-6964.

[22] Stefano Corazza, Lars Mndermann, Emiliano Gambaretto, Giancarlo Ferrigno, and ThomasP. Andriacchi. “Markerless Motion Capture through Visual Hull, Articulated ICP and Subject Specific Model Generation”. In: *International Journal of Computer Vision* 87.1-2 (2010), pp. 156–169. ISSN: 0920-5691.

[23] M. Costantini, F. Malvarosa, and F. Minati. “A novel approach for redundant integration of finite differences and phase unwrapping on a sparse multidimensional domain”. In: *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*. July 2010, pp. 1565 –1568.

[24] M. Costantini, F. Malvarosa, F. Minati, L. Pietranera, and G. Milillo. “A three-dimensional phase unwrapping algorithm for processing of multitemporal SAR interferometric measurements”. In: *Geoscience and Remote Sensing Symposium, 2002. IGARSS ’02. 2002 IEEE International*. Vol. 3. June 2002, pp. 1741–1743.

[25] C. Cruz-Neira, D. J Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart. "The CAVE: Audio visual experience automatic virtual environment". In: *Commun. ACM* 35.6 (June 1992), pp. 64–72. ISSN: 0001-0782.

[26] R. Cusack and N. Papadakis. "New Robust 3-D Phase Unwrapping Algorithms: Application to Magnetic Field Mapping and Undistorting Echoplanar Images". In: *NeuroImage* 16.3, Part A (2002), pp. 754–764. ISSN: 1053-8119.

[27] J. Davis, D. Nehab, R. Ramamoorthi, and S. Rusinkiewicz. "Spacetime stereo: A unifying framework for depth from triangulation". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.2 (Feb. 2005), pp. 296–302. ISSN: 0162-8828.

[28] M. Dou, H. Fuchs, and J.-M. Frahm. "Scanning and tracking dynamic objects with commodity depth cameras". In: *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*. Oct. 2013, pp. 99–106.

[29] G. Falcao, N. Hurtos, J. Massich, and D. Fofi. *Projector-Camera Calibration Toolbox*. 2009. URL: http://code.google.com/p/procamcalib.

[30] P. Felzenszwalb and D. Huttenlocher. "Efficient Belief Propagation for Early Vision". In: *International Journal of Computer Vision* 70 (1 2006), pp. 41–54. ISSN: 0920-5691.

[31] B. J. Frey, R. Koetter, and N. Petrovic. "Very Loopy Belief Propagation for Unwrapping Phase Images". In: *Advances in Neural Information Processing System*. Cambridge, MA: MIT Press, 2001, pp. 737–743.

[32] R. Furukawa, R. Sagawa, A. Delaunoy, and H. Kawasaki. "Multiview projectors/cameras system for 3D reconstruction of dynamic scenes". In: *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. Nov. 2011, pp. 1602–1609.

[33] R. Furukawa, R. Sagawa, H. Kawasaki, K. Sakashita, Y. Yagi, and N. Asada. "One-shot Entire Shape Acquisition Method Using Multiple Projectors and Cameras". In: *Image and Video Technology (PSIVT), 2010 Fourth Pacific-Rim Symposium on*. Nov. 2010, pp. 107–114.

[34] Y. Furukawa and J. Ponce. "Accurate Camera Calibration from Multi-View Stereo and Bundle Adjustment". In: *International Journal of Computer Vision* 84 (3 2009), pp. 257–268. ISSN: 0920-5691.

[35] J. Gall, C. Stoll, E. de Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel. "Motion capture using joint skeleton tracking and surface estimation". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. June 2009, pp. 1746–1753.

[36] R. R. Garcia and A. Zakhor. "Consistent Stereo-Assisted Absolute Phase Unwrapping Methods for Structured Light Systems". In: *Selected Topics in Signal Processing, IEEE Journal of* 6.5 (Sept. 2012), pp. 411–424. ISSN: 1932-4553.

[37] R. R. Garcia and A. Zakhor. "Geometric calibration for a multi-camera-projector system". In: *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*. Jan. 2013, pp. 467–474.

[38] R. R. Garcia and A. Zakhor. “Projector domain phase unwrapping in a structured light system with stereo cameras”. In: *3DTV Conference: The True Vision — Capture, Transmission and Display of 3D Video (3DTV-CON), 2011*. May 2011, pp. 1–4.

[39] R. R. Garcia and A. Zakhor. “Temporally-Consistent Phase Unwrapping for a Stereo-Assisted Structured Light System”. In: *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*. May 2011, pp. 389–396.

[40] M. Gdeisat, M. Arevalillo-Herráez, D. Burton, and F. Lilley. “Three-dimensional phase unwrapping using the Hungarian algorithm”. In: *Opt. Lett.* 34.19 (Oct. 2009), pp. 2994–2996.

[41] R. Gens. “Two-dimensional phase unwrapping for radar interferometry: Developments and new challenges”. In: *International Journal of Remote Sensing* 24.4 (2003), pp. 703–710.

[42] D. C. Ghiglia and M. D. Pritt. *Two-dimensional phase unwrapping: Theory, algorithms, and software*. Wiley-Interscience publication. 1998. ISBN: 978-0-471-24935-1.

[43] R. M. Goldstein, H. A. Zebker, and C. L. Werner. “Satellite radar interferometry — Two-dimensional phase unwrapping”. In: *Radio Science* 23.4 (1988), pp. 713–720.

[44] S. S. Gorthi and P. Rastogi. “Fringe projection techniques: Whither we are?” In: *Optics and Lasers in Engineering* 48 (Feb. 2010), pp. 133–140.

[45] A. Griesser and L. Van Gool. “Automatic Interactive Calibration of Multi-Projector-Camera Systems”. In: *Computer Vision and Pattern Recognition Workshop. CVPRW ’06. Conference on*. June 2006.

[46] A. Griesser, T.P. Koninckx, and L. Van Gool. “Adaptive real-time 3D acquisition and contour tracking within a multiple structured light system”. In: *Computer Graphics and Applications. Proceedings. 12th Pacific Conference on*. Oct. 2004, pp. 361–370.

[47] O. Hall-Holt and S. Rusinkiewicz. “Stripe Boundary Codes for Real-Time Structured-Light Range Scanning of Moving Objects”. In: *Computer Vision, IEEE International Conference on* 2 (2001), pp. 359–366.

[48] X. Han and P. Huang. “Combined stereovision and phase shifting method: A new approach for 3D shape measurement”. In: *Optical Measurement Systems for Industrial Inspection VI* 7389.1 (2009). Ed. by Peter H. Lehmann.

[49] A. Hooper and H. A. Zebker. “Phase unwrapping in three dimensions with application to InSAR time series”. In: *J. Opt. Soc. Am. A* 24.9 (Sept. 2007), pp. 2737–2747.

[50] P. S. Huang, C. Zhang, and F.-P. Chiang. “High-speed 3-D shape measurement based on digital fringe projection”. In: *Optical Engineering* 42.1 (2003), pp. 163–168.

[51] P. S. Huang and S. Zhang. “Fast three-step phase-shifting algorithm”. In: *Appl. Opt.* 45.21 (July 2006), pp. 5086–5091.

[52] J. M. Huntley. “Three-Dimensional Noise-Immune Phase Unwrapping Algorithm”. In: *Appl. Opt.* 40.23 (Aug. 2001), pp. 3901–3908.

[53] J. M. Huntley and H. Saldner. "Temporal phase-unwrapping algorithm for automated interferogram analysis". In: *Appl. Opt.* 32.17 (June 1993), pp. 3047–3052.

[54] M. Jenkinson. "Fast, automated, N-dimensional phase-unwrapping algorithm". In: *Magnetic Resonance in Medicine* 49.1 (2003), pp. 193–197. ISSN: 1522-2594.

[55] T. R. Judge and P. J. Bryanston-Cross. "A review of phase unwrapping techniques in fringe analysis". In: *Optics and Lasers in Engineering* 21.4 (1994), pp. 199–239. ISSN: 0143-8166.

[56] L. Kavan, S. Collins, Jiří Žára, and Carol O'Sullivan. "Skinning with Dual Quaternions". In: *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*. I3D '07. Seattle, Washington: ACM, 2007, pp. 39–46. ISBN: 978-1-59593-628-8.

[57] H. Kawasaki, R. Furukawa, R. Sagawa, and Y. Yagi. "Dynamic scene shape reconstruction using a single structured light pattern". In: *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on* (2008), pp. 1–8.

[58] M. Kazhdan, M. Bolitho, and H. Hoppe. "Poisson surface reconstruction". In: *Proceedings of the Fourth Eurographics Symposium on Geometry processing*. 2006.

[59] S. Kobayashi, F. Sakaue, and J. Sato. "Multiple View Geometry of Projector-Camera Systems from Virtual Mutual Projection". In: Lecture Notes in Computer Science (2009). Ed. by Toshikazu Wada, Fay Huang, and Stephen Lin.

[60] Don Koks. *Explorations in mathematical physics*. Springer, 2006.

[61] J. Langley and Q. Zhao. "A model-based 3D phase unwrapping algorithm using Gegenbauer polynomials". In: *Physics in Medicine and Biology* 54.17 (2009), pp. 5237–5252.

[62] H. Li, L. Luo, D. Vlasic, P. Peers, J. Popović, M. Pauly, and S. Rusinkiewicz. "Temporally Coherent Completion of Dynamic Shapes". In: *ACM Trans. Graph.* 31.1 (Feb. 2012), 2:1–2:11. ISSN: 0730-0301.

[63] Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. "Robust Single-View Geometry And Motion Reconstruction". In: *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009)* 28.5 (2009).

[64] W.-S. Li and X.-Y. Su. "Phase unwrapping algorithm based on phase fitting reliability in structured light projection". In: *Optical Engineering* 41.6 (2002), pp. 1365–1372.

[65] Z. Li, Y. Shi, C. Wang, and Y. Wang. "Accurate calibration method for a structured light system". In: *Optical Engineering* 47.5, 053604 (2008).

[66] K. Liu, Y. Wang, D. L. Lau, Q. Hao, and L. G. Hassebrook. "Dual-frequency pattern scheme for high-speed 3-D shape measurement". In: *Opt. Express* 18.5 (Mar. 2010), pp. 5229–5244.

[67] M. I. A. Lourakis and A. A. Argyros. "SBA: A Software Package for Generic Sparse Bundle Adjustment". In: *ACM Trans. Math. Software* 36.1 (2009), pp. 1–30.

[68] O. Marklund, J. M. Huntley, and R. Cusack. "Robust unwrapping algorithm for three-dimensional phase volumes of arbitrary shape containing knotted phase singularity loops". In: *Optical Engineering* 46.8, 085601 (2007), pp. 08560–1–08560–13.

[69] N. J. Mitra, S. Flöry, M. Ovsjanikov, N. Gelfand, L. Guibas, and H. Pottmann. "Dynamic Geometry Registration". In: *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*. SGP '07. Barcelona, Spain: Eurographics Association, 2007, pp. 173–182. ISBN: 978-3-905673-46-3.

[70] T. B. Moeslund and E. Granum. "A Survey of Computer Vision-Based Human Motion Capture". In: *Computer Vision and Image Understanding* 81.3 (2001), pp. 231–268. ISSN: 1077-3142.

[71] L. Mundermann, S. Corazza, and T.P. Andriacchi. "Accurately measuring human movement using articulated ICP with soft-joint constraints and a repository of articulated models". In: *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*. June 2007, pp. 1–6.

[72] K. Murphy, Y. Weiss, and M. I. Jordan. "Loopy belief propagation for approximate inference: An empirical study". In: *Proceedings of Uncertainty in AI*. Vol. 9. 1999, pp. 467–475.

[73] S. I. Park and J. K. Hodgins. "Capturing and Animating Skin Deformation in Human Motion". In: *ACM SIGGRAPH 2006 Papers*. SIGGRAPH '06. Boston, Massachusetts: ACM, 2006, pp. 881–889. ISBN: 1-59593-364-6.

[74] T. Popa, I. South-Dickinson, D. Bradley, A. Sheffer, and W. Heidrich. "Globally Consistent Space-Time Reconstruction". In: *Computer Graphics Forum* 29.5 (2010), pp. 1633–1642. ISSN: 1467-8659.

[75] R. Raskar, M.S. Brown, Ruigang Yang, Wei-Chao Chen, G. Welch, H. Towles, B. Scales, and H. Fuchs. "Multi-projector displays using camera-based registration". In: *Visualization '99. Proceedings*. Oct. 1999, pp. 161–522.

[76] N. Saad and S. Peled. "Easy 3D phase unwrapping". In: *Proc Int Soc Magn Reson Med* 13 (2005), pp. 2251–2251.

[77] J. Salvi, J. Pags, and J. Batlle. "Pattern codification strategies in structured light systems". In: *Pattern Recognition* 37.4 (2004), pp. 827–849. ISSN: 0031-3203.

[78] D. Scharstein and R. Szeliski. "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms". In: *International Journal of Computer Vision* 47 (1 2002), pp. 7–42. ISSN: 0920-5691.

[79] A. P. Shanker and H. Zebker. "Edgelist phase unwrapping algorithm for time series InSAR analysis". In: *J. Opt. Soc. Am. A* 27.3 (Mar. 2010), pp. 605–612.

[80] X. Su and Q. Zhang. "Phase unwrapping in the dynamic 3D measurement". In: *AIP Conference Proceedings* 1236.1 (2010), pp. 467–471.

[81] R. W. Sumner, J. Schmid, and M. Pauly. "Embedded Deformation for Shape Manipulation". In: *ACM SIGGRAPH 2007 Papers*. SIGGRAPH '07. San Diego, California: ACM, 2007.

[82] J. Süßmuth, M. Winter, and G. Greiner. "Reconstructing Animated Meshes from Time-Varying Point Clouds". In: *Computer Graphics Forum* 27.5 (2008), pp. 1469–1476. ISSN: 1467-8659.

[83] T. Svoboda, D. Martinec, and T. Pajdla. "A convenient multicamera self-calibration for virtual environments". In: *Presence: Teleoper. Virtual Environ.* 14.4 (Aug. 2005), pp. 407–422.

[84] J.-P. Tardif, S. Roy, and M. Trudeau. "Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction". In: *3-D Digital Imaging and Modeling. 3DIM 2003. Proceedings. Fourth International Conference on*. Oct. 2003, pp. 217–224.

[85] C. Theobalt, N. Ahmed, H. Lensch, M. Magnor, and H.-P. Seidel. "Seeing People in Different Light-Joint Shape, Motion, and Reflectance Capture". In: *Visualization and Computer Graphics, IEEE Transactions on* 13.4 (July 2007), pp. 663–674. ISSN: 1077-2626.

[86] T. Ueshiba and F. Tomita. "Plane-based calibration algorithm for multi-camera systems via factorization of homography matrices". In: *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. Vol. 2. Oct. 2003, pp. 966–973.

[87] F. J. van Leijen, R. F. Hanssen, P. S. Marinkovic, and B. M. Kampes. "Spatio-temporal phase unwrapping using integer least-squares". In: *Fringe 2005 Workshop*. Vol. 610. ESA Special Publication. Feb. 2006.

[88] D. Vlasic, I. Baran, W. Matusik, and J. Popović. "Articulated Mesh Animation from Multi-view Silhouettes". In: *ACM SIGGRAPH 2008 Papers*. SIGGRAPH '08. Los Angeles, California: ACM, 2008, 97:1–97:9. ISBN: 978-1-4503-0112-1.

[89] D. Vlasic, P. Peers, I. Baran, P. Debevec, J. Popović, S. Rusinkiewicz, and W. Matusik. "Dynamic Shape Capture Using Multi-view Photometric Stereo". In: *ACM SIGGRAPH Asia 2009 Papers*. SIGGRAPH Asia '09. Yokohama, Japan: ACM, 2009, 174:1–174:11. ISBN: 978-1-60558-858-2.

[90] T. Weise, B. Leibe, and L. Van Gool. "Fast 3D Scanning with Automatic Motion Compensation". In: *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*. June 2007, pp. 1–8.

[91] T. Weise, B. Leibe, and L. Van Gool. "Fast 3D Scanning with Automatic Motion Compensation". In: *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*. June 2007, pp. 1–8.

[92] S. Yamazaki, M. Mochimaru, and T. Kanade. "Simultaneous self-calibration of a projector and a camera using structured light". In: *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*. June 2011, pp. 60–67.

[93] R. Yang, D. Gotz, J. Hensley, H. Towles, and M.S. Brown. "PixelFlex: A reconfigurable multi-projector display system". In: *Visualization, 2001. Proceedings*. Oct. 2001, pp. 167–554.

[94] M. Young, E. Beeson, J. Davis, S. Rusinkiewicz, and R. Ramamoorthi. "Viewpoint-Coded Structured Light". In: *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*. June 2007, pp. 1–8.

[95] L. Zhang, B. Curless, and S. M. Seitz. “Spacetime stereo: Shape recovery for dynamic scenes”. In: *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. Vol. 2. June 2003, II –367–74 vol.2.

[96] S. Zhang and P. Huang. “High-Resolution, Real-time 3D Shape Acquisition”. In: *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW ’04. Conference on*. June 2004, pp. 28–28.

[97] S. Zhang and P. S. Huang. “Novel method for structured light system calibration”. In: *Optical Engineering* 45.8 (2006).

[98] S. Zhang, X. Li, and S.-T. Yau. “Multilevel quality-guided phase unwrapping algorithm for real-time three-dimensional shape reconstruction”. In: *Appl. Opt.* 46.1 (Jan. 2007), pp. 50–57.

[99] S. Zhang and S.-T. Yau. “Absolute phase-assisted three-dimensional data registration for a dual-camera structured light system”. In: *Appl. Opt.* 47.17 (June 2008), pp. 3134–3142.