Markerless 3D Reconstruction of Temporally Deforming Surfaces from Multiple Cameras

by

Derek Bradley

M.C.S., Carleton University, 2005

B.C.S., Carleton University, 2003

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

The Faculty of Graduate Studies

(Computer Science)

The University Of British Columbia
(Vancouver)

June, 2010

Abstract

3D content generation is a major challenge in computer graphics, and generating realistic objects with time-varying shape and motion is a big part of creating realistic virtual environments. An attractive approach for generating content is to acquire the shape of real objects using computer vision techniques. Capturing the real world avoids the complexity of mathematical simulation as well as the time-consuming and difficult process of creating content manually.

Traditional acquisition methods, such as range scanning, work well for static objects. However, temporally deformable objects like garments and human faces pose greater challenges. Here, the capture process must reconstruct both the time-varying shape as well as the motion of the object. Additionally, per-frame texture maps should be acquired for realistic rendering. Previous methods contain various limitations, such as the use of structured illumination patterns, or hand-placed markers on the surface in order to guide the reconstruction. Placing markers on garments limits the types of garments that can be captured. Both markers and structured light on human faces limit the ability to capture high-quality per-frame texture maps, which are essential for reconstructing performance-specific details such as sweating, wrinkles and blood flow.

In this thesis we propose new, entirely passive techniques for advancing the state-of-the-art in 3D reconstruction of temporally deformable surfaces. We reconstruct high-resolution, temporally compatible geometry using multiple video cameras, without the need for either markers or structured light patterns.

This thesis contains several contributions, both in computer graphics as well as computer vision. We start with new methods for multi-camera calibration and synchronization, which allow us to employ inexpensive consumer video cameras for reconstruction. We also present a novel technique for multi-view stereo reconstruction, capable of building high-resolution geometric meshes from a number of images. Finally, we combine these ideas and provide two inexpensive solutions for high-resolution 3D reconstruction of temporally deforming surfaces without the need for markers. The first is an application for capturing the shape and motion of garments, and the second is a new method for acquiring facial performances of actors. These new techniques have the potential for high impact in the film and video game industries.

Contents

List of Tables

List of Figures

Acknowledgements

First and foremost I would like to thank my supervisor, Dr. Wolfgang Heidrich, for his guidance throughout my PhD degree, and for helping me to improve my research skills, preparing me for my next endeavour as a post-doctoral researcher. Thank you Wolfgang.

Second, I would like to thank Dr. Alla Sheffer for also taking a keen interest in my research, working very closely with me and helping me to write higher-quality papers. Thanks also to Dr. David Lowe for serving as the third member of my supervisory committee, providing valuable feedback on my thesis proposal and initial dissertation.

Thanks to my thesis examiners, Dr. Michiel van de Panne, Dr. Tim Salcudean, and Dr. Marc Pollefeys. Your comments during and after my oral defence helped me to improve the final dissertation.

I would like to thank my parents, my sister and the rest of my family for their support throughout my academic endeavours.

Finally, thanks to all my friends in Vancouver who shared in the fun times and helped me through the more challenging times, making the past five years at UBC an experience I'll never forget.

Chapter 1

Introduction

Computer generated graphical content is ubiquitous in the entertainment industry, as realistic virtual environments filled with computer generated objects are becoming increasingly popular in modern-day films and video games. This content usually consists of polygonal models, including both the shape of objects and their motion or deformation over time. Creation of such content is a wide area of study, covering topics such as physical simulation, real world capture, and tools for manual interactive generation. In the research community, current work strives to provide new techniques that improve both the quality of the content as well as the speed at which it can be generated, while often aiming for methods that can be automated to decrease the demand on the artists who create the 3D models.

With many different approaches available for content generation, the choice of which method to employ usually depends on the type of content desired. For example, the simulation approach aims at mathematically reconstructing physical phenomena, such as the fluid motion of water or the collision and contact behaviour between different surfaces. Capture methods try to reconstruct the shape and sometimes the motion of real-world objects, creating a virtual "copy" of the object. These techniques include the use of laser-scanners for static reconstructions, motion capture suits for piece-wise rigid human motion, and most recently, image-based approaches that aim to reconstruct both the motion and the shape of static or deformable objects. Finally, there are many commercial tools for manual content generation which are very commonly used in the film industry, although they typically require an artist with a lot of training and skill.

In this thesis, we propose new techniques in the area of real-world capture, specifically using video cameras. As a data-driven approach, we believe that image-based capture has certain benefits over other methods. For example, assume we wish to generate the realistic deformation of a sheet of silk cloth under a specific motion. This could be accomplished using simulation, however it becomes a problem of specific parameter tuning to make the result behave exactly like silk fabric should. With motion capture using videos, the deformation of a real silk sheet can be reconstructed precisely, automatically reflecting the behaviour of the silk fabric. Furthermore, if the content we wish to generate is even more specific, such as the shape and deformation of an actor's face during a specific performance, then 3D acquisition is the only feasible way to accomplish this. This sort of content is widely used in films and video games. An additional benefit of video-based capture is that it allows for simple post-production and augmentation of films, allowing artists to change the appearance of items in a video after it is recorded. Once the shape of objects are acquired, these objects are already geometrically aligned with the video cameras. Any edits to the appearance of the object can then be trivially rendered from the perspective of the camera and blended with the original video. Typical appearance changes that artists desire include texture, surface reflectance, or even the geometry itself.

While video-based capture techniques have been improving in recent years, many objects remain a challenge to reconstruct, such as temporally deforming surfaces like garments and human faces. This is because the capture process must recover both the time-varying shape of the object as well as its motion over time, usually encoded as a dense point-to-point mapping of the surface between all time steps. Additionally, per-frame texture maps should be acquired for realistically rendering the virtual object. Deforming objects tend to yield incomplete reconstructions due to occlusion, and their deformation makes it challenging to establish the inter-frame surface mapping. While some research has been performed for video-based capture in this domain, many challenges still remain. Specifically, most state-of-the-art techniques require the use of structured illumination patterns projected onto the object, or hand-placed markers on the surface in order to guide the reconstruction. Covering an object with markers can be a tedious process, and the markers may affect how the surface deforms. In some cases, such as certain cloth fabrics, it may not even be possible to use markers. If the markers must be ironed onto the cloth surface then this would limit the different types of fabrics that could be captured, as many fabrics cannot be ironed. Additionally, both markers and structured light can limit the ability to acquire high-quality per-frame texture maps of the surface. These texture maps are essential for some applications, such as face reconstruction, where texture maps contain performance-specific details of the actor's face. In this thesis we introduce new methods for capturing deforming surfaces without the need for markers or structured illumination, specifically focusing on garments and human facial performances. With our new *marker-free* reconstruction techniques, we aim to advance the state-of-the-art in 3D reconstruction of temporally deforming objects from video cameras.

The process of shape and motion capture from video consists of many different steps, which we classify in a hierarchy of three levels. At the lowest level there are hardware issues such as the physical acquisition setup, camera synchronization and calibration. At the middle level there are software tools, such as multi-view stereo, required for the geometric reconstruction of objects given a set of calibrated camera images. Finally, specific combinations of these hardware components and tools can be used to form higher level applications for time-varying, object-specific capture (see Figure 1.1).

In this thesis we explore all three hierarchical levels, covering topics in both computer graphics as well as computer vision. The main contributions include new methods for multi-camera synchronization and calibration, allowing the use of inexpensive consumer video cameras in novel physical acquisition setups. We also present a new technique for multi-view stereo reconstruction, capable of building high-resolution geometric meshes from a number of images. Finally, using these tools we provide two inexpensive solutions for high-resolution 3D reconstruction of temporally deforming surfaces without the need for markers or structured light. The first is an application for capturing the shape and motion of garments, and the second is a new method for acquiring facial performances of actors. We illustrate the individual parts of this thesis in Figure 1.1, along with a diagram of how they are related in the hierarchy.

Multi-Camera Synchronization. In Chapter 3, we discuss the problem of temporally synchronizing video streams from multiple cameras. Camera synchronization is often achieved using machine vision cameras with hardware triggers. However, these cameras tend to be very expensive and they require an array of computers for streaming the video, making capture setups less portable. A cheaper, more portable solution is to use high-resolution consumer camcorders with on-board hard-drives. However most of those do not support hardware synchronization and contain a rolling-shutter camera model that can create a temporal shear in the video stream. We present two solutions for synchronizing these inexpensive camcorders. The first solution is to track a moving object in the scene, visible by all cameras (Section 3.2). Our second, more accurate approach is to perform optical synchronization using stroboscopic illumination (Section 3.3). Strobe lights create simultaneous exposure images for all cameras, automatically synchronizing the streams. This method

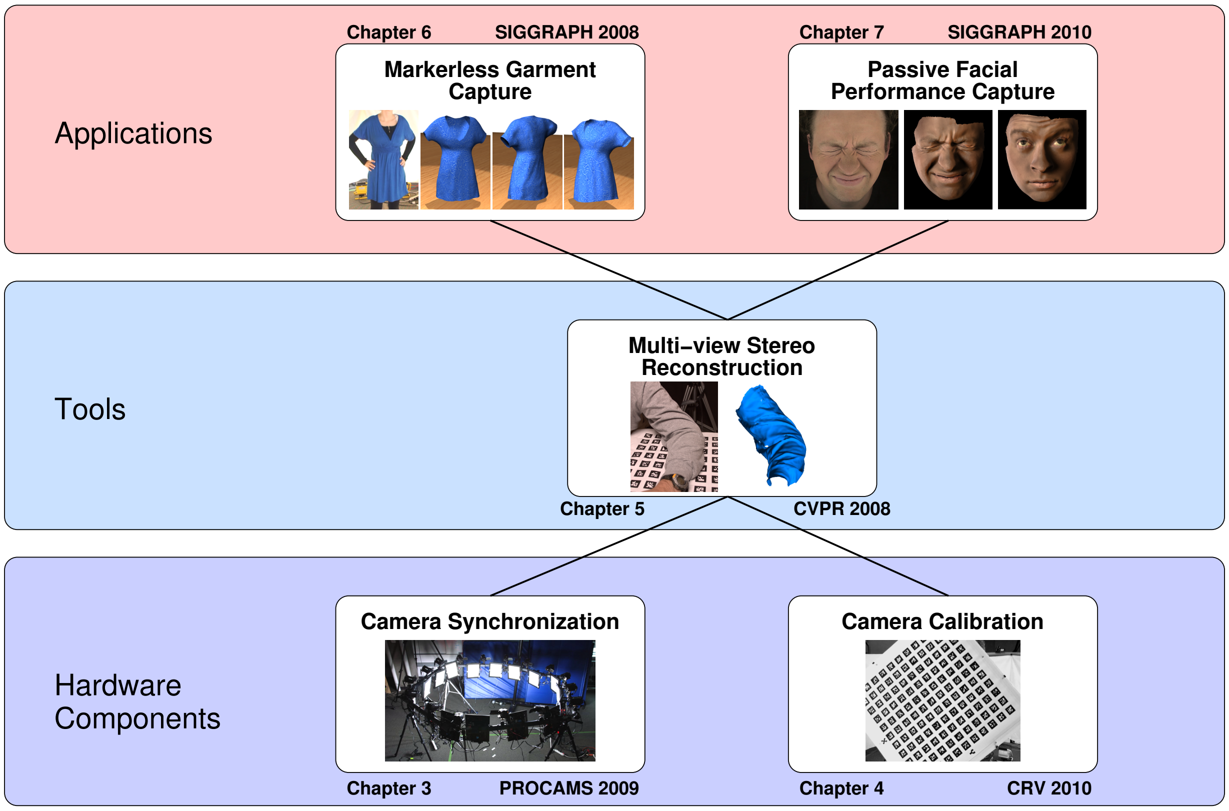


Figure 1.1: Individual thesis contributions and our hierarchy of shape and motion capture.

also removes any temporal distortion caused by the rolling shutter model found in these cameras. This work forms part of a publication that was presented at the International Workshop on Projector-Camera Systems (PROCAMS) in 2009 [Brad 09], which won the second-best paper award. I was the primary author on this work, although the paper contained an additional solution for synchronization and rolling shutter correction, which is not a contribution of this thesis. That additional method was designed by Bradley Atcheson, who was a co-author.

Multi-Camera Calibration. Camera calibration is the process of computing the intrinsic and extrinsic parameters of a camera model, usually from multiple images of a specific calibration pattern. Robust camera calibration is essential for performing 3D reconstruction from images. This is accomplished by minimizing the reprojection error of a number of known points on the calibration plane, which are detected in the images. The reprojection error is also commonly used for comparing the quality of different camera calibrations, for example when choosing the best calibration from a set. While this measure is suitable for single cameras, we show in Chapter 4 that we can improve calibrations in a multi-camera setup by calibrating the cameras in pairs using a new *rectification error*. The rectification error determines the mismatch in epipolar constraints between pairs of cameras. We use this error measure to calibrate multi-camera setups more accurately than using the reprojection error, yielding higher-quality 3D reconstructions in all of our experiments. This research has been published at the Canadian Conference on Computer and Robot Vision in 2010 [Brad 10a].

Multi-View Stereo Reconstruction. Multi-view stereo reconstruction is the process of generating a 3D model from multiple images. Chapter 5 describes a new algorithm for multi-view stereo that demonstrates both accuracy and efficiency. Our method is based on robust binocular stereo matching, followed by adaptive point-based filtering of merged point clouds, and finally efficient, high-quality mesh generation. All aspects of our method are designed to be highly scalable with the number of views. To date, our technique produces some of the most accurate results among current algorithms for a sparse number of viewpoints according to the Middlebury datasets [mvie]. Additionally, our method proves to be the most efficient method among non-GPU algorithms for the same datasets. Finally, our technique also excels at reconstructing deformable objects with high-curvature surfaces, which we demonstrate with a number of static examples. Our multi-view stereo technique was published at the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) in 2008 [Brad 08a], which is one of the top venues for computer vision research. My contribution for this thesis focuses on the novel binocular stereo matching approach. The point-based filtering and mesh generation steps were designed by Tamy Boubekeur.

Multi-view stereo is used as the basis for capturing time-varying, deformable objects using video cameras. The efficiency of our multi-view technique allows us to process hundreds or even thousands of video frames per capture sequence. Also, the accuracy for a small number of viewpoints makes the physical camera setups more practical, feasible and affordable.

Using our advances in camera synchronization, calibration and multi-view stereo reconstruction, we present two applications for 3D reconstruction of temporally deforming surfaces.

Markerless Garment Capture. The first application is a method for markerless garment capture, described in Chapter 6. A lot of research has recently focused on the problem of acquiring the geometry and motion of garments [Gusk 03, Scho 05, Whit 07]. Such work usually relies on special markers printed on the fabric to establish temporally coherent correspondences between points on the garment surface at different times. Unfortunately, this approach is tedious and prevents the capture of off-the-shelf clothing made from interesting fabrics. We present a *marker-free* approach to reconstruct garment motion that avoids these downsides. Since motion can be acquired from establishing a consistent mapping between all frames, we start by computing temporally coherent parameterizations between incomplete geometries that we extract at each time-step with our multi-view stereo algorithm. We then fill holes in the geometry using a template mesh, which is built from a single photo of the garment. The template is a complete mesh, containing any missing geometry that may not be reconstructed in the sequence, for example due to occlusions. This approach, for the first time, allows us to capture the geometry and motion of unpatterned, off-the-shelf garments made from a range of different fabrics. Our work on markerless garment capture was presented at the ACM SIGGRAPH conference in 2008 [Brad 08b], and published as a journal article in ACM Transactions on Graphics. This publication venue is the most prestigious in computer graphics. This research was joint work with another Ph.D. student, Tiberiu Popa. I was the primary author and was involved in all aspects of the project. The main contributions for this thesis include the physical acquisition design (Section 6.2), the initial reconstruction of the garment models (Section 6.3), and shared contributions to the high-level design of the geometry processing algorithms (Section 6.4).

Passive Facial Performance Capture. The final contribution of this thesis is a method for capturing human facial performances, described in Chapter 7. Facial performance capture is an indispensable tool in the film and video game industries, since faces are very difficult to model correctly by an artist. The main goal is to record the face of an actor and extract both the time-varying geometry and per-frame texture maps from the videos. We introduce a purely passive facial capture approach that uses only an array of video cameras, and requires no template facial geometry, no special makeup or markers, and no active lighting. We obtain initial geometry using an enhanced version of our multi-view stereo technique, and then use a novel approach for automatically tracking texture detail across the frames using optical flow. This method supports automatic drift correction, as well as an edge-based mouth tracking approach, which together yield realistic, time-varying, textured facial models. The resulting sequence can be rendered with our dynamically captured textures, while also consistently applying texture changes such as virtual makeup. This work has been accepted for publication at the ACM SIGGRAPH conference in 2010 [Brad 10b], which as mentioned previously, is the highest level of publication possible in computer graphics. I am the primary author on this work and was responsible for all aspects of the project, although I received valuable advice from Tiberiu Popa, who is a co-author. As part of this project, I supervised Steven Stuber, an undergraduate student who designed a customized hardware solution for controlling our arrays of cameras and lights.

This thesis advances the state-of-the-art in 3D reconstruction of temporally deforming objects from multiple video cameras, by introducing markerless methods for capturing garments and human facial performances. These new, inexpensive applications are made possible by our novel approaches for multi-camera synchronization and calibration, and our robust technique for multi-view stereo reconstruction. The results presented in this thesis have the potential for high impact in the entertainment industry, because most commercial products currently rely on marker-based technology. We also believe that this work is among the first to define markerless video-based capture as an important topic in the research community, potentially inspiring further innovative ideas in a field that is gaining more and more interest every year.

Chapter 2

Related Work

This chapter outlines previous work related to the different areas of this thesis. We start with the lower level topics of camera synchronization and calibration, followed by multi-view stereo reconstruction, and end with techniques related to our work in capturing garments and human facial performances.

2.1 Multi-Camera Synchronization

Camera synchronization is the process of temporally aligning the individual frames from multiple cameras. This is often performed in hardware using machine vision cameras, however these cameras tend to be very expensive and they require an array of computers for streaming the video. The alternative is to use consumer camcorders, which are both inexpensive and portable, however they cannot be hardware-synchronized. These cameras yield an additional challenge in that they are *complementary metal-oxide-semiconductor (CMOS)* cameras, which can suffer from spatio-temporal distortions in the video due to a "rolling" shutter model. In this model, every scanline is exposed slightly later in time than the previous one, causing a shear in the images. Removing this shear is known as the process of rolling-shutter distortion correction. In Chapter 3 we present solutions for both synchronization of multiple cameras and rolling shutter distortion correction, enabling the use of these inexpensive and portable camcorders for computer vision applications. This section describes related work in the areas of camera synchronization and rolling shutter distortion correction.

Rolling Shutter Cameras in Computer Vision. Due to the time-sheared exposures of different scanlines, rolling shutter cameras are not commonly used in computer vision despite their affordability. However, over the past several years, analysis of this sensor type has increased and a few applications have been described in the literature. Wilburn et al. [Wilb 04] use an array of rolling shutter cameras to record high-speed video. The camera array is closely spaced and groups of cameras are hardware triggered at staggered time intervals to record high-speed video footage. Geometric distortions due to different view points of the cameras are removed by warping the acquired images. To compensate for rolling shutter distortions, the authors sort scanlines from different cameras into a virtual view that is distortion free. Ait-Aider et al. [Ait 07] recover object kinematics from a single rolling shutter image using knowledge of straight lines that are imaged as curves.

Wang and Yang [Wang 05] consider dynamic light field rendering from unsynchronized camera footage. They assume that images are tagged with time stamps and use the known time offsets to first compute a virtual common time frame for all cameras and afterwards perform spatial warping to generate novel views. Camera images are assumed to be taken with a global shutter.

Rolling Shutter Camera Models and Distortion Correction. Although there are hardware solutions for the CMOS rolling shutter problem, e.g. [Wany 03], these are often not desirable since the transistor count on the chip increases significantly, which reduces the pixel fill-factor of the chip. Lately, camera models for rolling shutter cameras have been proposed, taking camera motion and scene geometry into account. Meingast et al. [Mein 05] develop an analytic rolling shutter projection model and analyze the behavior of rolling shutter cameras under specific camera or object motions. Rolling shutter images can be undistorted in software. For example, Liang et al. [Lian 08, Lian 05] describe motion estimation based on coarse block matching. They then smooth the results by fitting Bézier curves to the motion data, and the motion vector field is used for image compensation. Nicklin et al. [Nick 07] describe rolling shutter compensation in a robotic application. They simplify the problem by assuming that no motion parallax is present, and so their solution does not generalize to arbitrary camera setups.

Synchronization of Multiple Video Sequences. Computer vision research has been concerned with the use of unsynchronized camera arrays for purposes such as geometry reconstruction. For this purpose it is necessary to virtually synchronize the camera footage of two or more unsynchronized cameras. All work in this area has so far assumed the use of global shutter cameras. The problem of synchronizing two video sequences was first introduced by Stein [Stei 98]. Since Stein's seminal work, several authors have investigated this problem. Most synchronization algorithms are based on some form of feature tracking [Casp 06]. Often, feature point trajectories are used in conjuction with geometric constraints relating the cameras like homographies [Dai 06, Stei 98], the fundamental matrix [Carc 04, Sinh 04a] or the tri-focal tensor [Lei 06]. The algorithms differ in how the feature information is matched and whether frame or sub-frame accuracy can be achieved. Most authors consider the two-sequence problem, but N-sequence synchronization has also been considered [Carc 04, Lei 06].

A different approach to N-sequence synchronization has been proposed by Shrestha et al. [Shre 06]. The authors investigate the problem of synchronizing video sequences from different consumer camcorders recording a common indoor event. By assuming that in addition to the video cameras, the event is being captured by visitors using still cameras with flashes, they propose to analyze flash patterns in the different video streams. By matching binary flash patterns throughout the video sequences, frame-level synchronization can be achieved. However, this approach is designed for special events with visitors taking photos, such as weddings.

Stroboscopic Illumination. One of the methods we propose for synchronization of multiple cameras in Chapter 3 makes use of stroboscopic illumination. Strobe lighting has been used to capture multi-exposure images. Classic examples include early photographic work by Harold E. Edgerton and Gjon Mili to capture high-speed events on film. Lately, computer vision techniques have used this principle to recover trajectories of high speed motions, e.g. Theobalt et al. [Theo 04] track the hand motion and ball trajectory of a baseball player. Linz et al. [Linz 08] recover flow fields from multi-exposure images to generate intermediate single exposure views and synthetic motion blur.

Summarizing, related work has been concerned with analysis of the rolling shutter camera model and separate methods for distortion correction and multi-camera synchronization in software. However, we demonstrate a combined solution in Chapter 3 that performs optical synchronization and automatically removes rolling shutter distortion using stroboscopic illumination.

2.2 Multi-Camera Calibration

Calibrating a camera is the process of computing both the internal and external parameters of a camera. The external parameters define the position and orientation of the camera in the world, and the internal parameters specify how the camera creates an image. Together, the full set of parameters form the projection matrix for the camera. Camera calibration is one of the most fundamental aspects of computer vision. Although this thesis focuses on *multi-camera* setups, we start by describing the state-of-the-art in *single* camera calibration and then discuss multi-camera calibration.

Single Camera Calibration. The two most widely used methods for camera calibration are the techniques presented by Tsai [Tsai 86] and Zhang [Zhan 99, Zhan 00]. Both techniques require a number of 3D to 2D point correspondences as input. That is to say, the 2D pixel locations for a number of known 3D points in the world. These correspondences are often found using a checkerboard or other calibration pattern (e.g. [Fial 08]). The camera parameters are then estimated by minimizing the reprojection error between the measured 2D pixels and the projection of the corresponding 3D points using the estimated parameters. The method of Tsai can solve for the calibration using a single image of a calibration grid, while the method of Zhang typically requires multiple different grid orientations. In this case the algorithm solves for a single set of internal parameters common for all input images, and multiple sets of external parameters, one for each of the input images. The external parameters that produced the lowest reprojection error would then be chosen for the camera. In this thesis we use the method of Zhang in a multi-camera calibration algorithm, described in Chapter 4.

Multi-Camera Calibration. In a naive way, the method of Zhang can be used to calibrate multiple cameras simultaneously. Assume that all cameras in a particular setup observe the same sequence of calibration grid orientations, each from their own respective viewpoint. Then Zhang's method can be applied independently to each camera, providing per-camera internal parameters and multiple sets of external parameters. The standard approach is then to use the external parameters with the lowest average reprojection error among all cameras. We will show that this standard approach can be inaccurate for some cameras, and that better accuracy can be achieved by solving for the calibration of pairs of cameras, considering the epipolar geometry between the pairs. Rather than using the reprojection error, we define a *rectification error* measure for this purpose.

Combining epipolar geometry and calibration has proven useful in past research. A similar measure to our rectification error was used by Furukawa and Ponce to evaluate their calibration algorithm [Furu 09a], and Sinha et al. also use epipolar geometry to calibrate camera networks from the silhouettes of objects [Sinh 04b].

A related technique for finding the parameters of a multi-camera setup is bundle adjustment [Hart 94]. The goal of bundle adjustment is to simultaneously solve for optimal scene structure and camera parameters given a set of 2D interest points that are common among subsets of views. However, bundle adjustment can fail if there is insufficient overlap between the camera views. It has been suggested that each feature point should be visible in at least four views in order to produce reliable camera parameter estimates [Trig 99, Hung 06]. It is not always possible to provide such redundancy, for example in 360° sparse multi-view reconstruction, where only a small number of cameras are available [Vlas 08, Agui 08]. Since we only require pairs of cameras to have overlapping views, our rectification error can be used to find very accurate camera calibrations in these cases, more accurate than using the standard reprojection error, according to our experiments. Our method can be considered an alternative to bundle adjustment, specifically designed to target pairs of cameras and establish accurate relative calibrations.

2.3 Multi-View Stereo Reconstruction

Multi-view stereo (MVS) reconstruction is the process of building a 3D model from multiple images of an object or scene. The multi-view stereo problem has received a lot of attention recently, yielding a variety of reconstruction algorithms. Following the taxonomy of Seitz et al. [Seit 06], MVS algorithms can be categorized into four classes: 3D volumetric approaches [Horn 06, Tran 06, Vogi 07, Sorm 07, Sinh 07, Ladi 08, Kole 09], surface evolution techniques [Este 04, Pons 05, Garg 07, Zaha 07, Zaha 08, Dela 08], algorithms that compute and merge depth maps [Szel 99, Goes 06, Stre 06, Zach 07, Merr 07, Camp 08, Zach 08, Lamb 09, Liu 09a, Liu 09b], and techniques that grow regions or surfaces starting from a set of extracted feature or seed points [Goes 07, Laba 07, Furu 07, Furu 09b, Habb 07, Aucl 08, Janc 09, Hiep 09]. The algorithm we propose in Chapter 5 falls into the third category, and thus our discussion of earlier work focuses on other techniques that also compute and merge depth maps. We refer the reader to [Seit 06] and the MVS evaluation benchmark [mvie] for a more thorough discussion of the other techniques. This evaluation benchmark is the most widely used method for comparing MVS algorithms, in terms of accuracy, efficiency, and overall completeness of the reconstructed models. In the following, we refer to this benchmark several times when comparing previous techniques to our method.

Depth Map Approaches. These techniques are generally two-step approaches. First, depth maps are computed from the input images. Second, the multiple depth maps are merged to create a 3D model. While some algorithms consider only the first step, and some only the second, many techniques are designed to solve both steps. Our algorithm in Chapter 5 solves both steps, however the contributions for this thesis include only the depth map computation technique.

A multi-view framework for computing dense depth estimates was first proposed by Szeliski [Szel 99], who formulates the problem as a global optimization over the unknown depth maps. Strecha et al. [Stre 06] jointly solve for depth and visibility using a generative model, where input images are assumed to be generated by either an inlier process or an outlier process. Depth and visibility are modeled as a Hidden Markov Random Field (MRF). Computation times are comparatively low for a sparse set of viewpoints, however they do not scale well. Campbell et al. [Camp 08] also use MRF optimization to improve the quality of depth maps in sparse camera setups. They store multiple depth hypotheses for each pixel and enforce spatial consistency to remove outliers. The accuracy of their results are on par with our algorithm, however their running times are an order of magnitude slower. Lambert and Hébert [Lamb 09] also consider multiple depth hypotheses. However, rather than using traditional photo-consistency matching, they globally minimize a frequency criterion defined for surface light field rendering, which is free from any reflectance properties. This helps to remove outliers in the surface reconstruction under both Lambertian and non-Lambertian reflectance. Unfortunately, this global optimization requires dense viewpoint sampling and makes their algorithm one of the slowest among all state-of-the-art techniques. Liu et al. [Liu 09b] build depth maps, merge point clouds and create final meshes for video sequences using a similar pipeline to ours. Their technique includes a matching process that is robust to occlusion, noise and lack of texture. However, their results tend to be over-smoothed due to the use of Poisson surface reconstruction, and neither the accuracy nor the speed of this technique matches our algorithm. Liu et al. [Liu 09a] have a second algorithm for multi-view reconstruction that computes continuous rather than discrete depth maps using a variational approach. They also employ the common technique of computing multiple depth hypotheses, which are generated from starting at different scales. Again they use Poisson reconstruction to merge the depth maps, which can lead to over-smoothed meshes. While the accuracy and efficiency of this method outperforms their first algorithm, our technique is both more accurate and faster than both.

As mentioned, some algorithms consider only the second step of the process, taking as input a set of depth maps and integrating them into a 3D model [Zach 07, Merr 07, Zach 08]. However, the work presented in this thesis focuses mainly on the first step, which is to compute the depth maps from the input images.

Our work is most similar to that of Goesele et al. [Goes 06], who showed that simple modifications to original window-based stereo algorithms can produce accurate results. In their algorithm, depth maps are computed by backprojecting the ray for each pixel into the volume and then reprojecting each discrete location along the ray onto neighboring views where window-based correlation is performed with sub-pixel accuracy. They choose only the points that correlate well in multiple views, and thus reconstruct only the portion of the scene that can be matched with high confidence. Finally, depth maps are merged with an off-the-shelf volumetric technique [Curl 96]. Although their method is simple to implement, their models suffer from a large number of holes and the processing time of their algorithm is very long.

In contrast to these previous methods, we present an algorithm in Chapter 5 that is both more efficient and achieves better accuracy combined with high density, when compared to other state-of-the-art techniques.

Two-View Stereo. Our algorithm for multi-view stereo starts by performing binocular or two-view stereo between pairs of adjacent cameras in the physical setup. Binocular stereo on its own is a fundamental topic in computer vision, and numerous algorithms have been proposed (see Scharstein and Szeliski [Scha 02] for a survey). However, our two-view stereo approach is specifically designed to be the first step in a multi-step process that combines stereo results from many pairs of cameras in a multi-view setting. Our algorithm is not designed to be a stand-alone two-view stereo solution. While traditional binocular stereo methods must deal with filtering of outliers and completion of occluded regions directly in the depth map, our binocular stereo algorithm avoids complicated filtering of depth maps, since these steps can be performed more accurately after combining the depth maps from several camera pairs into a single point cloud. For this reason, we do not compare our two-view stereo technique to the large body of work that is designed solely for binocular stereo.

2.4 Garment Capture

Garment motion capture is the process of reconstructing the time-varying shape and motion of garments from video cameras. This problem is particularly challenging due to the complex deformation that cloth can undergo over time. Over the last several years, techniques for capturing the motion of cloth have evolved from methods for tracking a single cloth sheet to systems for reconstructing full garments. This section provides an overview of these methods.

Sheets of Cloth and Partial Garments. Early work on this topic focused on single sheets of cloth with existing textures [Prit 03, Scho 04], or small subsets of a full garment with patterns that were custom-printed onto the fabric [Gusk 03]. Hasler et al. [Hasl 06] use an analysis-by-synthesis approach that sets their method apart from the other work. They too, however, rely on markers in the form of patterned cloth.

This early work identifies the basic task of combining the geometry of the cloth with a temporally coherent parameterization and introduces the use of markers as a solution to this problem. Capturing full garments with large areas of occlusion is, however, significantly more challenging.

Full Garments. A first solution to this problem was proposed by Scholz et al. [Scho 05], who suggested a special marker pattern composed of a matrix of colored dots, in which each 3×3 neighborhood of dots is uniquely identifiable. A simple thin-plate approach was used to fill any holes in the geometry. White et al. [Whit 07] improved on these results by thoroughly analyzing the design of suitable marker patterns. They also improve on the hole filling, using a new data-driven technique. This work is among the state of the art in garment capture and produces excellent results. However, like the work by Scholz et al., White et al.'s method requires custom-tailored garments made from fabric on which the marker patterns have been printed. It thus cannot deal with arbitrary off-the-shelf clothing.

Markerless Approaches. Recently, there has been a major push for markerless human capture [Rose 07, Hern 07, Agui 07, Ahme 08a, Ahme 08b, Rose 08, Furu 08]. Garment capture has different challenges than human capture due to the numerous folds and occlusions that occur over time. Vlasic et al. [Vlas 08] and de Aguiar et al. [Agui 08] propose solutions for capturing humans with loose clothing. Both techniques involve deforming a scanned template mesh to match the image silhouettes from a number of cameras. With these methods, fine scale details in the resulting meshes come from the scan of the template rather than the image sequence. Image silhouette extraction can also be unreliable and temporally inconsistent. More recently, Liu et al. [Liu 09b] and Vlasic et al. [Vlas 09] present methods for markerless reconstruction of humans and clothing. The method of Liu et al. relies on multi-view stereo, while Vlasic et al. fuse multi-view and photometric stereo to capture fine-scale wrinkle details. However, these two methods reconstruct different meshes for each time step independently, and do not produce temporally compatible mesh sequences. A temporally compatible mesh sequence is essential for re-texturing or editing the result in any way. Finally, Wand et al. [Wand 09] present a technique for reconstructing non-rigid shape and motion, which can be applied to garments. Starting with time-varying point clouds, their method avoids the need for a template mesh by recursively merging two subsequences which are adjacent in time, filling in missing data on the fly. The result is a sequence of compatible meshes, depicting a single shape and its deformation over time.

However, fine-scale details tend to be propagated throughout the entire sequence, yielding a result that does not entirely reflect the input data.

Unlike these previous methods, we propose a technique in Chapter 6 that is able to reconstruct high resolution models of full garments, deforming under normal human motion, without the need for printed on-surface markers or high-frequency surface texture. Using unique geometric properties of garments, we are able to produce temporally compatible mesh sequences which can be edited and re-textured for rendering.

2.5 Facial Performance Capture

Facial performance capture involves reconstructing the shape and motion of an actor's face from video cameras. With the addition of per-frame texture maps, the recovered geometry can be rendered to create realistic facial animations. This technique is evolving into a major tool for 3D content creation in both the movie and video game industries.

Previous techniques for facial performance capture can largely be divided into the following categories: methods that use markers or special face paint to guide the reconstruction, structured light approaches, and finally methods that fit parametric face models to observed video sequences.

Markers and Face Paint. Traditional marker-based face capture dates back to Williams [Will 90], and consists of covering the face with a large number of black dots or fluorescent colors [Guen 98, Lin 05] in order to track the geometry. Bickel et al. [Bick 07] augment the traditional methods to incorporate multiple scales of geometry and motion using markers, face paint, and an initial scan. Most recently, Furukawa and Ponce [Furu 09c] present a face capture technique that deforms a laser-scanned model to match a highly-painted face while regularizing non-rigid tangential deformations.

Unfortunately, these techniques have trouble reconstructing accurate per-frame face textures, and require expensive hardware to perform initial scans.

Structured Light. Another approach to facial performance capture is to combine cameras with at least one projector that casts a structured light pattern onto the face in order to provide dense surface texture. Zhang et al. [Zhan 04] use space-time stereo with structured light to reconstruct temporally smooth depth maps of a face. They then fit a deformable template model at each time step using optical flow. Wang et al. [Wang 04] project phase-shifted color-fringe patterns onto the face and acquire the 3D shape in real-time. They establish correspondences between frames using a multi-resolution model fitting approach. However, the resulting meshes have insufficient resolution (at most 16K vertices) for capturing fine-scale facial details such as wrinkles.

Ma et al. [Ma 08] achieve high-resolution reconstructions by interleaving structured light with spherical gradient photometric stereo [Ma 07] using the USC Light Stage. New facial performances are then synthesized using a marker-based, data-driven approach. However, this method requires expensive hardware.

Structured light approaches are unattractive because they can be distracting to the actor, and they suffer from the inability to reconstruct face textures without sacrificing temporal resolution by interleaving ambient illumination with the structured light.

Parametric Models from Video. The goal of this category of methods is to determine the parameters of a deformable face model from observing a video sequence without markers or structured light [Li 93, Essa 96, DeCa 96, Pigh 99]. However, the resulting face reconstructions tend to be very low resolution, lacking any person-specific details.

In a related work, Blanz et al. [Blan 03] re-animate faces in video by parameterizing a database of different laser-scanned faces and expressions. They can then estimate the 3D shape and pose of new faces in video images with a parameter fitting algorithm. But like the other parametric approaches, the final models tend to lack facial details.

Commercial Systems and Other Projects. A number of commercial systems have been developed for facial performance capture. Vicon's marker-based system[1] and Mova's fluorescent makeup CONTOUR Reality Capture[2] are two prominent examples, while Dimension Imaging 3D is among the first to use markerless facial reconstruction in industry[3]. Recently, Alexander et al. [Alex 09] created a photoreal facial modeling and animation system in the Digital Emily Project. This technology was used recently for realistic facial animation in *Avatar*. Finally, Borshukov et al. [Bors 03] recreate actors for *The Matrix Reloaded* using their Universal Capture system. These systems both start with laser-scanned models. The approach of Borshukov is most similar to ours. Optical flow and camera triangulation advances a face model over time, and a time-varying texture map is computed from multiple videos. Unlike our automatic approach, however, optical flow errors and drift are corrected using tedious manual geometry reshaping.

To our knowledge, ours is the first fully-automatic method to achieve the same quality as current state-of-the-art techniques, but without the need for markers, face paint, expensive hardware or structured light. It is worth noting that concurrent to our work, Beeler et al. [Beel 10] present a similar technique for passive face reconstruction that captures pore-scale geometry of static faces. Our automatic surface tracking method for generating temporally compatible animations could complement their approach.

[1]Vicon MX - www.vicon.com

[2]CONTOUR Reality Capture - www.mova.com

[3]www.di3d.com

Chapter 3

Multi-Camera Synchronization

In this chapter we discuss the problem of camera synchronization, while also looking at the issue of rolling shutter distortion correction.

In order to capture time-varying objects using multiple cameras, the cameras must be temporally synchronized. This is the problem of temporally aligning the individual frames from all cameras. Most often this is achieved using scientific machine vision cameras with hardware triggers, however these cameras have several drawbacks. First, they tend to be very expensive, and they require an array of computers with expensive hardware in order to stream the video, making capture setups less portable. Additionally, these cameras can have inconsistent frame rates. Consumer camcorders are evolving as promising alternatives to scientific cameras in many computer vision applications. They offer high resolution and guaranteed high frame rates at a significantly reduced cost. Also, integrated hard drives or other storage media eliminate the need to transfer video sequences in real-time to a computer, making multi-camera setups more portable.

However, there are also a number of challenges that currently limit the use of such camcorders, especially in multi-camera and camera array applications. First, consumer camcorders typically do not have support for hardware synchronization. Second, most consumer cameras employ a "rolling" shutter, in which the individual scanlines use a slightly different temporal offset for the exposure interval (see, e.g. [Wilb 04]). The resulting frames represent a sheared slice of the spatio-temporal video volume that cannot be used directly for many computer vision applications. In order to benefit from these inexpensive and portable cameras, we explore alternative methods for establishing the synchronization and removing the rolling shutter distortion.

We present two different techniques for camera synchronization. The first one is a simple approach, based on tracking a moving object that is visible by all cameras. The second method performs optical synchronization by using strobe illumination [Brad 09]. Strobe lights create simultaneous exposure images for all cameras that can be used for synchronization. The simultaneous strobe flash also removes the rolling shutter problem, although the scanlines for a single flash are usually distributed across two frames (or fields, with interlacing).

Before we discuss our two synchronization approaches, we first describe the rolling shutter camera model on which we base our experiments.

3.1 Camera Model

Both of our synchronization methods target inexpensive consumer-grade video cameras and camcorders. Recently, there has been a push to replace CCD chips with CMOS sensors. These sensors have a number of advantages, two of which include lower price and power consumption, however they can also introduce rolling shutter distortions that we aim to model and eliminate.

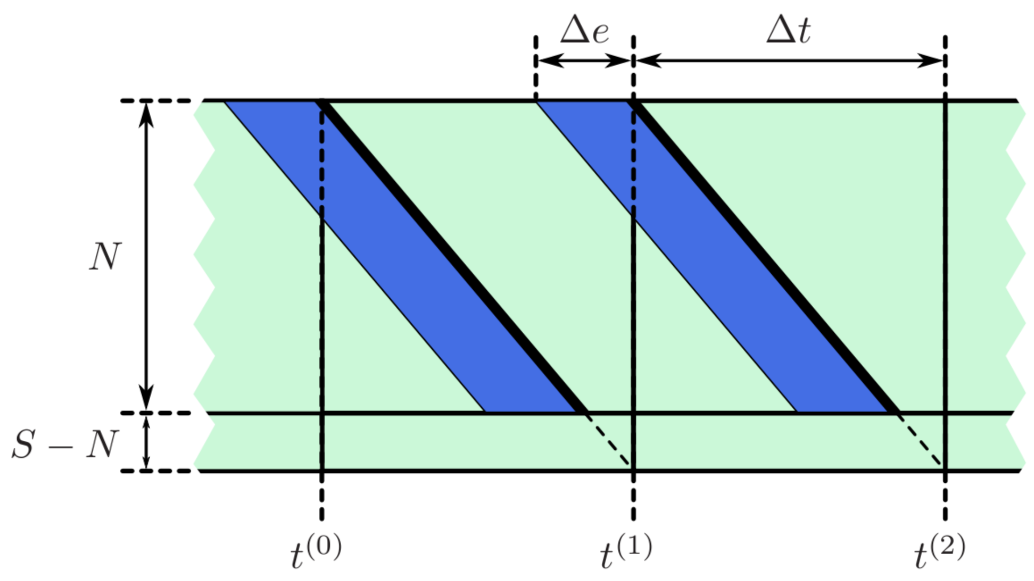


Figure 3.1: Rolling shutter camera model. Just-in-time exposure and readout of the individual scanlines creates a shear of the exposure intervals along the time axis. The slope of this shear is a function of the camera frame rate and the period is determined by the number of scanlines in the video format.

Rolling Shutter. To minimize buffer memory, the less expensive CMOS cameras read out each individual scanline from the sensor chip just in time for generating the video signal for that scanline. The exposure interval for each scanline starts some fixed time before that readout time, so that effectively the exposure interval is temporally offset for each individual scanline. More specifically, we can model the readout time $r_y^{(j)}$ for scanline y in frame j as follows (also see Figure 3.1):

$$r_y^{(j)} = t^{(j)} + \frac{y}{S} \cdot \Delta t = t^{(0)} + \left(j + \frac{y}{S} \right) \cdot \Delta t, \tag{3.1}$$

where Δt is the frame duration (one over the frame rate), S the total number of scanlines per frame, and $t^{(j)}$ the readout time of the topmost (visible) scanline in frame j. Since CMOS sensors operate similar to RAM, we can model pixel readout as instantaneous for our purposes. The exposure interval for scanline y in frame j is then given as

$$E_e^{(j)} = \left[r_y^{(j)} - \Delta e, r_y^{(j)} \right], \tag{3.2}$$

where Δe is the duration of exposure (exposure time).

Note that the total number S of scanlines may be larger than the number N of *visible* scanlines. For example, the specification for high definition video [ITU 90] calls for $S = 1125$ total lines for video standards with $N = 1080$ visible lines. The extra 45 invisible lines correspond to the vertical synchronization signal. For standard definition video, the number of invisible scanlines is 39 ($S = 525$) in NTSC [ITU 07].

Interlacing. Most consumer cameras trade spatial for temporal resolution by recording the even and the odd scanlines in separate fields (interlacing).

The theoretical analysis from above can easily be adapted to interlaced video, where the temporal separation between fields is now $\Delta t/2$, and the number of visible and total scanlines is $N/2$ and $S/2$, respectively. Note that S is an *odd* number for most video standards, which means that the duration of a field is slightly different for odd and even frames. However, this difference is only a fraction of a percent of the field time, and will be ignored for our purposes.

Synchronization. For synchronization of multiple cameras, we assume that all cameras follow the same video standard, i.e. that Δt, S, and N are identical for all cameras, and that either all or none of the cameras use interlacing. These assumptions are easily met if all cameras in the array are the same model. A possible concern is the potential for slight differences in the frame rate across individual cameras. However, even inexpensive cameras appear to have very good accuracy and stability with respect to frame rate. In our experiments with up to 16 cameras and several minutes of video duration, per-camera differences did not appear to have a visible impact, see also Section 3.4.1 for a detailed experiment description.

Aperture, Exposure and Gain. Ideally we would like to explicitly control certain camera parameters such as aperture, exposure and gain, in order to maximize the depth of field while minimizing camera noise and motion blur. However, most inexpensive consumer camcorders provide limited control over these parameters. The cameras we use in this thesis do not provide any control over the aperture. The exposure and gain parameters are linked to a single rotating dial, which controls exposure unless the maximum exposure time is surpassed and then the camera gain is increased as the dial is rotated further. The numerical values of these parameters are not exposed to the user, however through experience we were able to discover suitable exposure and gain settings for all of our experiments.

3.2 Moving-Object Synchronization

For cameras that trigger at different times, frame-level precise synchronization can be established by finding an integer frame offset for each camera, relative to one reference camera. We use a *moving-object* technique that is fully automatic and very robust. Moving-object synchronization methods work by tracking the motion of some object in each camera simultaneously, and then using the motion to compute the frame offsets. This must be performed at the start of every capture session since the offsets will be different each time the cameras start recording. Each camera must be able to observe and measure the motion of the object, so the type of motion depends on the physical arrangement of the cameras. In our method we use rotation, since this type of motion fits naturally in the camera ring setups used most often for reconstruction. We rotate a calibration grid on a turntable, by hand, in the center of the reconstruction volume (see Figure 3.2). Each camera automatically locates the grid in each frame, and the frame offsets are computed as rotational offsets relative to one camera that is used as the reference. We assume that the grid rotates about a single point in one plane. Note that computing this rotation is independent of camera parameters.

In order to visualize the grid rotation from each static camera, we can instead illustrate the rotation of the cameras about a static grid. This visualization is shown in Figure 3.3 (top) for a setup of sixteen cameras, where the camera positions are projected onto the plane of rotation and rendered as points, and each camera

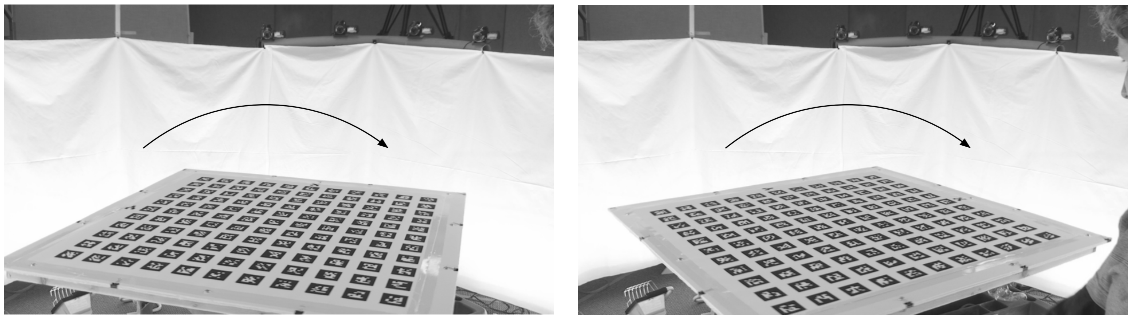


Figure 3.2: Moving-object synchronization by rotating calibration grid.

is drawn in a different color. For each camera, a circle is fit to the motion on the plane and then a common center of rotation is computed as the average center from all cameras. The first frame is captured before the grid starts rotating, so every camera has a common starting orientation. Then for all frames of each camera, a 2D rotation angle is computed relative to the starting orientation. The rotation angles are plotted for all cameras in Figure 3.3 (bottom). Finally, the temporal offset for camera c_i relative to camera c_0 is equivalent to the horizontal translation that best matches the rotation angle curve corresponding to c_i with the curve corresponding to c_0 in the plot. In other words, the camera offset is the average frame offset among all rotation angles. For robustness, we consider only rotation angles where the temporal gradient is large (i.e. when the object was moving fast).

Using this technique we measure sub-frame offsets for each camera, with a standard deviation of approximately 0.25 frames. For many applications, rounding the offsets to the nearest integer frame is sufficient. However, sub-frame synchronization could be achieved by computing optical flow for every camera and using the synchronization offset as a flow vector weight to generate an intermediate image between every pair of adjacent frames in the entire sequence. This would simulate images captured at the desired trigger time. Note, however, that this optical flow technique for sub-frame synchronization has not actually been implemented, as the stroboscope synchronization method described next is more precise.

3.3 Stroboscopic Synchronization

Stroboscopes have long been used for obtaining instantaneous exposures of moving objects using standard cameras without rolling shutters (e.g. [Theo 04]). An extension of this approach to multi-camera systems results in multiple video streams that are *optically* synchronized through illumination. Unfortunately, this straightforward approach fails for rolling shutter cameras, which we address in the following.

With our stroboscope approach, we can solve the rolling shutter problem for individual cameras, or simultaneously solve the synchronization and rolling shutter problems for an array of cameras. With no ambient illumination, stroboscopes create simultaneous exposures for all cameras. However, with rolling shutters, the exposed scanlines are usually divided between two adjacent frames. In our technique, we combine two partially exposed frames to form a single synchronized exposure image for each camera. Since all scanlines are exposed by the flash at the same time, this method avoids the temporal shear usually caused by rolling shutter cameras.

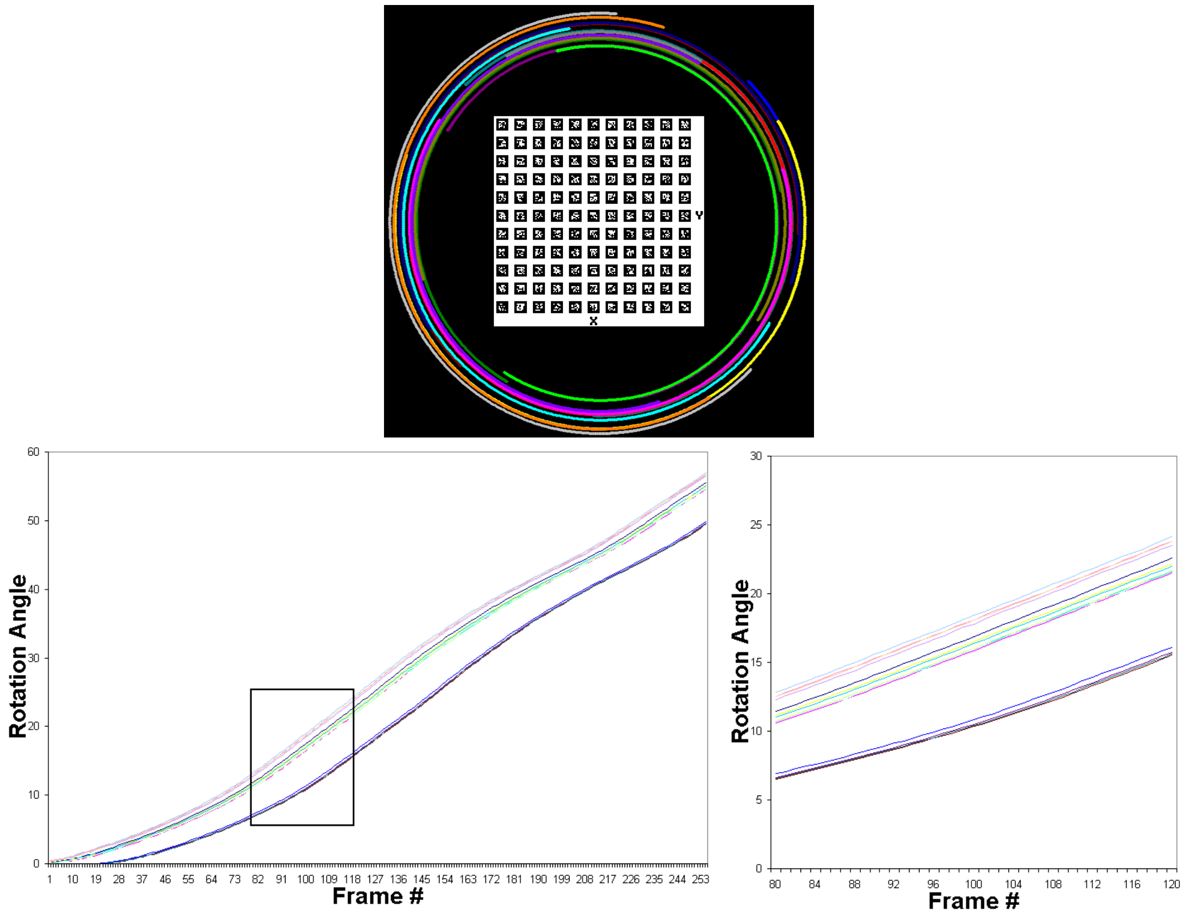


Figure 3.3: Grid rotation visualization. Top: camera positions shown on rotation plane. Bottom left: Rotation angles plotted for all cameras. Bottom right: Zoom region of rotation angles.

Basic Approach. In the single camera setting, the camera starts recording a dark scene in a normal way. Stroboscopic illumination is then activated, creating the exposures. For now, we will assume the flash is instantaneous. The flash is captured by all scanlines that are exposing at the time of the flash. The number of scanlines that record the event is determined by the exposure time of the camera, Δe (recall Figure 3.1). In the most basic approach, we ensure that all scanlines record the flash by maximizing Δe (see Figure 3.4) creating a continuous exposure with respect to the camera. Due to the overlapping exposure windows of the scanlines in rolling shutter cameras, the strobe flash is usually split between two consecutive frames. The two frames containing the instantaneous exposure can be combined with addition, or in this simple case a maximum operator as we illustrate in Figure 3.4. Note that we use a binary marker grid simply as a basic scene for demonstrating the technique.

In a multi-camera setting, each camera independently captures the scene with the strobe illumination. The per-camera rolling shutter compensation as described above automatically synchronizes the array.

Virtual Frame Rate. Although the cameras record frames at a certain frame rate ($1/\Delta t$), the frequency of the strobe lighting creates a *virtual frame rate* for the video sequence. This is because one output frame

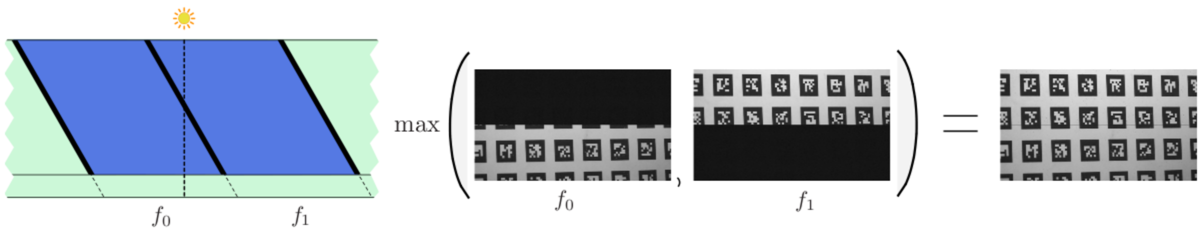


Figure 3.4: In rolling shutter cameras, consecutive frames that contain the instantly exposed scanlines are combined to make the final image.

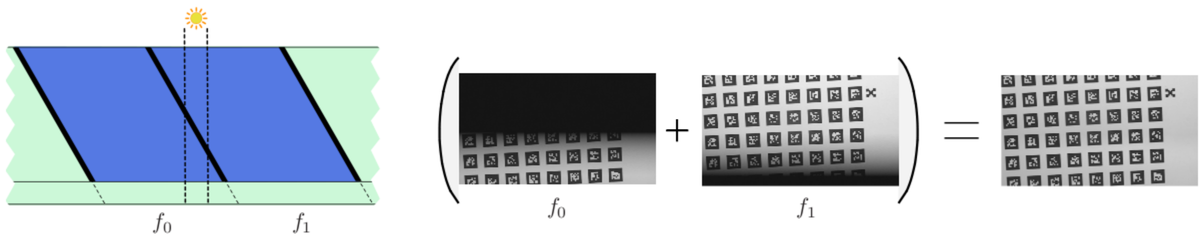


Figure 3.5: Increasing the flash duration creates a virtual exposure time. The exposed scanlines overlap with a ramp up at the beginning and ramp down at the end. Summing the frames in linear intensity space creates the final image.

is generated for each flash of the stroboscope. The maximum frequency that avoids double-exposure of scanlines is $1/\Delta t$. However, flashing at this frequency tightly packs the instantaneous exposures with no gap of nearly-black pixels between them. Leaving a gap between the exposures helps to separate them, especially in the case of minor drift if the stroboscope frequency cannot be set precisely to $1/\Delta t$. The simplest approach is to set the strobe frequency to half the camera frame rate ($1/2\Delta t$), creating a full frame of unexposed scanlines between every exposure. Note that the unexposed scanlines are also split between two consecutive frames, exactly like the exposed scanlines. If this reduction in temporal resolution is acceptable, then every pair of adjacent frames can be combined in the straightforward manner described above. If a higher virtual frame rate is desired, the frames can still be combined automatically with a little more computational effort to explicitly search for the unexposed scanlines that separate the frames. This technique is robust to any minor drift that may occur over time, if the strobe frequency cannot be set with high precision.

This method removes motion blur from the video frames, which allows us to capture very fast moving objects (see Figure 3.7). The downside of this approach is that an instantaneous strobe flash may not produce very much illumination, resulting in increased camera noise. We now describe how a controllable, longer flash duration allows us to trade off noise for motion blur.

Virtual Exposure Time. As we mentioned previously, varying the flash rate creates a virtual frame rate for the output sequence. We can also create a *virtual exposure time*, Δe_v, by varying the duration of the flashes. As we see in Figure 3.5, increasing the flash duration creates an overlap of exposed scanlines in consecutive frames. The amount of overlap is directly related to the virtual exposure time, i.e. the number

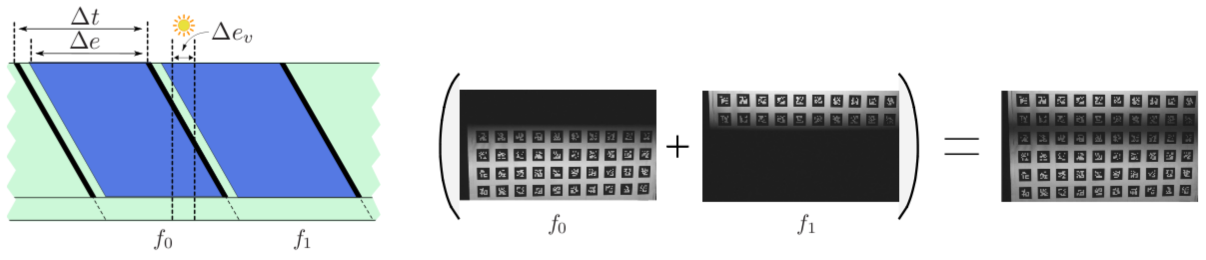


Figure 3.6: A non-continuous camera exposure results in scanlines with less than full intensity after the summation.

of scanlines that can be read out during the flash duration. The exposure also ramps up at the beginning and ramps down at the end, though summing the two frames in linear intensity space gives the correct final image. In practice the camera data is not linear, however many cameras follow a standard gamma curve which can be inverted before adding the frames. It can also happen that the exposed scanlines span three frames due to the scanline overlap, but the linear summation is still the same in this case.

Having a longer virtual exposure time creates less noisy images as the camera gain can be minimized. However, this comes at the expense of motion blur for fast moving objects. By allowing a controllable flash duration, the trade-off between noise and motion blur can be chosen depending on the application (see Section 3.4.3 for a detailed experiment).

Non-Continuous Exposure. In the most general camera model, the exposure time Δe can also vary such that the camera is not recording continuously. In this case, summing the adjacent frames no longer gives a full-intensity result for all scanlines, since there is less overlap between the frames (see Figure 3.6). However, as long as $\Delta e_v > \Delta t - \Delta e$, we can artificially boost the gain per scanline to recover the frame. The per-scanline scale factor can be computed by recording a diffuse white surface. If the above inequality does not hold, i.e. either the camera exposure or the virtual exposure is too short, then there will be missing scanlines that cannot be recovered.

Interlaced Video. Thus far, our method has been described for non-interlaced video cameras, however it easily extends to interlaced video as well. Interlaced cameras separately record odd and even fields at twice the frame rate. This implies that a stroboscopic exposure is split between two fields rather than two frames. The two partially exposed fields can be combined into a single *half-frame* with the same technique described above. Note that we refer to this as a half-frame rather than a field since some of the scanlines are even and some are odd. The half-frame can then be converted to a full frame by interpolating the missing in-between scanlines. Although we only capture half the spatial resolution with each flash of the strobe, we gain up to twice the temporal resolution since the field-rate is twice the frame rate. This means that we can set the desired virtual frame rate and virtual exposure time, and then combine fields instead of frames at the cost of spatial resolution.

Our stroboscope illumination technique works effectively by starting the cameras first and the strobe lighting second. The first exposure flash can then be located in each camera, identifying the first synchronized frame. With this approach we can robustly synchronize multiple cameras without requiring specific image content such as trackable features or detailed texture.

3.4 Stroboscope Experiments

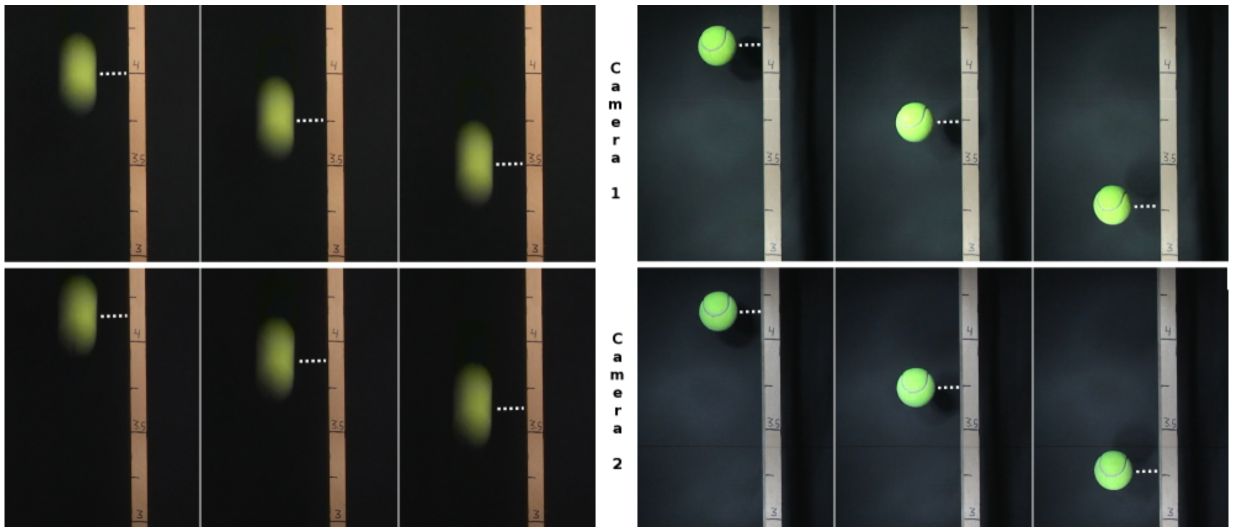


Figure 3.7: Stroboscope synchronization experiment. Left: 3 consecutive frames (left to right) from 2 unsynchronized cameras (top and bottom). Right: 3 consecutive frames (left to right) from 2 cameras (top and bottom) synchronized by strobe lighting.

For our experiments, we use between 1 and 16 Sony HDR-SR7 camcorders. These camcorders follow the 1080i/30 format [ITU 90]. That is, video is recorded at 29.97 frames per second (approximately 60 fields per second interlaced), and each frame has a final visible resolution of 1920×1080. Like in many consumer devices, the video is recorded in *anamorphic* mode, where the horizontal direction is undersampled by a factor of $4/3$, meaning that each frame is represented as two fields with a resolution of 1440×540. However, this anamorphic distortion does not affect any of the algorithms presented in this Chapter.

For the stroboscope illumination experiments with instantaneous exposure, we use 3 hardware-synchronized Monarch Instrument Nova-Strobe DAX stroboscopes. This model allows very precise flash rates (between 30 and 20,000 flashes per minute), and short flash durations (20-50 μs). For experiments with varying virtual exposure, we use up to 16 Chauvet LED Techno Strobes that we re-programmed to allow precise, variable flash duration and flash rates. We use multiple spatially distributed strobes instead of just one in order to increase the intensity and uniformity of the illumination.

We now discuss the synchronization and rolling shutter compensation experiments we performed in order to validate our stroboscopic technique. We conclude this section with an experiment demonstrating the trade-off between camera noise and motion blur when setting the virtual exposure time via strobe flash duration.

3.4.1 Synchronization

Our stroboscopic method for synchronization is demonstrated with the classic example of a falling ball, depicted in Figure 3.7. Two cameras observe a tennis ball falling beside a measuring stick. On the left side of Figure 3.7, we show three consecutive frames for each camera, captured with regular, constant illumination. The ball is falling quickly, so the cameras record motion blur. We measure the height of the ball at the center of the blur, as indicated by the dashed white line. It is clear that the cameras are not synchronized. On the right side of the figure, we show the same example using stroboscope illumination. Measuring the height of the ball demonstrates the precise optical synchronization. Note also that this method avoids motion blur, since the frames are captured at instantaneous moments in time. This benefit allows us to accurately capture very fast motions.

Note that the amount of motion between consecutive frames in the synchronized example is roughly twice that of the unsynchronized case, since the strobe frequency was set to half the camera frame rate.

Assessing Camera Drift. We tested frame rate stability and consistency for 16 consumer cameras of the same model. The cameras were arranged in a half-circle and pointed at a diffuse ball in the center. The ball was illuminated by stroboscope illumination set to the NTSC frame rate of 29.97 Hz. As we discussed, flashing at this rate results in a split image where the scanline at which the split occurs should not move. If either the camera frame rate or the strobe were exhibiting temporal drift, the split scanline would move up or down, indicating a mismatch in illumination and recording frequency. While observing video footage of the 16 cameras recorded for more than 2 minutes, we did not see temporal drift in any camera. Since all cameras were observing the same strobe signal this indicates that frame rates are very stable across different cameras. From this we conclude that the cameras have negligible temporal drift and good frame rate stability for extended periods of time.

3.4.2 Rolling Shutter Compensation

Rolling shutter distortion occurs when there is a motion in the scene. If the motion is orthogonal to the rolling shutter's direction this results in a warping of straight lines, and vertical motion results in stretch. Static scenes are obviously unaffected by the rolling shutter, whereas too fast a motion causes blur that somewhat hides the distortion. However, reasonably fast motion can cause quite severe distortion. The rotating checkerboard in Figure 3.8 (left) shows how straight lines are rendered as curves under a rolling shutter.

Our stroboscope illumination technique completely avoids distortion caused by rolling shutter cameras since all scanlines in a frame are exposed simultaneously by the strobe lighting. This is demonstrated in Figure 3.8 (right), where the rotating checkerboard is captured with stroboscope illumination. Despite the fast motion, straight lines are captured correctly.

Table 3.1 contains the residual vector error (L_2) as well as the worst case perpendicular distance (L_∞) for the (approximately) vertical and horizontal rows of indicated corner points to straight line fits. As we can see, the stroboscopic method effectively removes the rolling shutter distortion.

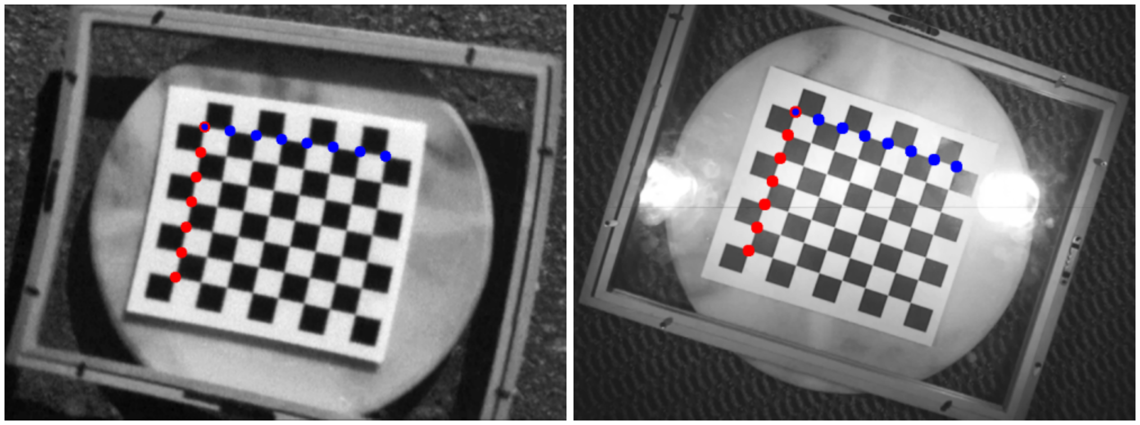


Figure 3.8: Left: a fast rotating checkerboard (1 rev. per second) in which straight lines are warped into curves. Right: the same scene captured with strobe illumination.

	Vertical line		Horizontal line	
	L_∞	L_2	L_∞	L_2
Fast-moving	1.38	2.55	0.19	0.29
Stationary	0.10	0.15	0.11	0.19
Stroboscope	0.10	0.18	0.09	0.17

Table 3.1: Norms of perpendicular distance error vectors for the indicated corner points to straight line fits. The vertical line (red), which crosses many scanlines, is more distorted than the horizontal (blue) line.

3.4.3 Camera Noise vs. Motion Blur

As we discussed, the virtual exposure time can be varied to trade off camera noise for motion blur, while keeping the camera exposure constant. We show this effect in Figure 3.9, with a scene consisting of a stationary checkerboard and a fast rotating ball with constant speed. On the left is a frame captured with a short virtual exposure (0.3 ms), and on the right is a similar frame captured with a longer virtual exposure (6 ms). For the short exposure, we set the camera gain to match the same overall intensity of the longer exposure.

The zoom regions at the bottom of Figure 3.9 show the noise and motion blur difference between the two different virtual exposures, highlighting the trade-off.

3.5 Discussion

We present two methods for synchronizing multiple video sequences. The first method automatically tracks a rotating grid pattern and achieves frame-level synchronization. This technique is used for camera synchronization in our markerless garment capture application described in Chapter 6.

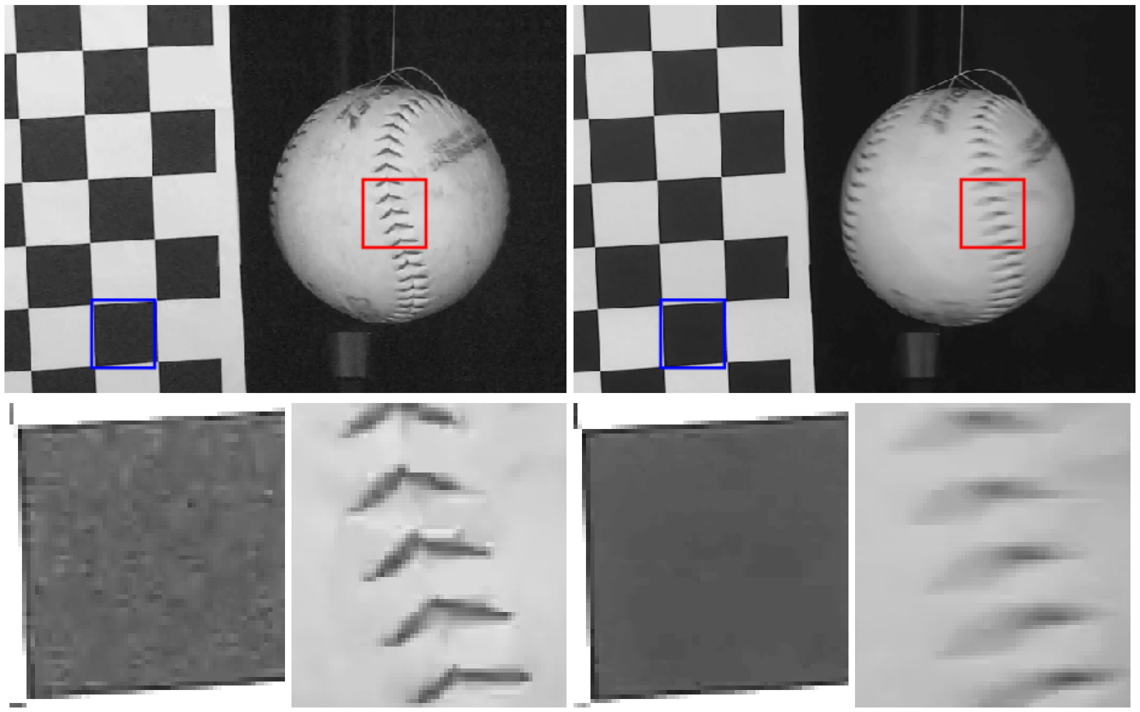


Figure 3.9: Noise vs. motion blur trade-off. A ball with fast constant rotation and a stationary checkerboard. Top left: short virtual exposure. Top right: long virtual exposure. Bottom row: zoom regions show noise in the short virtual exposure and motion blur in the long virtual exposure. The noise in the zoom regions is amplified by a factor of 2 for better visualization.

The second method uses stroboscopic illumination to optically synchronize multiple cameras, and also remove spatio-temporal distortions in the video sequences generated by the rolling shutter. Additionally, since the stroboscope can eliminate motion blur, this method can deal with very fast motion. With a controllable virtual exposure time, we allow a trade-off between motion blur and camera noise. This technique is used to synchronize cameras in our human facial performance capture application described in Chapter 7.

Our approaches for rolling shutter compensation and synchronization enable the use of rolling shutter cameras in a large range of computer vision applications. With these issues resolved, we believe that consumer camcorders are very attractive solutions for computer vision settings due to their low cost, guaranteed frame rate, and easy handling.

Chapter 4

Multi-Camera Calibration

One of the most common problems in computer vision is camera calibration. The process of calibration is to determine the intrinsic and extrinsic parameters of a camera from a number of correspondences between 3D points and their projections onto one or multiple images [Tsai 86, Zhan 00]. Most often this task is accomplished using a calibration plane with a checkerboard or other known marker pattern [Fial 08]. In this Chapter, we focus on the problem of multiple camera calibration, where the relative projection matrices between cameras must be very accurate. This will be very important for the algorithms we will present later, such as multi-view stereo reconstruction, garment motion capture and facial performance capture. For these applications, the quality of the camera calibration has a direct impact on the quality of the results. We will specifically explore binocular camera calibration, and later discuss how our approach can be used in a many-camera setup.

The two most common techniques for camera calibration are those of Tsai [Tsai 86] and Zhang [Zhan 00]. While the method of Tsai has the advantage that it can handle both coplanar and non-coplanar input points, the easiest and most practical approach is to use a calibration grid or checkerboard of coplanar points. When the input points lie on a single plane, it is wise to have multiple input images containing different planar grid orientations in order to ensure a robust calibration, even though Tsai's method can operate on a single input image. Zhang's calibration method strictly enforces these conditions, requiring multiple images of a planar calibration grid. In this Chapter we will employ the method of Zhang, although our algorithm is applicable to other calibration methods, including that of Tsai.

Once multiple calibration images are collected, the calibration process proceeds by finding the projection of known grid points in the images and then solving for the camera parameters that minimize the reprojection error of the detected points. The result is a single set of intrinsic parameters for the entire image sequence, as well as multiple sets of extrinsic parameters, one for each calibration grid location. All of the images are used to compute the intrinsic parameters, but each set of extrinsic parameters is computed from a single image and the corresponding grid location. The problem, as pointed out by Zaharescu et al. [Zaha 06], is to determine which extrinsic parameters to use. When collecting the images, a small number of precisely oriented grid locations could be recorded to ensure that the entire capture space is sampled by points. Alternatively, a short video could be captured where the calibration grid is rotated more or less at random, resulting in hundreds of input images but also covering the entire space. The latter approach is more practical, however there will be many more calibrations to choose from. Each different location of the calibration grid produces different extrinsic parameters for the camera, with varying accuracy depending on the grid orientation, visibility, illumination, noise and a variety of other parameters. The standard approach to determine which grid location to use is to keep the extrinsic parameters that give the lowest reprojection error in the *single* image from which the extrinsics are calculated. This is the only image for which 3D points are known, and thus the best we can hope for in single camera calibration. However, the reprojection error for a single grid location is only guaranteed to be accurate for points that lie on the plane of that grid, and other points off the plane can have a much higher reprojection error. Although multiple grid locations are captured in the sequence, the 3D location of one grid relative to another is usually not available, and so we are unable to compute reprojection errors for points off the plane of each grid.

In this Chapter we show that in a binocular camera setup we can use *all* the grid locations in all images to evaluate *each* potential set of extrinsic parameters and more accurately determine the calibration for a pair of cameras [Brad 10a]. We do this by estimating a reprojection error for the entire volume spanned by the calibration grid over the whole sequence of images, rather than simply a reprojection error for only the points that lie on the calibration plane in one image, as with the single-camera approach described above. Our new approach is partly inspired by the *normalized stereo calibration error* (NSCE) of Weng et al. [Weng 92], which evaluates multi-calibration by measuring the triangulation error of a pair of cameras using known 3D points. However, for any given calibration, we do not know the 3D point locations on any calibration plane except the one that defines the extrinsic parameters, so the NSCE is not applicable. Even though we do not know the 3D locations of points on the other grids, the projection of common points onto the two cameras can be found and we can measure the accuracy of the epipolar constraints for each potential set of extrinsic parameters. This accuracy measure, which we call the *rectification error*, measures the subpixel scanline difference between the projection of common points onto the rectified versions of the two camera images. If the calibration of the two cameras is accurate then any scene point visible by both cameras will project onto the same scanline in the rectified images. Any discrepancy indicates an error in the calibration.

4.1 Camera Calibration Overview

We begin by describing the camera model, a typical planar-based calibration method, and the standard rectification error used to evaluate calibration.

4.1.1 Camera Model

A camera model consists of a set of intrinsic parameters, which define how the camera forms an image, and a set of extrinsic parameters, which define the position and orientation of the camera in the world. The intrinsic parameters include the focal length in pixels (fx, fy), the principal point (px, py), and a skew factor, s (which is often ignored and set to zero [Sun 06]). These parameters form the camera matrix K, defined as

$$K = \begin{bmatrix} fx & s & px \\ 0 & fy & py \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.1}$$

The extrinsic parameters include a rotation matrix $R \in \mathbb{R}^{3\times 3}$ and a translation vector $t = [\ tx \quad ty \quad tz\ ]^\top$, which relate the world coordinate frame to the camera coordinate frame. The full calibration forms a pro-

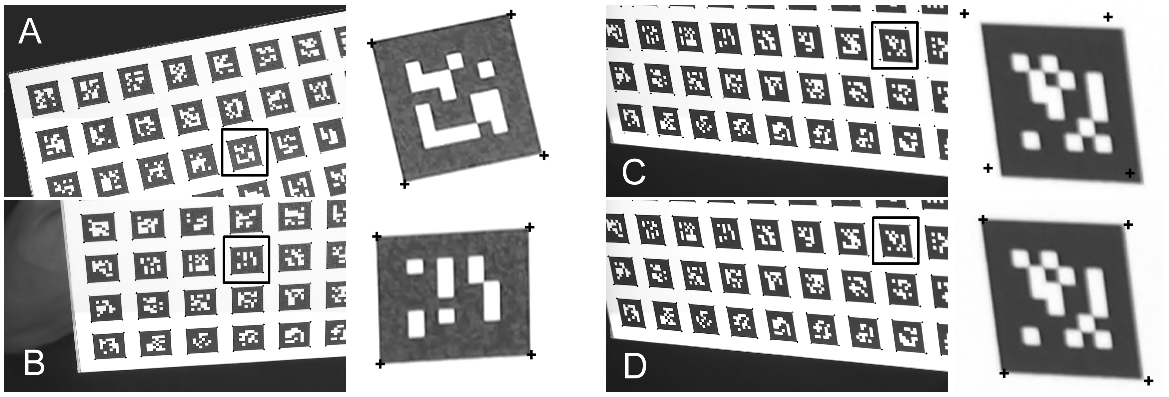


Figure 4.1: Visualizing reprojection errors for two different calibrations of a single camera. The top row is calibrated with extrinsic parameters from grid A, while the bottom row is calibrated from grid B (intrinsic parameters are the same for both). The points on the common grid in C and D lie off the planes of both A and B. The calibration from grid B is more accurate than the one from grid A, as shown in the zoom images on the right.

jection matrix $P \in \mathbb{R}^{3\times4}$, defined as

$$P = K \cdot [R|t], \tag{4.2}$$

which maps world points to camera pixels.

The calibration parameters also include radial distortion coefficients to correct lens aberration. From this point on we will assume that radial distortion has been corrected for all camera images.

4.1.2 Calibration

We will focus on the common calibration technique of Zhang [Zhan 00]. This method is widely used and an implementation is readily available in the OpenCV library [Open]. The input is a set of several views of a known calibration grid, where every view is described by several world-to-2D point correspondences. Typically 20-30 views are required. The world points are defined by the calibration plane ($z = 0$), and an elegant marker pattern and corner detection scheme has been proposed by Fiala and Shu [Fial 08] to detect the 2D correspondences. The calibration method then solves for all the camera parameters such that the reprojection error of the points is minimized. The result is a single K matrix and multiple R_i and t_i transformations, one for each view of the calibration grid. Since all grid locations are used to estimate K we assume the intrinsic parameters are computed robustly. This assumption is common for multi-camera setups [Zaha 06], and in practice we have observed this to be true. However, since an R_i and t_i are determined for each input image, the accuracy of each transformation depends on a single grid location and how well the grid pattern was detected in the single image. Therefore, some transformations can be more accurate than others. We illustrate this effect in Figure 4.1. Here we choose two different grid locations, A and

B, each resulting in a different projection matrix $P_A = K \cdot [R_A|t_A]$ and $P_B = K \cdot [R_B|t_B]$. Both P_A and P_B appear to be accurate when reprojecting the *planar* grid points onto the image, as we see in the first zoom images. However, when reprojecting points that are not on the calibration plane used to compute the extrinsic parameters (e.g. the plane in Figure 4.1, C and D), we see that P_A (top row) is less accurate than P_B (bottom row).

The problem is to determine which of the multiple R_i and t_i transformations is the most accurate. The common approach is to select the transformation with the lowest reprojection error for the single calibration grid used to compute the transformation, as we describe next.

4.1.3 Reprojection Error

Let $P_i = K \cdot [R_i|t_i]$ be the projection matrix of camera c for calibration grid view i. Assume we have detected k grid points x_j in the image, corresponding to world points X_j. Then the reprojection error for image i is

$$e^c_{rep}[i] = \frac{1}{k} \sum_{j=1}^{k} \|P_i(X_j) - x_j\|. \tag{4.3}$$

The reprojection error has been widely used as the main tool for evaluating camera calibration, either in the form presented above [Fial 08, Ouya 05, Zaha 06], or in a normalized form [Sun 06, Weng 92]. This is because a low reprojection error indicates an accurate projection matrix, at least for the points on the plane that were used to compute the projection matrix. The problem is that the reprojection error may increase for 3D points off the plane, as we saw in Figure 4.1. The worst case is when the planar grid is perpendicular to the optical axis of the camera, and the calibration may only be accurate for essentially a single depth. The reprojection error would be more accurate if there were additional 3D points available, off the plane of the calibration grid, for which we had corresponding detected 2D pixels. The image sequence does in fact contain many different grid locations for which corner points are detected, however the corresponding 3D locations of those points are not known. Figure 4.2 (a) illustrates the problem. The camera is calibrated with respect to calibration grid i_1, which defines the world coordinate system. Unfortunately, the 3D location of point Q on grid i_2 is not available, and so Equation 4.3 cannot be applied. For this reason, the reprojection error can only be evaluated on plane i_1. However, if two cameras observe the same calibration grid sequence (such as the case of binocular stereo), then we propose a pair-wise calibration algorithm using the rectification error.

4.2 Rectification Error

When two cameras observe the same sequence of calibration grid locations, all grids can be used to evaluate the calibration accuracy for each individual set of extrinsic parameters. As we have seen, the standard reprojection error in Equation 4.3 cannot be applied to points off the main grid (grid i_1 in Figure 4.2). However, if a point, Q, on some other grid is visible in both cameras then epipolar constraints tells us that the projection of Q onto the rectified versions of the left and right images should lie on the same scanline if the calibration of the cameras is accurate (see Figure 4.2 (b) and (c)). This property is independent of the

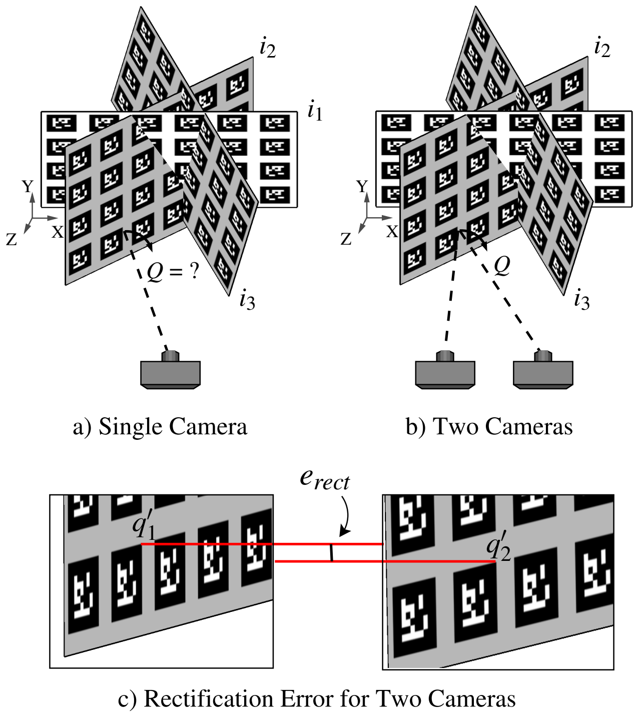


Figure 4.2: Evaluating the extrinsic parameters computed using calibration grid i_1. a) With a single camera we cannot compute reprojection errors for points on other planes. b) With two cameras, although the 3D location of Q is not known, its projection onto the two cameras can be found. c) The rectification error is the scanline difference between the projection of Q onto the rectified versions of the two cameras.

3D location of Q, and thus we are able to use all detected points from all grid locations that are common in both views.

From the above observation, we form a measure of *rectification error* for two cameras c_1 and c_2, and calibration grid view i as follows. For each calibration grid, let $q_1^k = (u_1^k, v_1^k)$ be the k^{th} detected grid point on the image plane of c_1 corresponding to unknown 3D point Q^k. This point Q^k is also detected in c_2 as $q_2^k = (u_2^k, v_2^k)$. For $c \in \{1,2\}$, we denote $q_c^k[0]$ to refer to u_c^k and $q_c^k[1]$ to refer to v_c^k. Then,

$$e_{rect}^{c_1}[i] = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{M_j}\sum_{k=1}^{M_j}\left|(T_i^{c_1}q_1^k)[1] - (T_i^{c_2}q_2^k)[1]\right|\right), \tag{4.4}$$

where $T_i^{c_1}$ is the rectifying transformation for camera c_1 using calibration i, and $T_i^{c_2}$ is defined similarly for camera c_2. N is the total number of grid positions in the sequence, and M_j is the number of grid points that are commonly detected in both camera views for grid position j. Note that the rectification error is symmetric, so $e_{rect}^{c_2} = e_{rect}^{c_1}$. We compute the rectifying transformations using the method of Fusiello et al. [Fusi 00]. We illustrate the rectification error for a particular point Q in Figure 4.2 (c), where

$$q'_1 = T_i^{c_1} q_1,$$
$$q'_2 = T_i^{c_2} q_2.$$

We can now use this rectification error measure when computing binocular camera calibrations. In our experience, this measure yields more accurate calibrations than the standard method of using the reprojection error. In fact, the calibration grid *A* in Figure 4.1 was the one with the lowest reprojection error, and grid *B* had the lowest rectification error. As we saw in that figure, the rectification error determined a more accurate calibration.

4.3 Experimental Results

We demonstrate the quality of the rectification error by calibrating a multi-camera setup of 14 cameras, arranged as seven binocular pairs. We show qualitative results by performing stereo reconstruction of a static object, and quantitatively show that the rectification error was a better tool for evaluating binocular camera calibrations than the reprojection error in all seven binocular pairs.

Our static object is a human head model made from styrofoam, which we have painted in order to give it a high-frequency surface texture to aid the stereo reconstruction. We use an ARTag calibration grid [Fial 08] and the method of Zhang [Zhan 00] to compute one set of intrinsic parameters and multiple sets of extrinsic parameters for each camera. In all of our experiments we record a calibration video for each camera, where the calibration grid is rotated and translated throughout the capture volume, resulting in over one thousand planar grid orientations. As we have discussed, the problem lies in choosing which extrinsic parameters to use. We will show in Section 4.3.4 that this choice is critical to the quality of the calibration and the resulting stereo reconstruction.

In this Chapter we are in fact advocating two principles. One, cameras should be calibrated in binocular pairs, and two, cameras should be calibrated using the rectification error. In order to show the importance of combining these principles we have performed three experiments. In the first experiment we find the best extrinsic parameters for all cameras globally using reprojection error. This is what we refer to as the standard approach, which we use as a baseline for comparison. In the second experiment we calibrate the cameras in pairs, but still use the reprojection error. This experiment will show that simply calibrating in pairs is not sufficient, if the rectification error measure is not used. Finally, in the third experiment we calibrate in pairs and use the rectification error. As we will see, the calibration quality achieved in the third experiment is consistently superior to that of the first two experiments. Additionally, we will see that the rectification error is directly proportional to the quality of the stereo reconstruction for each binocular pair, no-matter which method is used to choose the extrinsic parameters, unlike the reprojection error which can be misleading.

4.3.1 Exp. 1 - Best Global Reprojection Error

Let S be the set of all calibration grids visible in *every* camera view. We choose the single grid location that yields the lowest average reprojection error among all N_c cameras. Specifically, the grid i that minimizes

$$\min_{i \in S} \sum_{c=1}^{N_c} (e^c_{rep}[i]). \tag{4.5}$$

The benefit of this approach is that all cameras are calibrated to the same world coordinate system. However the drawback is that some cameras will be calibrated better than others (see Figure 4.3 and Section 4.3.4). In practice, depending on the camera setup it may also be difficult to find calibration grids that are visible in all camera views.

4.3.2 Exp. 2 - Best Pair-wise Reprojection Error

Let S be the set of all calibration grids visible by a specific *pair* of cameras, c_1 and c_2. We choose the single calibration grid that yields the lowest average reprojection error for those two cameras. Specifically, the grid i that minimizes

$$\min_{i \in S} \sum_{c \in [c_1, c_2]} (e^c_{rep}[i]). \tag{4.6}$$

The benefit of this approach is that there will be more grids to choose from, resulting in lower reprojection errors (see Table 4.1 and Section 4.3.4). The drawback is that the low reprojection errors are misleading, as some calibrations are still not very accurate (Figure 4.3). Also, the cameras will not be calibrated in the same world coordinate system, so combining stereo results from different pairs is no-longer trivial.

4.3.3 Exp. 3 - Best Pair-wise Rectification Error

Let S be the set of all calibration grids visible by a specific *pair* of cameras, c_1 and c_2. We choose the single calibration grid that yields the lowest average rectification error for those two cameras. Specifically, the grid i that minimizes

$$\min_{i \in S} \sum_{c \in [c_1, c_2]} (e^c_{rect}[i]). \tag{4.7}$$

This approach has the same benefit as experiment two, in that there are more grids to choose from than in experiment one, and the same drawback in that the cameras will not be calibrated in the same coordinate system. However, this approach yields the most accurate calibrations (again, see Figure 4.3).

4.3.4 Stereo Reconstruction Analysis

We analyze the quality of the calibrations in each experiment by performing stereo reconstruction of our styrofoam head. We expect that the accuracy of each calibration will be reflected in the number of outliers in the corresponding depth map. We use the reconstruction method that we will describe later in Chapter 5, although other methods could equally be employed for this evaluation. In this reconstruction algorithm, many depth outliers are automatically rejected by thresholding on the correlation score, and also through a simple filtering post-process. In effect, the quality of the calibration is related to the completeness of the depth map.

Our cameras are Sony HDR-SR7 camcorders, which capture high-definition video, although we only reconstruct a single frame. The cameras are placed very close to the reconstruction object and zoomed in to see the painted surface details. This type of close-range setup is very challenging to calibrate as even the slightest calibration error results in incomplete or inaccurate reconstruction results.

Qualitative Analysis. Figure 4.3 shows the depth maps for each experiment. As we can see, the results from experiment one tend to be rather poor, with only one camera pair (8-9) producing a valid and mostly complete depth map. We can also see that depth maps for some pairs are only accurate at a single depth (i.e. pairs 2-3 and 10-11), which indicates that the chosen calibration grid was likely nearly-perpendicular to the optical axes of the cameras, resulting in a low reprojection error but also low calibration accuracy. We see that experiment two sometimes produces better depth maps than experiment one, although sometimes they are worse. This indicates that calibrating in pairs alone is not sufficient. The results of experiment three produce the most complete depth maps for all binocular pairs, indicating the importance of calibrating in pairs and using the rectification error.

Quantitative Analysis. Table 4.1 shows reprojection errors (Equation 4.3) *and* rectification errors (Equation 4.4) for the resulting calibration of every camera from each experiment, measured in pixels. This data corresponds to the stereo results in Figure 4.3. Using this table we can quantitatively analyze the difference between the two measures. As expected, the reprojection errors for experiment two are lower than for experiments one and three, although this has no correlation with the quality of the stereo reconstruction and so the reprojection error is not indicative of calibration accuracy. On the other hand, the rectification error varies between experiments one and two, and a lower error value directly correlates with better stereo reconstruction (refer again to Figure 4.3). As expected, experiment three produced the lowest rectification errors and thus the best reconstruction results. It is clear from comparing Table 4.1 to the depth images in Figure 4.3 that our proposed rectification error has accurately measured the quality of all seven binocular camera calibrations.

Merging Stereo Pairs. If we wish to generate a 3D model from the stereo reconstructions, the cameras must be aligned in the same world coordinates. For experiments two and three, the resulting pair-wise calibrations need to be transformed rigidly into a global coordinate frame. This can be achieved in a number of ways. First, experiment one could be performed to establish the world coordinate frame, and then each pair could be commonly transformed into the global coordinates. In this approach, the same transformation would be applied to each camera in a pair, keeping the relative transformations between the two cameras fixed and therefore very accurate. The drawback is that inter-pair alignment would be less accurate than the

	Reprojection Error			**Rectification Error**		
Cam	Exp. 1	Exp. 2	Exp. 3	Exp. 1	Exp. 2	Exp. 3
0	1.57	0.66	1.45	1.20	1.21	1.01
1	2.04	1.08	1.84	1.20	1.21	1.01
2	1.77	0.94	2.08	12.64	1.75	1.08
3	1.80	1.02	2.19	12.64	1.75	1.08
4	1.75	0.54	1.40	4.48	1.28	0.75
5	0.83	0.56	1.62	4.48	1.28	0.75
6	1.89	0.85	2.10	2.09	1.81	0.78
7	1.76	1.18	1.19	2.09	1.81	0.78
8	1.96	0.93	1.83	1.43	3.33	1.02
9	2.65	1.51	2.22	1.43	3.33	1.02
10	2.03	0.90	1.33	14.93	7.76	1.22
11	1.64	1.16	1.89	14.93	7.76	1.22
12	1.63	1.04	3.42	3.05	3.59	1.49
13	1.56	1.12	4.60	3.05	3.59	1.49

Table 4.1: Comparing reprojection and rectification errors for the three experiments. Note that the reprojection errors are quite similar and non-indicative of the stereo results in Figure 4.3, although the rectification errors can be quite different and they directly reflect the quality of the stereo reconstruction.

pair-wise alignment. A second option is to perform pair-wise calibration between non-stereo pairs and *build up* a global calibration by sequentially adding pairs. For example, if the stereo pairs are [c_1,c_2], [c_3,c_4], and [c_5,c_6], then pair-wise calibration could be performed between cameras c_2 and c_3, thus aligning the first two pairs, and then between c_4 and c_5, thus aligning the last pair with the first two. Finally, a third option is to complete the binocular stereo reconstruction in pairs and then align the different depth maps or 3D surfaces using a rigid alignment technique such as ICP [Besl 92]. In practice, this is how we generate the final 3D model in Figure 4.4. This result, using calibrations from experiment three, shows the high accuracy of calibrations computed using the rectification error.

4.4 Discussion

We propose a new technique for calibrating binocular cameras using a pair-wise rectification error. In our experience, this technique can be used to significantly improve stereo reconstruction results, as compared to using the standard reprojection error for calibration.

In order for a camera pair to be considered as a binocular pair for calibration, the only requirement is that the two cameras observe the same sequence of calibration grid images. This means that the cameras should be placed fairly close together with significant overlap in their views. However, this condition is already met in most binocular camera setups, if the cameras are to be used for applications such as stereo reconstruction or novel view interpolation.

We validate our approach by calibrating seven binocular pairs using our rectification error measure, and further demonstrate how the cameras can be combined into an accurate multi-view reconstruction setup. We believe our method can benefit a number of applications that make use of camera arrays, where binocular pairs are available.

Camera calibration is required for most of the techniques presented in this thesis. The standard reprojection error approach is employed in Chapters 5 and 6, where we define our techniques for multi-view stereo reconstruction and markerless garment capture. Our new rectification error approach was specifically designed for our facial performance capture algorithm, described in Chapter 7.

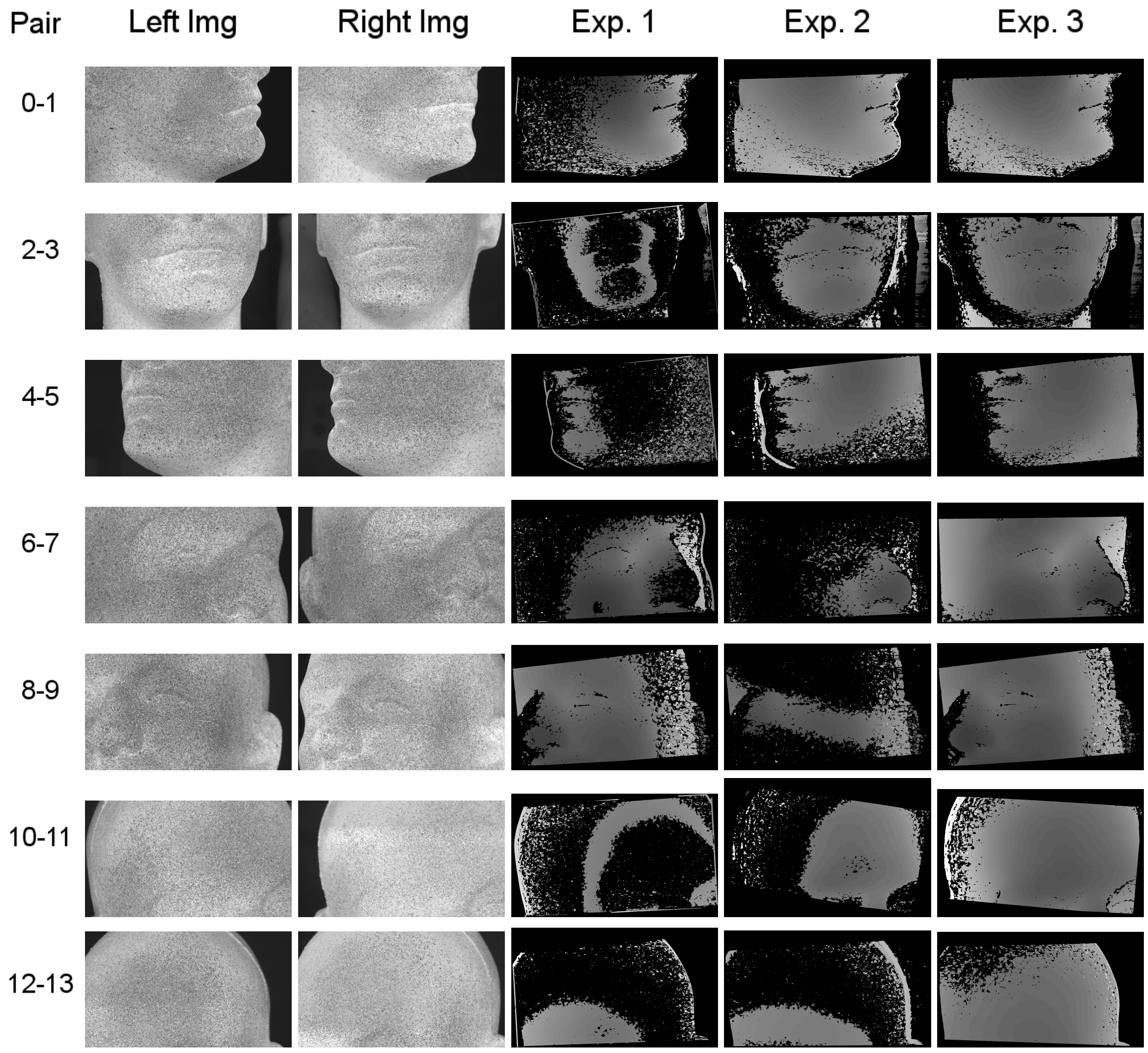


Figure 4.3: Binocular stereo results for a styrofoam head using the calibrations from the three experiments. Each row is a separate camera pair. From left to right we show the two camera views, then the depth maps from experiments one, two and three respectively. Corresponding quantitative results are shown in Table 4.1.

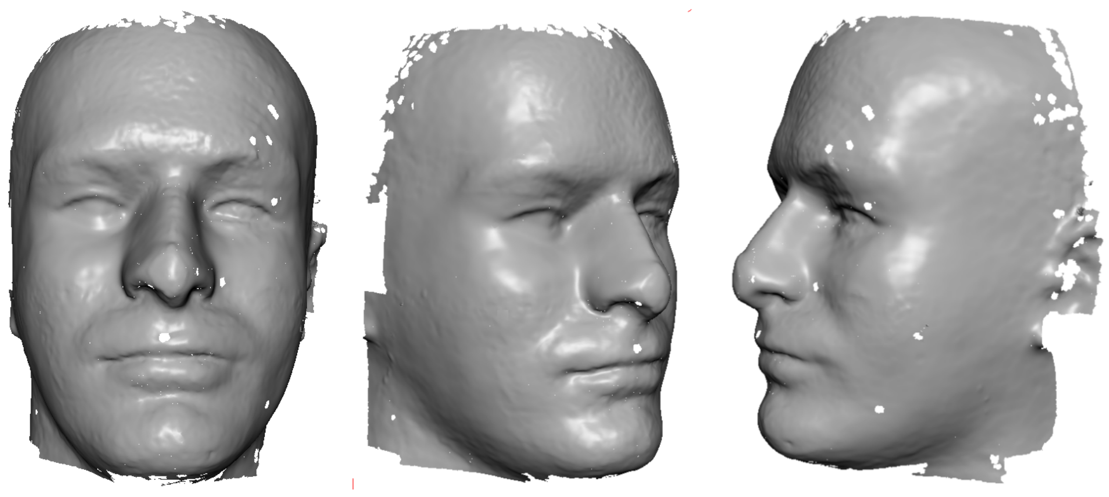


Figure 4.4: Combined multi-view 3D result from Experiment three.

Chapter 5

Multi-View Stereo Reconstruction

Multi-view stereo (MVS) reconstruction is the process of reconstructing a 3D model of an object or scene from multiple images captured from different viewpoints. MVS algorithms have seen a surge of interest in recent years. Much progress has been made, both in terms of precision and in terms of performance, although methods with a combination of both high efficiency and high quality remain elusive.

In our work, we revisit one of the most simple approaches to multi-view reconstruction, namely that of merging multiple depth maps estimated using binocular stereo. We show that, with careful design, this simple approach can yield some of the highest quality reconstructions of any multi-view stereo algorithm, while remaining highly scalable and offering excellent performance [Brad 08a].

MVS algorithms that rely on depth maps can typically be divided into two separate processes. First, a depth map is computed for each viewpoint. Secondly, the depth maps are merged to create a 3D model, and a triangle mesh is generated. The contributions of this thesis include a new method for depth map computation, which is combined with a state-of-the-art surface meshing technique to form a novel end-to-end MVS algorithm. The surface meshing algorithm reliably removes outliers and high frequency noise, enabling us to use a simple and fast binocular stereo algorithm that produces very dense, but potentially unreliable points. In particular, the MVS algorithm has the following characteristics:

- The binocular stereo algorithm makes use of *scaled window matching* to improve the density and precision of depth estimates. Filtering and constraints are used to further improve the robustness.
- The surface reconstruction algorithm uses adaptive, point-based filtering and outlier removal of the joint depth information from multiple views. This point-based formulation of the filtering process avoids resampling and quantization artifacts of other approaches.
- The meshing algorithm works in a lower dimensional space, resulting in high performance and well-shaped triangles.

All geometry processing algorithms work only on local neighborhoods, and have a complexity of $O(k \log k)$, where k is the number of points processed by the respective algorithm. Since the binocular stereo part is linear in the number of images, the complete algorithm remains very scalable despite the high quality reconstructions it produces.

5.1 Algorithm Overview

Our multi-view reconstruction algorithm takes as input a set of calibrated images, captured from different viewpoints around the object to be reconstructed. We assume that a segmentation of the object from the background is provided, so that the visual hull is represented as a set of silhouette images. As mentioned in the introduction, our MVS method is performed in two steps, binocular stereo on image pairs, followed by surface reconstruction. Figure 5.1 shows a diagram of the individual stages.

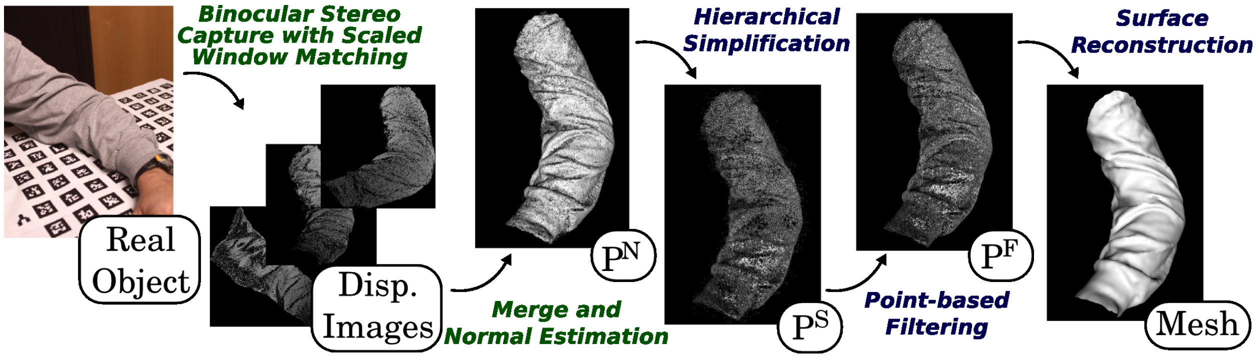

Binocular Stereo Capture with Scaled Window Matching
Real Object
Disp. Images
Merge and Normal Estimation
P^N
Hierarchical Simplification
P^S
Point-based Filtering
P^F
Surface Reconstruction
Mesh

Figure 5.1: Acquisition pipeline: the binocular stereo algorithm generates a 3D point cloud that is subsequently processed and converted to a triangle mesh.

The binocular stereo part of our algorithm creates depth maps from pairs of adjacent viewpoints. After rectifying the image pairs, we observe that the difference in projection between the views causes distortions of the comparison windows. We compensate for the most prominent distortions of this kind by employing a *scaled-window matching* technique, which improves the quality especially in high curvature regions and for sparse viewpoints (i.e. large baselines). The depth images from the binocular stereo pairs are converted to 3D points and merged into a single dense point cloud.

The second part of the algorithm is not a contribution of this thesis, however we will provide a short overview for the sake of completeness. The goal is to reconstruct a triangular mesh from the initial point cloud. This is accomplished with a three-step approach that includes **downsampling**, **de-noising** and **meshing**.

In the following sections, we elaborate on the two main steps of our algorithm.

5.2 Stereo Matching

The first step of our MVS algorithm involves estimating depth maps for each camera view using binocular stereo with one of the neighboring cameras as a reference view. For each image pair, the views are first rectified [Fusi 00], such that corresponding scanlines in the primary and reference view correspond to epipolar lines. For each pixel in the rectified primary view we then find the closest matching pixel on the corresponding epipolar (scan)line in the rectified reference view. Specifically, we assume brightness constancy, and use *normalized cross correlation* (NCC) [Fors 02] on square pixel regions as a metric for the best match:

$$NCC(v_0, v_1) = \frac{\sum_{j=1}^{N^2}(v_0(j) - \overline{v_0}) \cdot (v_1(j) - \overline{v_1})}{\sqrt{\sum_{j=1}^{N^2}(v_0(j) - \overline{v_0})^2 \cdot \sum_{j=1}^{N^2}(v_1(j) - \overline{v_1})^2}}, \tag{5.1}$$

where v_0 and v_1 are local neighborhoods of size $N \times N$ in the primary and the reference view, and $\overline{v_0}$ and $\overline{v_1}$ represent the intensity averages over the same neighborhoods. We ignore very bad matches by thresholding the NCC metric to the range of [0.5 - 1]. Because some uncertainty remains in the resulting matches, we further remove outliers with simple constraints and filtering, as discussed below.

Scaled Window Matching. As observed previously [Faug 98, Ogal 05, Ogal 07, Goes 07, Gall 07], the difference in projection between the primary and reference view can distort the matching window, such that a square window in the primary view is non-square in the reference view. In our rectified setup, the vertical scale is identical in both views, since corresponding scanlines represent corresponding epipolar lines. The horizontal scale, however, depends on the 3D surface orientation.

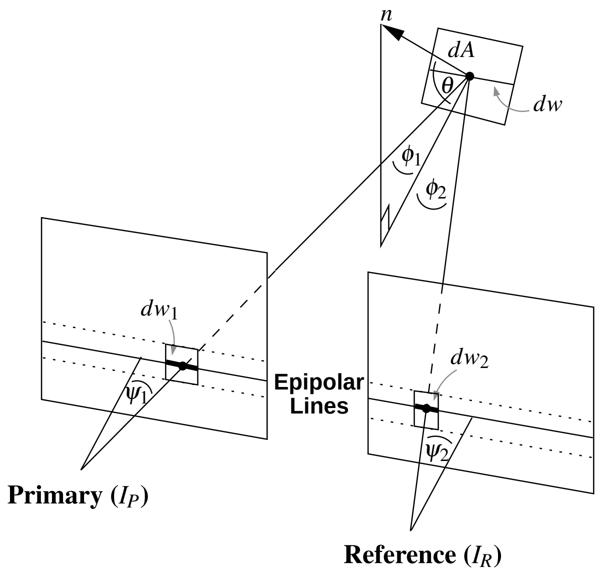


Figure 5.2: Corresponding pixel windows in the primary and reference view can have different width depending on the orientation of the 3D surface.

A first-order differential model of this effect can be described as follows. Consider a differential surface patch dA in the tangent plane of an object point with normal n (see Figure 5.2). Let dw denote the (differential) length of the intersection of dA and the epipolar plane. This line segment dw projects to a segment $dw_1 = (dw \cdot \cos\phi_1)/(\cos\psi_1 \cdot z)$ on the image plane of the primary view (I_P), and to a segment $dw_2 = (dw \cdot \cos\phi_2)/(\cos\psi_2 \cdot z)$ on the image plane of the reference view (I_R). Here, z is the perspective depth, the ϕ_i correspond to the angles between the viewing rays and the projection of n into the epipolar plane, and the ψ_i correspond to the incident angles of the viewing rays on the respective image planes (see Figure 5.2). As a first-order effect, we therefore expect that any window in the primary view will appear scaled horizontally by a factor of

$$\frac{\cos\psi_1}{\cos\psi_2}\cdot\frac{\cos\phi_2}{\cos\phi_1}. \tag{5.2}$$

This horizontal scaling is particularly pronounced for wide camera baselines, i.e. "sparse" MVS setups with few cameras, as well as horizontally slanted surfaces, common when imaging high curvature geometry. To improve robustness in those settings, we employ a similar approach to Ogale and Aloimonos [Ogal 05, Ogal 07], by performing window matching between the rectified primary view and a number of versions of the rectified reference view with different horizontal scale factors. The best match is then defined as the largest NCC of any $N \times N$ neighborhood on the epipolar line in any of the differently scaled reference views.

We experimentally found that scale factors of $\frac{1}{\sqrt{2}}$, 1, and $\sqrt{2}$ yield excellent results for most datasets. Larger scale factors do not generate a significant number of reliable matches since view dependent effects start to dominate. For strongly asymmetric camera configurations, we can first pre-scale the reference image with a global average of $\cos\psi_1/\cos\psi_2$, and then use window matching at the three scales mentioned above.

The above first order approximation is valid for small window sizes and surface orientations where the normal n is close to parallel to the epipolar plane. Larger angles of θ introduce a shear along the horizontal direction in addition to the horizontal scaling. We analyzed this issue, and determined that the shear is negligible for moderate camera baselines (up to 45° between viewing rays), and θ up to about 60°. We therefore decided against also using sheared versions of the reference view for the block matching.

Our scaled-window technique is related to the plane-sweeping stereo method of Gallup et al. [Gall 07], in which a small number of oriented planes are swept through the scene to reconstruct surfaces that are slanted with respect to the cameras. In their work, each plane-sweep is intended to reconstruct surfaces having a particular normal, and the sweep orientations must be determined ahead of time. Their method is particularly useful for scenes with a small number of dominant normal directions, such as urban environments where the ground plane and building facades can be identified. However, time-varying deformable surfaces are much more complex, with largely varying normal fields. For these surfaces, our image-based scaled-window matching technique is better-suited.

Subpixel Optimization. The result of the window matching is a highly quantized disparity image for each viewpoint. We propose two methods for computing subpixel disparity values. The first technique is to explicitly perform a local search at the subpixel level. For each pixel (x_P, y_P) and its matched pixel (x_R, y_R) we perform correlation at a number of linearly interpolated positions between $(x_R - 1, y_R)$ and $(x_R + 1, y_R)$, that is, between the previous and next pixel in the scanline. With $2n$ additional correlation evaluations per pixel we achieve nominal subpixel precision of $1/n$ pixels. The calculations can be optimized by pre-computing an interpolated version of I_R which is $n \cdot width(I_R) \times height(I_R)$. In our experiments, we found that $n = 10$ gives us good precision; larger values have a diminishing return. This subpixel optimization technique was used when generating all the results in this chapter, including the benchmark evaluation results (see Table 5.1), and also for the garment capture results in Chapter 6.

Our second approach, which was used for the face capture results in Chapter 7, is inspired by optical flow.

Once the integer-valued disparity is found, the subpixel offset can be computed using one-dimensional Lucas-Kanade optical flow [Boug 99] along the scanline. The initial window match yields

$$\begin{aligned} I_P(x_P, y_P) &\approx I_R(x_R, y_R) && (5.3)\\ &= I_R(x_R + \delta x, y_R), && (5.4) \end{aligned}$$

where δx is the subpixel offset we seek. Using Taylor series we see that

$$I_P(x_P, y_P) = I_R(x_R, y_R) + \frac{\partial I_R}{\partial x}\delta x, \tag{5.5}$$

ignoring higher order terms. Here, $\frac{\partial I_R}{\partial x}$ is the horizontal image gradient at (x_R, y_R). Then the offset can be computed as

$$\delta x = \frac{I_P(x_P, y_P) - I_R(x_R, y_R)}{\frac{\partial I_R}{\partial x}}. \tag{5.6}$$

We use a Lucas-Kanade approach to solve for δx in local windows with Gaussian weights centered around (x_P, y_P) and (x_R, y_R), yielding a one-dimensional linear least-squares problem.

Constraints. In order to improve accuracy and reduce the number of outliers in the point cloud, we incorporate two constraints when performing window matching. Like many previous algorithms, we use the visual hull of the reconstruction object as a first constraint. We then use a disparity ordering constraint [Bake 81] for scenes where this assumption is valid, including all the datasets in this Chapter.

Filtering. As a final step, we perform some initial filtering of the depth maps. Large outliers in the disparity images are removed with a median-rejection filter. If a disparity value is sufficiently different from the median of its local neighbourhood, then it is rejected. To remove some high-frequency noise already in image space, disparity images are then smoothed with a trilateral filter, which is a standard bilateral filter [Toma 98] weighted by the NCC value that corresponds to the best match for each disparity pixel. The NCC value is a good indication of the confidence of each pixel, and we incorporate this into the smoothing function so that points of low confidence have less influence on their neighbours. Note that this is only preliminary filtering, as most of the filtering is performed after merging all the depth maps together, where redundant information from different views can be used to achieve more accurate filtering.

At the end of the binocular stereo stage, all depth maps are backprojected into 3D, and merged into a single point cloud.

5.3 Surface Reconstruction

The next step of the algorithm uses efficient local point-processing techniques to generate a triangle mesh from the dense point cloud. As we mentioned earlier, this part of the algorithm is not a contribution of this thesis, and so we provide only a short overview. The technique consists of three steps:

1. **Downsampling**: The point cloud is usually much denser than required for reproducing the amount of actual detail present in the data. Our first step is thus to downsample the data using *hierarchical vertex clustering* [Boub 06].
2. **De-noising**: The simplified point cloud remains noisy. While some methods integrate the noise removal in the meshing algorithm [Ohta 03, Kazh 06], we believe that this important data modification must be controlled explicitly, prior to any decision concerning the mesh connectivity. De-noising is accomplished using recent point-based filtering tools.
3. **Meshing**: The final step is to generate a triangle mesh without introducing excessive smoothing. We build on *lower dimensional triangulation* methods [Boub 05, Gopi 00], which are fast and run locally, ensuring scalability and good memory-computational complexity.

5.4 Results

We demonstrate our multi-view stereo algorithm with a number of reconstructions. First, we quantitatively evaluate our results using the *Middlebury* datasets [mvie, Seit 06]. The data consists of two objects, a dinosaur and a temple (see Figures 5.3 and 5.4), and three different sets of input images for each one, with viewpoints forming a sparse ring, a full ring, and a full hemisphere around the object.

Our method is well-suited for viewpoints arranged in a ring setup since the grouping of cameras into binocular pairs is straightforward. Thus we perform our evaluation on the *dinoRing* (48 images), *dinoSparseRing* (16 images), *templeRing* (47 images), and the *templeSparseRing* (16 images) inputs. The quantitative results of our algorithm are shown in Table 5.1. For each of the datasets our method is compared to a number of the state-of-the-art techniques. Algorithms are evaluated on the accuracy (Acc) and completeness (Cmp) of the final result with respect to a ground truth model, as well as processing time (Time). We highlight the best performing algorithm for each metric. Our technique proves to be the most accurate for the sparse-viewpoint datasets. Additionally, ours is the most efficient among non-GPU methods for all four datasets. For reference, we include a short list of methods that make use of the GPU. These results tend to be of much lower quality. Note that algorithms published after our technique are marked with an asterisk. We show our reconstruction of the dinoSparse dataset in Figure 5.3, including an image of the points generated by the stereo matching step. Note that it would be possible to use our method in camera setups other than rings. For instance, a full spherical setup could be handled by computing a Delaunay triangulation of the camera positions, and then computing a binocular image for each edge in that triangulation. This approach would still scale linearly in the number of cameras, since the number of edges in a Delaunay triangulation is linear in the number of vertices.

Following the visualization style of Gallup et al. [Gall 07], we highlight the effectiveness of the scaled-window matching technique (described in Section 5.2) by re-coloring two of the recovered disparity images

	dinoSparseRing			dinoRing			templeSparseRing			templeRing		
Method	Acc	Cmp	Time	Acc	Cmp	Time	Acc	Cmp	Time	Acc	Cmp	Time
Our technique	0.38	94.7	$\boxed{7}$	0.39	97.6	$\boxed{23}$	$\boxed{0.48}$	93.7	$\boxed{4}$	0.57	98.1	$\boxed{11}$
*Campbell [Camp 08]							0.53	98.6	22	0.48	99.4	59
Furukawa [Furu 07]	0.42	$\boxed{99.2}$	224	0.33	99.6	544	0.62	99.2	213	0.55	99.1	363
*Furukawa [Furu 09b]	$\boxed{0.37}$	$\boxed{99.2}$	151	$\boxed{0.28}$	$\boxed{99.8}$	300	0.63	$\boxed{99.3}$	128	0.47	99.6	211
Gargallo [Garg 07]	0.76	90.7	18	0.60	92.9	35	1.05	81.9	35	0.88	84.3	35
Goesele [Goes 06]	0.56	26.0	843	0.46	57.8	2516	0.87	56.6	687	0.61	86.2	2040
Hernandez [Este 04]	0.60	98.5	106	0.45	97.9	126	0.75	95.3	130	0.52	99.5	120
*Jancosek [Janc 09]	0.66	74.9	23	0.71	76.6	37	0.59	74.9	6	0.7	78.9	17
*Liu [Liu 09a]	0.51	98.7	20				0.65	96.9	16			
Strecha [Stre 06]	1.41	91.5	15	1.21	92.4	146	1.05	94.1	6	0.86	97.6	57
Zaharescu [Zaha 07]	0.45	$\boxed{99.2}$	20	0.42	98.6	42	0.78	95.8	25	0.55	99.2	59
GPU Methods												
*Hiep [Hiep 09]				0.53	99.7	1				$\boxed{0.45}$	$\boxed{99.8}$	1
Merrell_stab [Merr 07]				0.73	73.1	0.5				0.76	85.2	0.4
Merrell_conf [Merr 07]				0.84	83.1	0.3				0.83	88.0	0.3
Sormann [Sorm 07]				0.81	95.2	5				0.69	97.2	5
Zach [Zach 07]				0.67	98.0	7				0.58	99.0	7

Table 5.1: Results for the Middlebury Datasets. The top performing algorithm in each category is highlighted. *Accuracy* is measured in millimeters, *Completeness* as a percentage of the ground truth model, and *Normalized Running Time* is in minutes. Algorithms marked with an asterisk (*) were published after our technique.

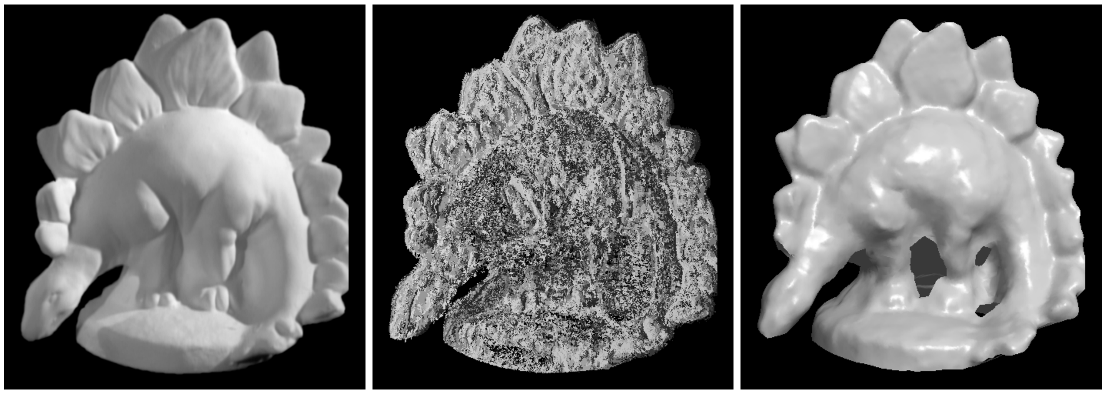


Figure 5.3: Dino reconstruction from sparse (16) viewpoints. Left: one input image (640x480 resolution). Middle: 944,852 points generated by the binocular stereo stage. Right: reconstructed model (186,254 triangles).

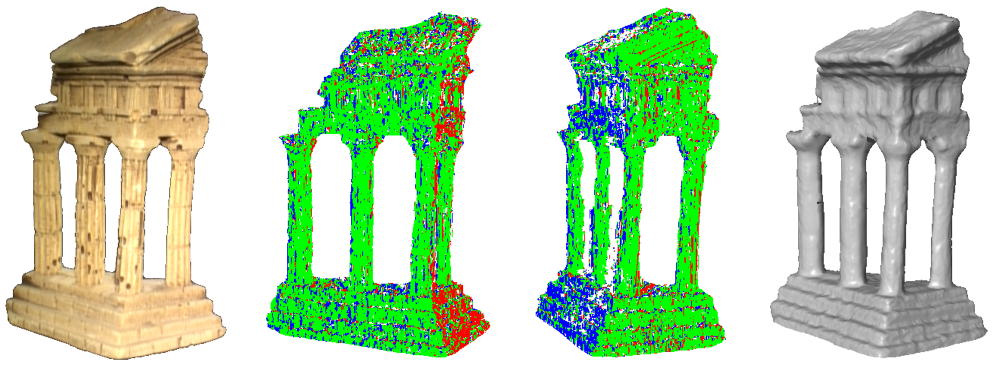


Figure 5.4: Scaled window matching on the templeRing dataset. Left: one of 47 input images (640x480 resolution). Middle: two disparity images re-colored to show which horizontal window scale was used for matching. $Red = \sqrt{2}$, $Green = 1$, and $Blue = \frac{1}{\sqrt{2}}$. Right: reconstructed model (869,803 triangles).

from the templeRing model, showing which of the three horizontal window scales were used to recover each depth estimate. In this visualization (shown in Figure 5.4) green pixels represent the un-scaled window, red pixels represent matches using $\sqrt{2}$ scaling, and blue pixels correspond to $\frac{1}{\sqrt{2}}$ scaling.

In addition to being accurate and efficient, our algorithm works particularly well for objects with deformable surfaces, which we demonstrate by reconstructing a shirt sleeve (Figure 5.5), a crumpled blanket (Figure 5.6), and a bean-bag chair (Figure 5.7). These datasets were captured using a Canon Digital SLR camera, calibrated using the technique of Fiala and Shu [Fial 05]. For the shirt sleeve we include a zoom region to show the high-resolution mesh reconstruction that captures the individual wrinkles. Previously, a result of this visual quality has only been achieved by placing colored markers directly on the shirt [Whit 07]. For the blanket model, we include a window-scale image similar to Figure 5.4. Here we can see the different horizontal window scales that are used in a single binocular pair as a result of the high-curvature surface of the blanket. Finally, we reconstruct a bean-bag chair showing deformation before and after placing a heavy weight on it. With a number of synchronized video cameras, we can reconstruct this deformation over time

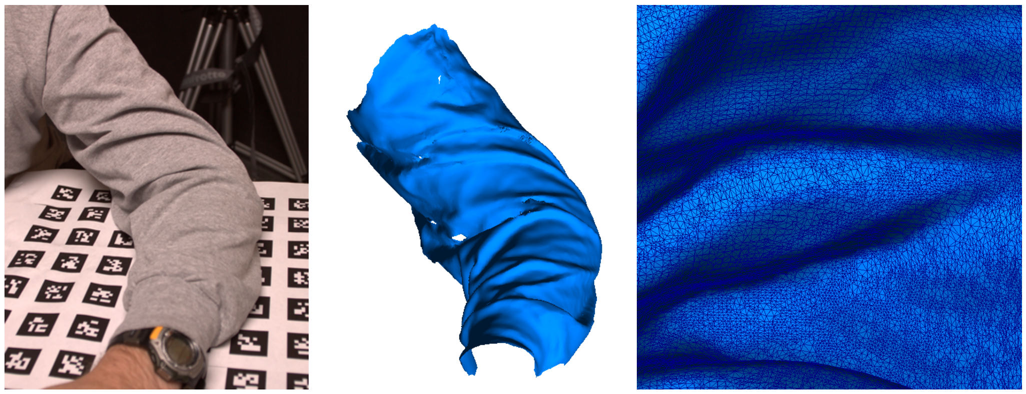


Figure 5.5: Sleeve reconstruction. Left: cropped region of 1 of 14 input images (2048x1364). Middle: final reconstruction (290,171 triangles). Right: zoom region to show fine detail.

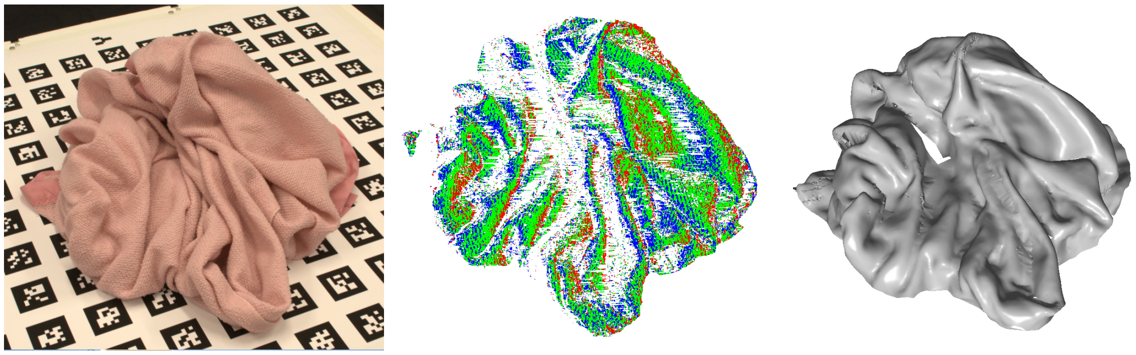


Figure 5.6: Crumpled blanket. Left: 1 of 16 input images (1954x1301). Middle: re-colored disparity image to show window scales. Right: final reconstruction (600,468 triangles).

by applying our MVS technique at each time-step. Since typical video sequences of deforming objects can have hundreds or even thousands of frames, the efficiency of our method makes it ideally suitable for such applications.

5.5 Discussion

In this work, we have introduced a novel multi-view stereo algorithm based on merging binocular depth maps with a state-of-the-art meshing algorithm. By employing an efficient, yet high-quality point-based processing pipeline, we are able to effectively remove outliers and suppress high-frequency noise. This robustness allows us to use relatively simple but efficient methods for the binocular stereo part, without paying too much attention to the reliability of the stereo matches. In addition, we increase the number of stereo matches by employing a scaled window matching approach that, in turn, provides the meshing stage with additional information. It is this combination of a simple stereo algorithm producing lots of matches with sophisticated point-based post-processing that allows us to create high-quality reconstructions very efficiently.

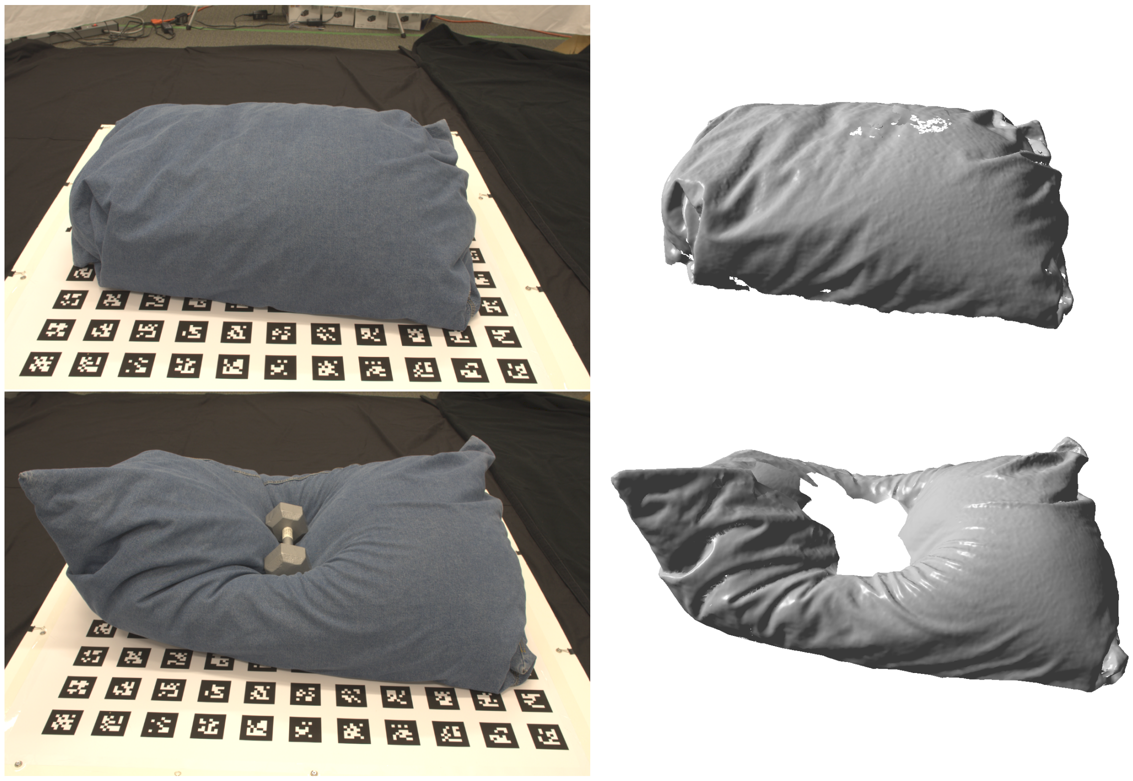


Figure 5.7: Bean-bag chair reconstruction, before (top) and after (bottom) deformation. Left: 1 of 16 input images (1954x1301). Right: final reconstruction (4,045,820 and 3,370,641 triangles).

Chapter 6

Markerless Garment Capture

Capturing the geometry of moving garments and other cloth is a problem that has recently seen a significant amount of research interest. One goal of such work is to provide a data-driven alternative to cloth simulation in much the same way motion capture provides an alternative to character animation. Perhaps more importantly, cloth capture promises to become an indispensable tool for postprocessing and augmenting live action sequences with computer rendered content.

Consider, for example, a live action sequence in which the texture or material properties of a garment are to be changed in post production (see Figure 6.1). In order for synthetic surface texture to move correctly with the real cloth geometry, it is not only necessary to capture the 3D shape of the garment in each time step, but also to establish a temporally consistent parameterization of the surface that prevents the texture from floating on it.

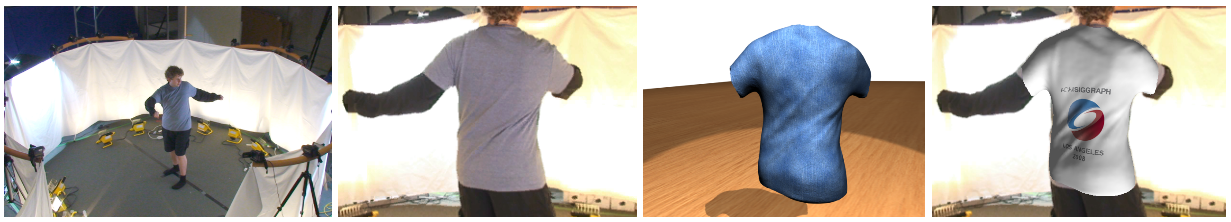


Figure 6.1: Markerless garment capture. From left to right: an actor in the capture setup, one of sixteen viewpoints, the recovered geometry, augmenting the original scene.

For this reason, the state of the art in cloth and garment capture is to use markers printed on the garment to simultaneously track geometry and parameterization. The need for such markers unfortunately makes existing cloth capture approaches unattractive for several reasons. First, the effort to create garments with unique marker patterns can be prohibitive for some applications. Second, the types of fabrics on which markers can be printed are typically limited, making it difficult to capture fabric-specific differences in garment motion. However, the characteristic differences between, say, silk, cotton, and fleece, are precisely the reason why one would want to capture cloth as opposed to simulating it. A key goal of our work is thus to capture the motion of specific existing garments, with a specific design, and made from a specific fabric.

In our work, we present a marker-free approach to garment capture that realizes this goal of capturing off-the-shelf garments [Brad 08b]. We use multi-view stereo for obtaining initial garment geometry for each time step. Although this initial geometry can have a significant number of holes, by leveraging low fabric stretch and utilizing the topology of the scanned garments, we are able to produce a parameterization of the geometry that is consistent across time, without explicitly tracking motion. This temporally consistent

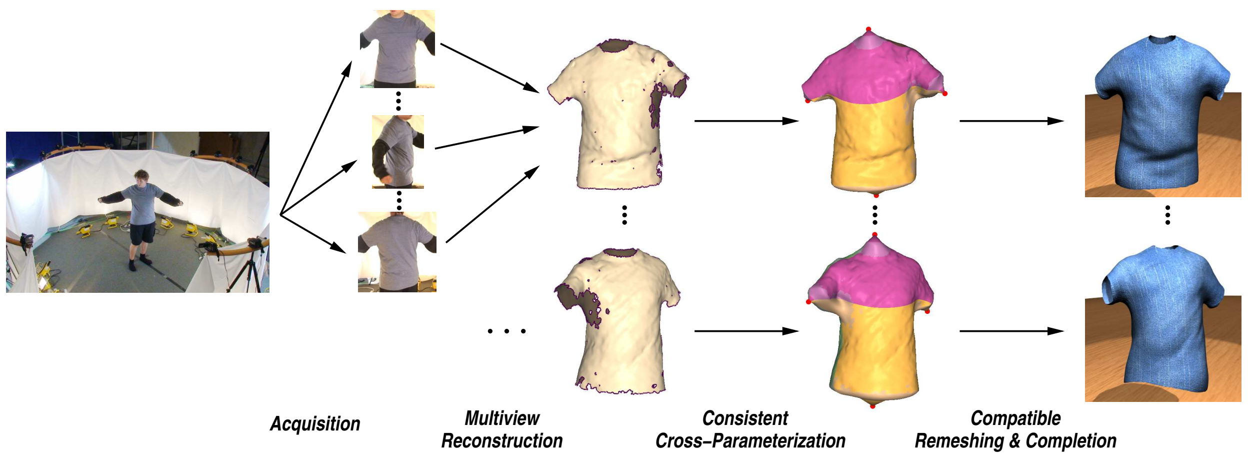

Acquisition
Multiview Reconstruction
Consistent Cross–Parameterization
Compatible Remeshing & Completion

Figure 6.2: Overview of our markerless garment capture technique.

parameterization helps us fill the holes in the geometry, producing a high quality final result.

6.1 Overview

Our method for markerless garment capture consists of four key components, illustrated in Figure 6.2. The first two components are contributions of this thesis, while the second two are not.

Acquisition: We deploy a unique 360° high-resolution acquisition setup using sixteen high-definition consumer video cameras. Our lighting setup avoids strong shadows by using indirect illumination. The cameras are set up in a ring configuration in order to capture the full garment undergoing a range of motions.

Multi-view Reconstruction: The input images from the sixteen viewpoints are fed into our multi-view stereo reconstruction algorithm (see Chapter 5) to obtain an initial 3D mesh for each frame. The resulting meshes contain holes in regions occluded from the cameras, and each mesh has a different connectivity.

Consistent Cross-Parameterization: We then compute a consistent cross-parameterization among all input meshes. To this end, we use strategically positioned *off-surface anchors* that correspond to natural boundaries of the garment. Depending on the quality of boundaries extracted in the multi-view stereo step, the anchors are placed either fully automatically, or with a small amount of user intervention.

Compatible Remeshing and Surface Completion: Finally, we introduce an effective mechanism for compatible remeshing and hole completion using a template mesh. The template is constructed from a photo of the garment, laid out on a flat surface. We cross-parameterize the template with the input meshes, and then deform it into the pose of each frame mesh. We use the deformed template as the final per-frame surface mesh. As a result, all the reconstructed per-frame meshes have the same, compatible connectivity, and the holes are completed using the appropriate garment geometry.

We now elaborate on the different components of the technique. Although I was involved in all aspects of this work, this thesis is more focused on the acquisition setup, the multi-view reconstruction step, and the overall concept of the algorithm, with less focus on the geometry processing steps. However, for the sake of completeness the cross-parameterization, remeshing and surface completion steps will also be described in this Chapter.

6.2 Acquisition

This section describes our acquisition setup, including the hardware, synchronization and lighting.

Cameras. Our acquisition setup consists of sixteen high-definition Sony HDR-SR7 video cameras, mounted in a ring around the actor wearing the garment to be captured. External and internal camera parameters are calibrated using ARTag markers [Fial 05], as described in Chapter 4.

De-interlacing. The cameras record thirty frames per second, stored as sixty interlaced fields per second. We de-interlace the sequence using a simple spatio-temporal technique in order to convert each field to a full resolution (1920×1080) image. Our method weights spatial interpolation higher in regions with fast motion, and uses simple temporal interpolation where there is no motion. A pixel $I(x,y,t)$ on a missing scanline is reconstructed as

$$\begin{aligned} I(x,y,t) = w \cdot \frac{I(x,y,t-1)+I(x,y,t+1)}{2} &+ \\ (1-w) \cdot \frac{I(x,y-1,t)+I(x,y+1,t)}{2}&, \end{aligned}$$

where w is a Gaussian function of the difference $I(x,y,t-1) - I(x,y,t+1)$, a choice that is inspired by the Bilateral filter [Toma 98]. The final data stream for each camera then takes the form of high-resolution (1920×1080) images, captured at sixty images per second.

Synchronization. Since all cameras operate independently, and cannot be synchronized using any external trigger, software synchronization needs to be applied in a post-process. We use the *moving-object* synchronization method described in Chapter 3. Although this method yields offsets with sub-frame precision, we choose to round the offsets to the nearest integer field. At sixty fields per second, we observe that artifacts due to sub-field synchronization differences are not noticeable during moderately fast motion.

Lighting. Some care has to be taken in the lighting setup as well. As in all movie and video shots, bright illumination is necessary to avoid noise and motion blur by reducing both the camera gain and the exposure time. Moreover, direct illumination by a few bright spot-lights causes hard shadows that exceed the dynamic range of the cameras, thus preventing geometry reconstruction in the shadowed regions. The ideal illumination is therefore bright, diffuse lighting, not unlike that used on movie sets.

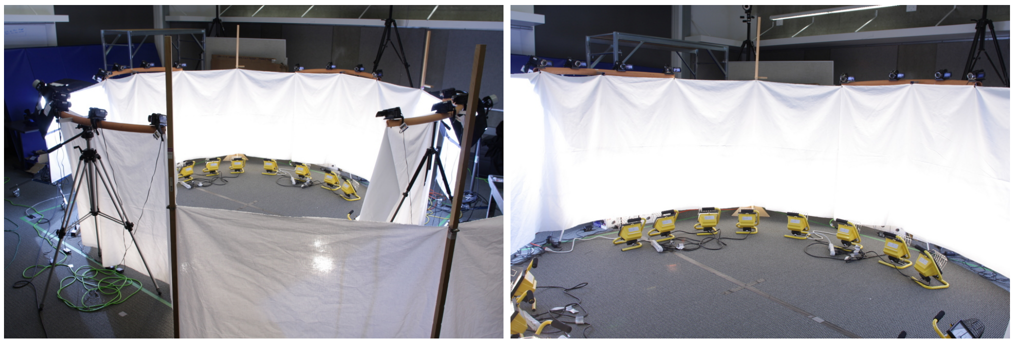


Figure 6.3: Acquisition setup.

To achieve this goal, we devised the indirect lighting setup depicted in Figure 6.3. The outer ring of our setup is lined with white cloth, which we use as a diffusing reflector. Sixteen inexpensive 500W halogen lights are placed inside the ring, pointing outwards at the cloth. This setup results in a bright, uniform illumination of the ring interior.

6.3 Multi-view Reconstruction

The captured and processed images are used as input for the multi-view stereo reconstruction algorithm described in Chapter 5.

The individual stages of our multi-view stereo algorithm in the context of garment capture are shown in Figure 6.4. We start with binocular stereo between pairs of adjacent cameras, and then merge the computed depth maps into a single dense point cloud, which is subsequently downsampled, de-noised and triangulated.

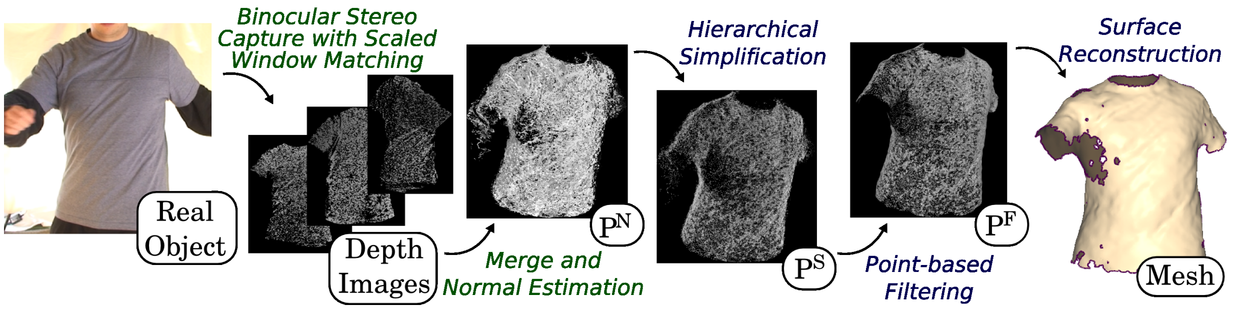

Binocular Stereo Capture with Scaled Window Matching
Hierarchical Simplification
Surface Reconstruction
Real Object
Depth Images
Merge and Normal Estimation
P^N
P^S
Point-based Filtering
P^F
Mesh

Figure 6.4: Multi-view reconstruction: starting from a set of depth images, a 3D point cloud is generated (P^N), simplified (P^S), filtered (P^F) and meshed.

Multi-view stereo is performed at each timestep. The efficiency of our method allows us to process hundreds of frames, and we do so in parallel. The output of the reconstruction is a triangle mesh for each video frame. Even though our multi-view stereo algorithm is among the state of the art for sparse views according to the Middlebury benchmark [mvie, Seit 06], significant holes remain in occluded and saturated image regions. The subsequent stages of our cloth capture system therefore have to robustly deal with missing data in the cross-parameterization between the frames, as well as fill in the missing regions from a template. Figure 6.5 shows the multi-view reconstructed frames from three different garments in motion, along with the final reconstruction result after all stages of our algorithm.

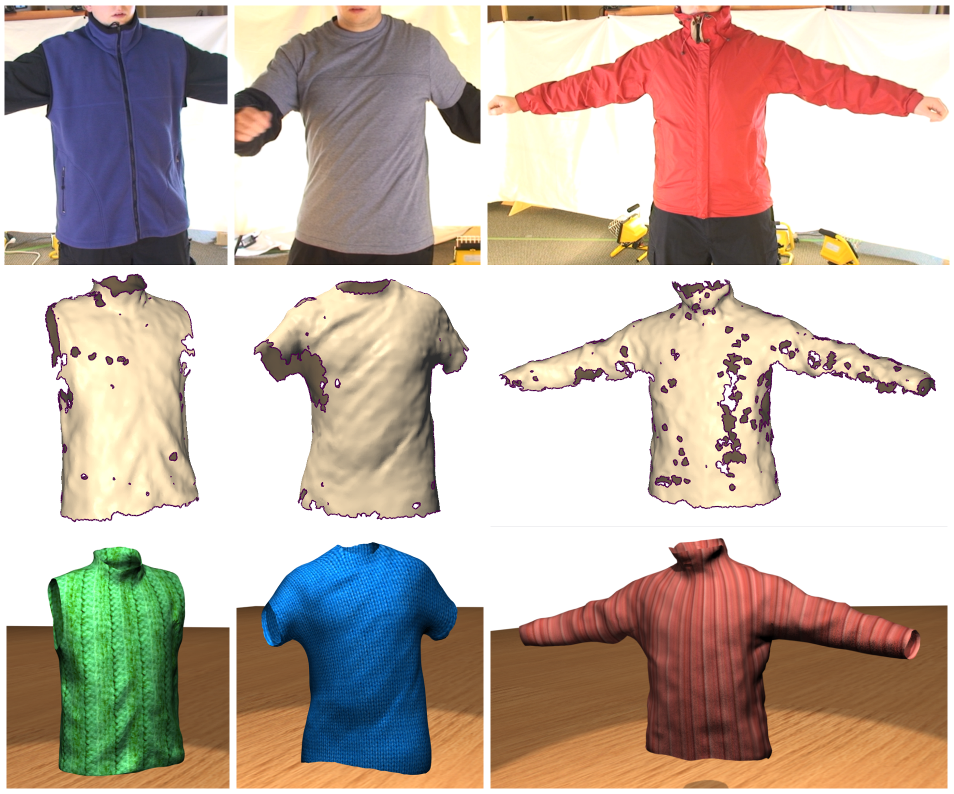


Figure 6.5: Result of multi-view reconstruction for different garments (middle row). One of the sixteen input images is shown above each result, and the final reconstruction after parameterization and completion is shown below.

6.4 Geometry Processing

The next step of our method is to compute a cross-parameterization of the geometry that is consistent across time, and to establish a compatible remeshing of the sequence using a template, so that each frame has the same connectivity and an explicit vertex correspondence.

6.4.1 Consistent Cross-Parameterization

In order for cloth capture to be useful, the computed parameterization needs to capture the motion of the garment, mapping a point on the garment at one frame to the position of the same point on the subsequent frame. Inconsistencies in the mapping are likely to lead to visible artifacts such as "floating" texture on

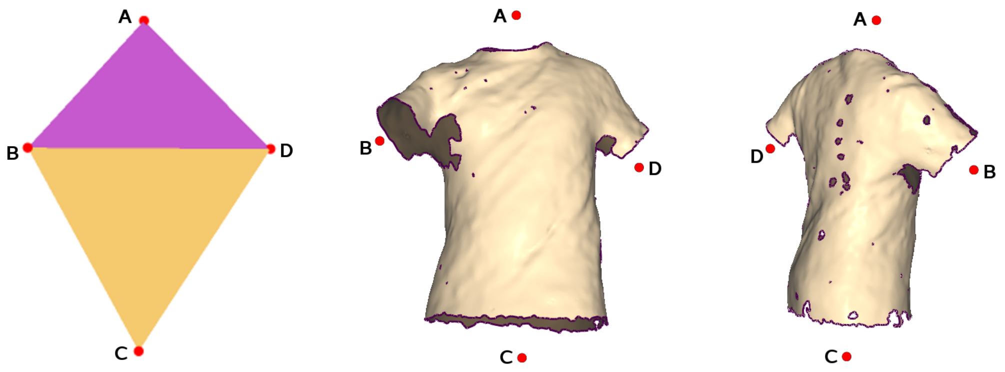


Figure 6.6: Off-surface anchor vertices are used to establish boundary correspondences between the tetrahedral base mesh (left) and the different frames of the same garment (right).

the garment surface. We have no explicit information on the motion of the garment, and estimating such motion is quite challenging. Instead we rely on geometric properties of the garments to compute a consistent parameterization. Many real fabrics exhibit very low stretch in regular use (see e.g. [Gold 07]), and thus we can assume that a consistent parameterization of garments is (nearly) isometric. Moreover, since most garments exhibit no rotational symmetries, such an isometric parameterization is unique. A key insight of our work is therefore that we can obtain a consistent parameterization between frames simply by finding an as-isometric-as-possible, i.e. stretch-minimizing, parameterization between the per-frame meshes, instead of employing some form of tracking.

There are several challenges in finding a cross-parameterization between captured garment frames. First, the per-frame meshes are incomplete due to occlusions. Second, we have a large number of frames to process. We cannot manually add markers in every frame to assist the parameterization, nor can we employ computation-intensive approaches such as the work by Anguelov et al. [Angu 05] or Schreiner et al. [Schr 04].

In order to parameterize the meshes in a reasonable time frame, we use the *base mesh* approach [Prau 01, Schr 04, Krae 04, Krae 05]. A base mesh (Figure 6.6, left) is a low-resolution mesh whose vertices correspond to a consistent set of marker vertices on the input meshes. In base mesh parameterization, each mesh is mapped to the base, such that the marker vertices map to the corresponding base vertices. This provides a cross parameterization between the inputs, where a map from one mesh to another is given by combining the map from the first mesh to the base with the inverse map from the base to the second mesh. In all previous methods, the on-surface marker vertices are manually specified by the user for each input mesh. As mentioned before, it would be infeasible in our setting to provide such markers manually for all frames. However, we observe that garments, in contrast to generic geometric models, have distinct, identifiable, topology with several boundary loops. For instance a T-shirt has four boundaries: a neckline, two sleeves and a waistline. Our algorithm identifies the corresponding boundaries across all frames and associates a floating, off-surface anchor vertex with each boundary (see Figure 6.6). The anchors replace the on-surface markers in the base mesh construction and subsequent parameterization.

In general, even when using a base mesh for stretch-minimizing cross-parameterization, we need to measure and optimize the stretch directly between the input meshes, a fairly time-consuming process. However, given two meshes that are isometric, mapping both of them using stretch-minimization to the same base mesh, we can expect the maps to be nearly identical, resulting in a nearly isometric map between the input meshes. Even in the presence of holes, careful design of a low-stretch base mesh parameterization algorithm allows us to compute cross-parameterizations that are consistent across the entire sequence by parameterizing the meshes independently to the base. This makes our method fast enough to be used in practice and allows for parallel processing of individual frames.

Positioning Off-Surface Anchors. The off-surface anchors in our setup correspond to the natural boundaries of the processed garment and are mapped to the vertices of the base mesh during parameterization (Figure 6.6). The anchors are placed manually for the first frame of each video sequence and are updated over time to follow the motion. Each anchor is updated automatically, except where the multiview algorithm has failed to reconstruct a sufficient amount of geometry on the corresponding mesh boundary (see Figure 6.7, right). As mentioned in Section 6.3, missing geometry can be a problem due to occlusion and the relatively small number of cameras.

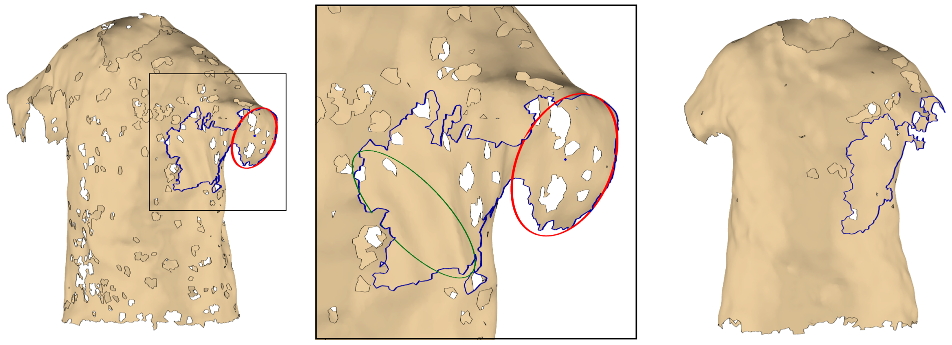


Figure 6.7: Garment boundary tracking. Left: the automatic method correctly chooses the red ellipse based on confidence, size and temporal similarity. Center: using confidence alone, the green ellipse would be incorrectly chosen. Right: the boundary loop contains very little of the actual garment boundary and user-assisted tracking is required.

Automatic Boundary Tracking. We observe that boundary loops of typical garments have an approximately elliptical shape. Given an elliptical approximation of each boundary, the corresponding anchor is placed a fixed distance away from the ellipse center with an associated normal vector pointing away from it. The normals are used in the hole filling process. To find the best fitting ellipse, we use a variant of the RANSAC algorithm [Fisc 81]: for each boundary loop of the mesh that corresponds to a garment boundary, we iteratively choose six boundary vertices at random and then fit an ellipse to these vertices. The *confidence* for any given ellipse is calculated using the number of boundary vertices that agree with that ellipse. The selection of a best fitting ellipse is further guided by size and temporal similarity to the position of the corresponding ellipse in the previous frame. The number of RANSAC iterations required in our setting is on the order of $\binom{n}{6}$, where n is the length of a typical boundary. Missing portions of the geometry such as sections under the arm, which are often not captured by the multiview stereo algorithm, can cause significantly prolonged boundary loops. As a result, 50K to 100K iterations may be required per boundary for such frames, taking fifteen to thirty seconds per frame.

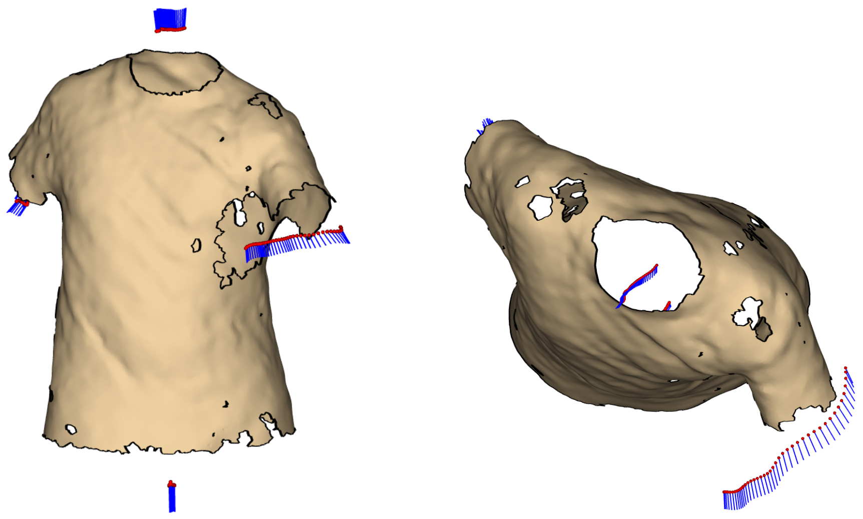
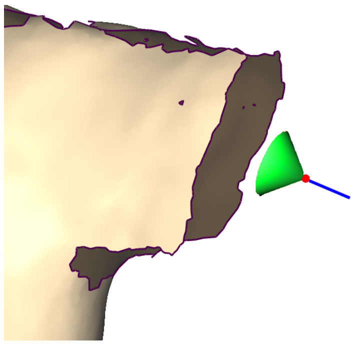


Figure 6.8: Automatic off-surface anchor positioning across a sequence of frames. Anchor vertices are shown in red, while normals are shown in blue. Front view (left) and top view (right).

Anchor positions and normals are smoothed in time using Gaussian smoothing. Figure 6.8 shows the resulting anchors for a short sequence of a garment undergoing motion. The anchors with normals are shown for each frame, and the garment geometry is shown for one frame of the sequence. This method can deal with significant missing geometry on the boundary, as shown in Figure 6.7 (left). However, for degenerate boundaries such as the one shown in Figure 6.7 (right), the automatic positioning fails. A short consecutive sequence of poor frames can still be handled gracefully, as the method will auto-detect and skip over the problematic boundaries when no suitable ellipse is found. However, this technique cannot recover from sequences with a large number of consecutive unrecognizable boundaries.

User-Assisted Boundary Tracking. For sequences where the automatic anchor placement fails for a larger number of consecutive frames, we have developed a semi-automatic alternative. For a number of keyframes, the user chooses an ellipse for each boundary from a selection of high-confidence ellipses determined by the RANSAC algorithm. Choosing an ellipse from a precomputed set typically takes about 10 seconds per boundary. In between keyframes, the anchor positions are interpolated using splines. The required density of keyframes depends on the speed of motion; for fast motions, a keyframe approximately every 30 frames (1/2 second) yields good results.

Base Mesh Parameterization. For each input garment sequence we first construct a base mesh and then map the per-frame meshes onto the base. The base is constructed by positioning the vertices at the locations of the corresponding anchors in a canonical pose (Figure 6.6, left). Given the common base, we compute a stretch-minimizing parameterization of each mesh onto the base using the computed anchors.



As an initialization step, a small cone is created around each anchor, with the apex pointing in the direction of the anchor normal (right). These cones are used later-on to facilitate smooth hole completion. Since our parameterization method mostly follows the work by Kraevoy and Sheffer [Krae 05], we focus our description on the differences with their method and refer the interested reader to their paper for more details.

The algorithm first segments the mesh into charts corresponding to the base mesh facets and constructs an initial parameterization by mapping each chart to the corresponding facet. Since the anchor cones form separate connected components, we triangulate the gaps between them and the corresponding boundaries before computing the segmentation. Given the initial parameterization, the method iterates between re-triangulating the holes in the mesh, including the gaps between the anchor cones and the actual surface, and reparameterizing the mesh onto the base. The main goal of the process is to parameterize the mesh onto the base with minimal stretch.

During re-triangulation, Kraevoy and Sheffer [Krae 05] first triangulate holes on the base, and then project the new vertices back to 3D using a Laplacian type formulation that minimizes

$$\sum_i (P_i - \sum_{j\in N(P_i)} \omega_{ij} P_j)^2, \tag{6.1}$$

where P_i are the projected vertices with neighborhood $N(P_i)$, and ω_{ij} are the normalized mean value weights [Floa 03] from the base embedding. This formulation creates a membrane type surface, filling the holes in 3D with C^0 continuity along hole boundaries (Figure 6.9, center). The discontinuities along the boundaries lead to a sub-optimal estimation of the actual hole shape and area and increase the parameterization stretch. We rectify this problem by introducing additional terms $(P'_i - \sum_{j\in N(P'_i)} \omega_{ij} P_j)^2$ into the minimized functional in Equation 6.1 for the vertices P'_i on the hole boundary. While the 3D positions of these boundary vertices remain fixed, the new terms influence the positions of the hole vertices adjacent to them. The additional terms effectively eliminate the discontinuities across hole boundaries, creating a C^1 effect (Figure 6.9, right).

Note that the membrane reprojection is suitable for establishing the base mesh parameterization, but is not always sufficient for final garment hole completion. Filling holes in complex geometric regions, such as armpits, using the generic membrane generates oversmoothed geometry (Figure 6.12, left). We address this problem in Section 6.4.2, with a template-based remeshing and surface completion technique.

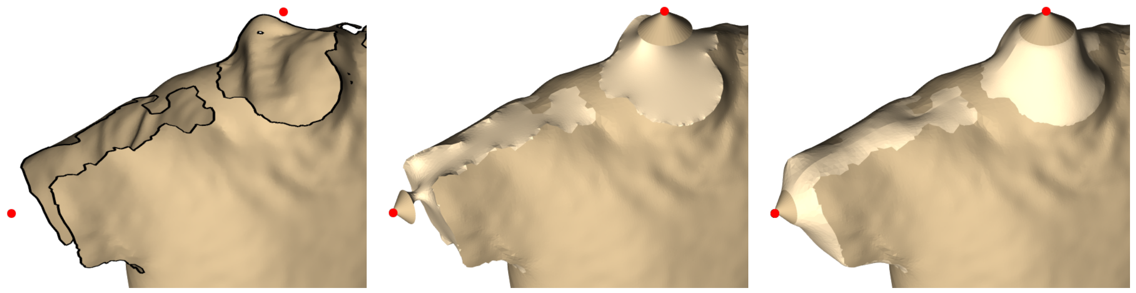


Figure 6.9: Smooth hole completion during parameterization. Left: an incomplete T-shirt. Center: C^0 membrane. Right: C^1 membrane. The membranes are rendered in lighter tones for visualization.

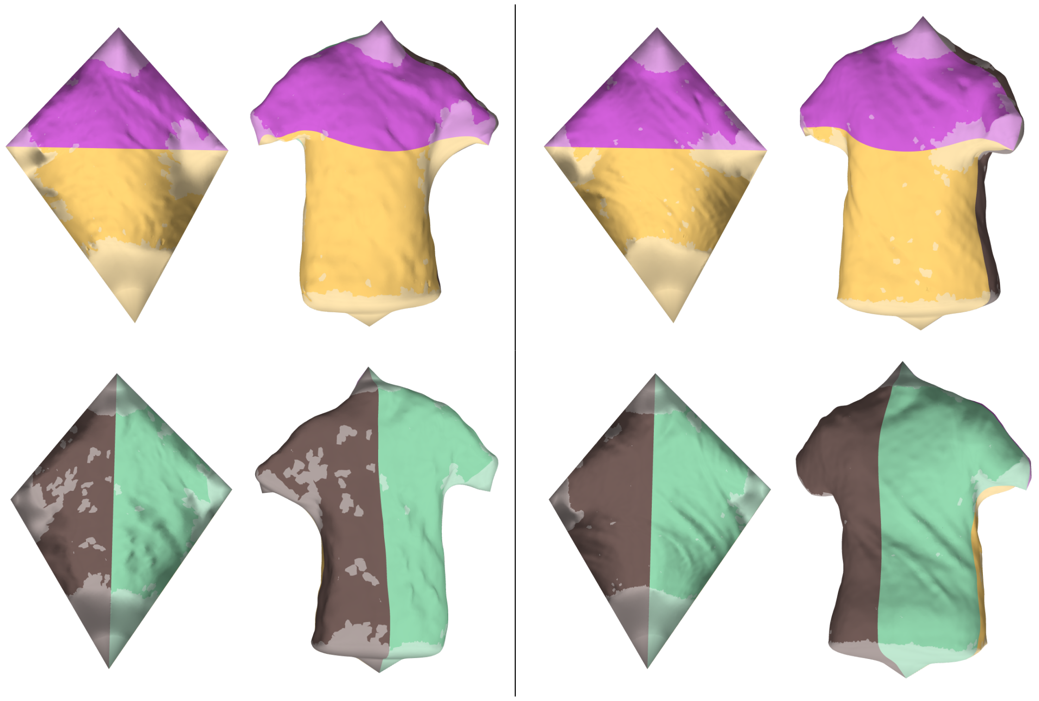


Figure 6.10: Two frames parameterized onto the base mesh, illustrated as normal maps (top row: front view, bottom row: back view). The charts corresponding to the base mesh faces are shown on the garments. Again, hole regions are rendered in lighter tones.

Since the geometry connecting the anchors to the mesh is iteratively updated as the parameterization improves, any inaccuracy in the anchor position is automatically corrected, thus the anchors need not be placed as accurately as on-surface markers.

Therefore, our anchor-based parameterization is fairly robust to noise and artifacts in the input meshes.

The result of the base mesh parameterization method for two input frames is illustrated in Figure 6.10. Thanks to the near-isometry between the frames, the natural boundaries on the normal maps are well aligned. Once each mesh has been parameterized onto the common base, a trivial cross-parameterization is established between all frames.

6.4.2 Compatible Remeshing and Surface Completion

The acquired garment frames typically have numerous holes, which are filled with a C^1 continuous membrane during the parameterization. For final reconstruction, these membranes work well for flat or convex regions, but generate an oversmoothed surface in saddle-like regions (see Figure 6.12). One option for replacing the membrane with more realistic geometry is data-driven hole filling [Whit 07]. White et al. complete hole geometry using the mapping from other input frames that do not contain the hole. However, some regions of the garment may be occluded in all input frames. For this reason, we use a template mesh, which we generate by "inflating" a photo of the garment to provide a more faithful geometry comple-

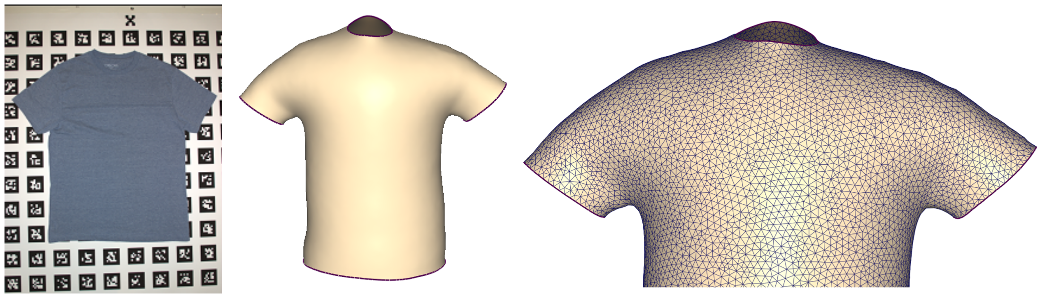


Figure 6.11: Template construction. A photo of the garment (left) is inflated to 3D.

tion. Additionally, we establish common, compatible connectivity by using the template triangulation for all frames.

Template Construction. We construct a template mesh from a single photo of the garment, laid out on a flat surface (Figure 6.11, left). The silhouette of the garment is extracted from the image and the interior is triangulated [Shew 96] creating a 2D mesh. The mesh is duplicated for the back surface, and garment boundaries are indicated by the user with brush strokes.

The 2D mesh is then inflated into 3D, using a variant of the iterative physical simulation approach of Mori et al. [Mori 07]. Each iteration consists of two steps. First, every face is moved slightly in its normal direction, to mimic the effect of internal pressure. Then the length of each edge is adjusted to preserve the stretch of the material. In the work of Mori et al., material stretching is prevented while compression is tolerated, a desired effect for their application. For our garment template, we wish to prevent both stretch and compression. Thus, we modify the second step of the algorithm to include a compression prevention condition. In this step, the displacement d_{v_i} of a vertex v_i is computed as a weighted sum of the forces (t_{ij}) from the neighboring edges (E_i):

$$d_{v_i} = \frac{\sum_{e_{ij}\in E_i}\{A(e.leftface)+A(e.rightface)\}t_{ij}}{\sum_{e_{ij}\in E_i}\{A(e.leftface)+A(e.rightface)\}} \tag{6.2}$$

$$t_{ij} = \begin{cases} 0.5\cdot(v_j - v_i)\cdot\frac{|v_i-v_j|-l_{ij}}{|v_i-v_j|} & \text{if } |v_i - v_j| \geq l_{ij} \\ 0.5\cdot(v_i - v_j)\cdot\frac{l_{ij}-|v_i-v_j|}{l_{ij}} & \text{if } |v_i - v_j| < l_{ij} \end{cases}, \tag{6.3}$$

where $A(f)$ is the area of a face f and l_{ij} is the length of an edge e_{ij} in the initial 2D mesh. This formulation differs from Mori et al. in Equation 6.3, where the first condition is for preventing stretch and the second condition is for preventing compression.

A template is created only once for each garment. The constructed template of a T-shirt is shown in Figure 6.11.

Remeshing and Completion The cross-parameterization of the input meshes will be used for consistent texture mapping and temporal smoothing of the geometry. Thus, we require that the meshes for all frames be compatible, i.e., have the same connectivity and explicit vertex correspondence. This is achieved by projecting the uniform triangulation of the template mesh onto each frame.

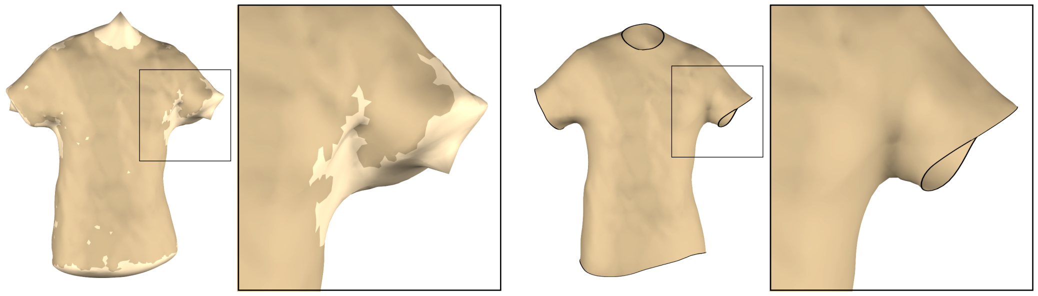


Figure 6.12: Initial C^1 smooth membrane (left) and result of surface completion using the template from Figure 6.11 (right).

The template is parameterized onto the base mesh in the same way as the input frames establishing a mapping between them. This mapping is sufficient for remeshing each frame with the template connectivity by simply mapping the vertices of the template to each frame. However, for the hole regions in each frame we prefer to use the geometry of the template instead of the membrane. We therefore use the mapping from the template onto each frame to map only those vertices of the template that map to the original frame geometry. To obtain the position of the remaining vertices, we deform the template into the pose of the frame, using the mapped vertex positions as constraints. We use the deformation method of Sheffer and Kraevoy [Shef 04], although other algorithms could be used instead. Figure 6.12 shows the result of the completion for one frame. This technique produces a compatible, uniform triangulation of the whole frame sequence, while completing the holes with surface patches that match the captured garment.

Finally, the frame sequence is smoothed in the temporal domain using Gaussian smoothing to remove temporal noise.

6.5 Results

The results of different garment captures are shown in Figures 6.13- 6.18. We are able to render the captured geometry separately, as a replacement of the original garment, and as an augmentation to the original video frame. Our technique is, to our knowledge, the first method that is able to produce all three of these results from a single capture. In particular, augmentation would not be possible without markers remaining visible in most of the previous work.

Figure 6.13 shows different frames from a T-shirt sequence. This is the same T-shirt that was used in illustrations throughout this chapter.

A fleece vest reconstruction is shown in Figure 6.14. We note that for this garment, the multiview algorithm produced only small holes, unlike the large holes caused by occlusion on the T-shirt. As a consequence, we were able to reconstruct the vest without creating a template mesh. Instead, a compatible remeshing was achieved by performing a uniform triangulation on the base mesh and then projecting the geometry to each frame. The fleece vest also illustrates that we can indeed capture garments made from different fabrics. While it would be relatively straightforward to print marker patterns onto the cotton fabric of the T-shirt, producing patterned fleece is not as easy, due to the fuzzy nature of this fabric. Note that this fuzziness does

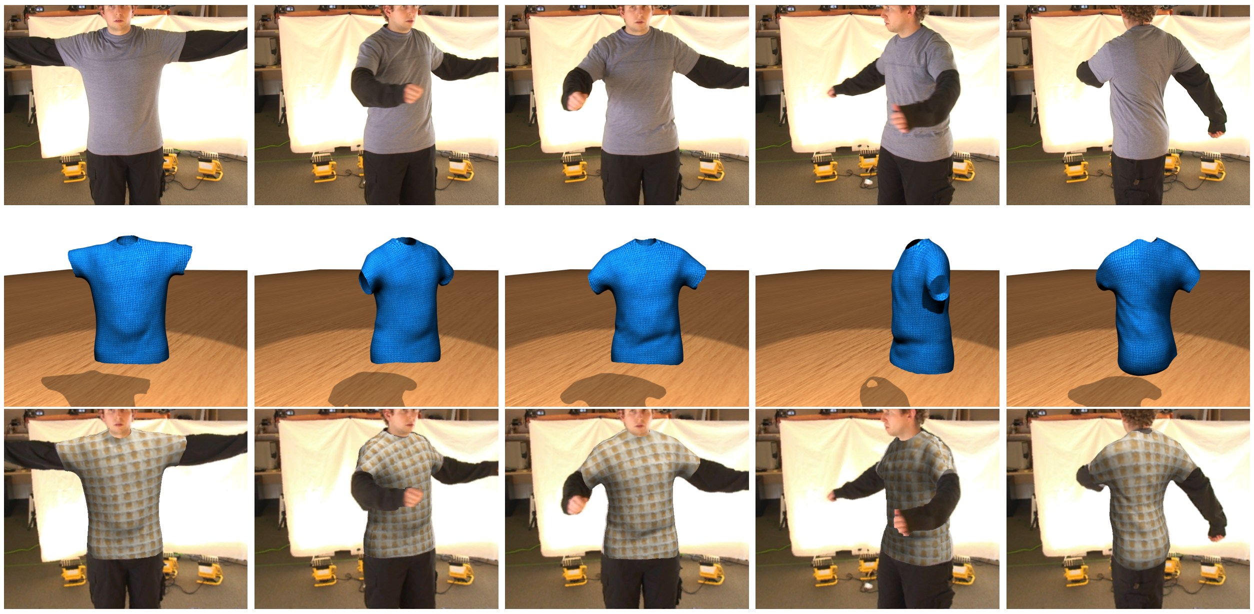


Figure 6.13: Capture results for five frames of a T-shirt. From top to bottom: input images, captured geometry, T-shirt is replaced in the original images.

not prevent reconstruction of features such as folds with our method.

A different T-shirt is shown in Figure 6.16. This garment is larger and more loosely fitting than the others, resulting in more ripples, which our capture method successfully reconstructs. Two full-body garments were also reconstructed, demonstrating the versatility of our approach. A pink dress is shown in Figure 6.17, and a blue dress in Figure 6.15. Figure 6.18 shows another result of garments made from different fabric, in this case a long-sleeve nylon-shell down jacket.

Processing a sequence from captured video to the final result requires approximately one hour per frame, which can be easily parallelized. The majority of the time is spent in the multi-view reconstruction step, since our input data is in the form of high-definition video and we establish dense correspondences between images. A typical frame for the T-shirt data sets reconstructs over a million surface points, which are used to construct an initial triangulation with over 100K vertices. The very large resolution is required at this point in order to preserve high-frequency detail in the garments, particularly since the mesh reconstruction is non-uniform. The re-meshed sequences typically have around 20K vertices.

6.6 Discussion

In this work, we have presented the first method for markerless capture of full garments and demonstrated its viability on a number of examples. Our acquisition and reconstruction methods generate per-frame meshes that capture the complex, changing structure of garments during human motion. The parameterization and completion algorithms leverage fabric incompressibility and garment topology to consistently parameterize the frame sequences, and generate realistic garment geometry and motion that is faithful to the input.

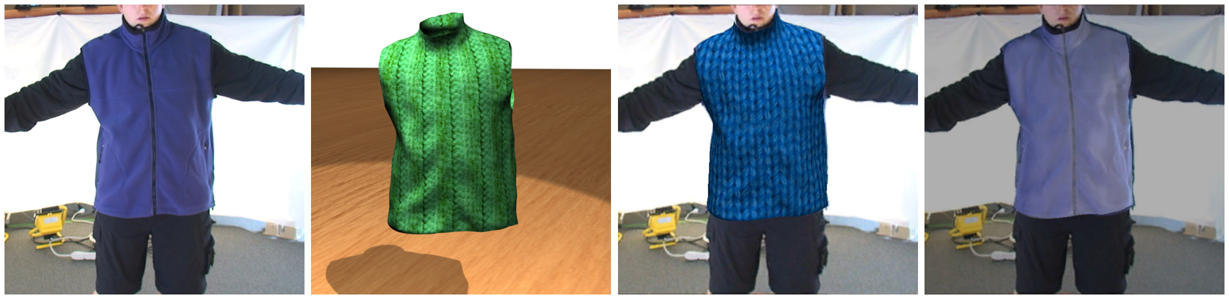


Figure 6.14: Capture results for a fleece vest. From left to right: one input frame, captured geometry, vest is replaced in the original image, the scene is augmented by adding a light source and making the vest more specular.

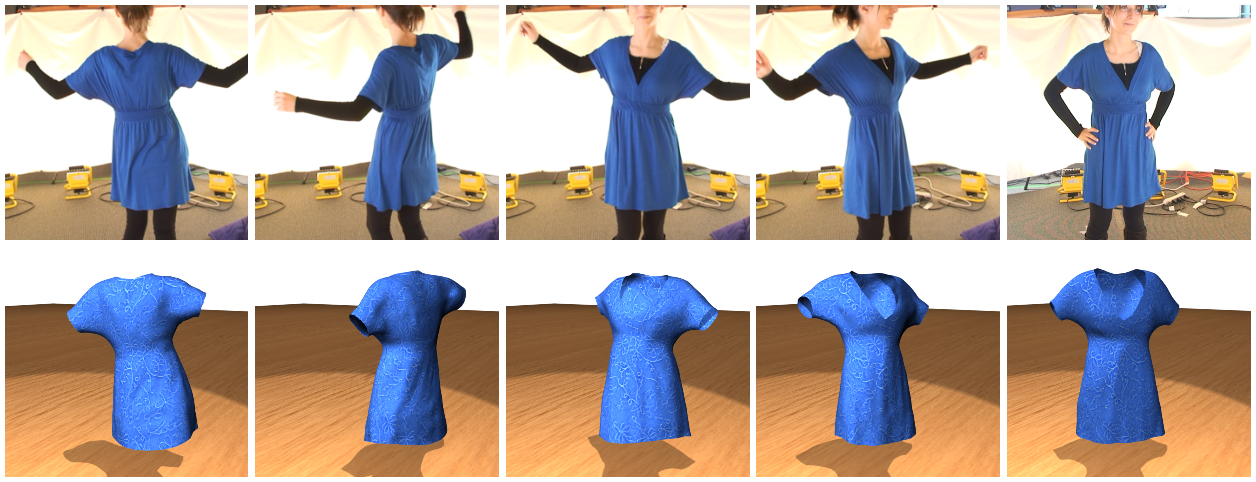


Figure 6.15: Capture results for five frames of a blue dress. Input images (top) and reconstructed geometry (bottom).

Our system has a few limitations. First, while our technique can handle significant occlusions, we do not handle situations where the garment surface comes in contact with itself, e.g. in case a sleeve touches the torso. The initial multi-view reconstructions in these situations can contain incorrect surface topology, which is not handled in the subsequent process. For this reason, the actors were asked to keep their arms away from their torso. A similar problem can also arise if the garment bunches up in one place, creating many small folds. The initial reconstruction in this case may not faithfully capture all the folds, smoothing over the surface and resulting in an apparent compression of the garment. This case may locally violate the incompressibility assumption and cause inaccurate inter-frame mappings. Finally, the necessary spatial and temporal smoothing steps in order to produce noise-free results tend to remove the fine details in the geometry. As a result, the reconstructed garments appear to have fewer wrinkles than the original input images. Despite these limitations, we believe that our method greatly advances the current state-of-the-art in garment capture.

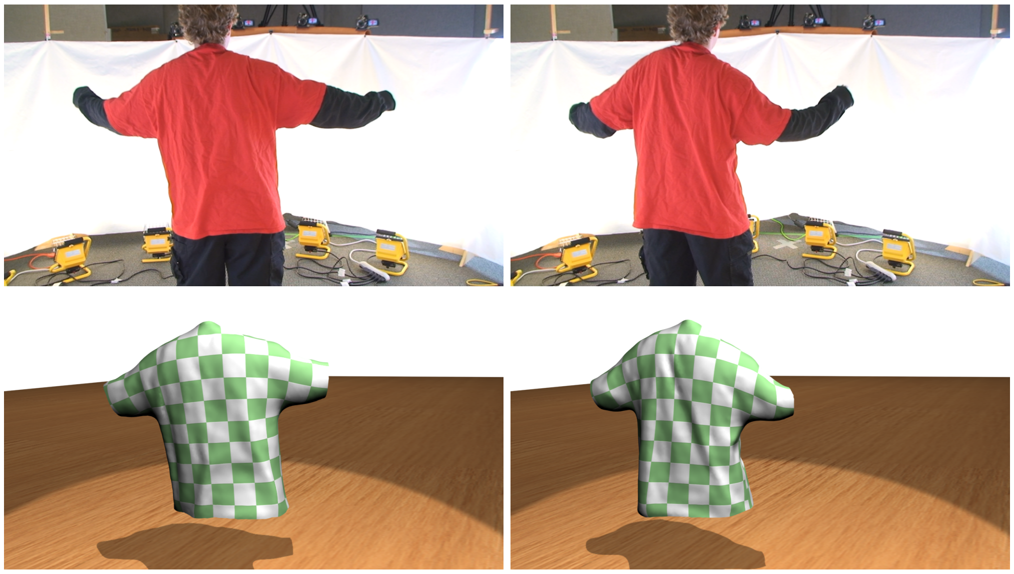


Figure 6.16: Capture results for two frames of a large T-shirt. Input images (top) and reconstructed geometry (bottom).

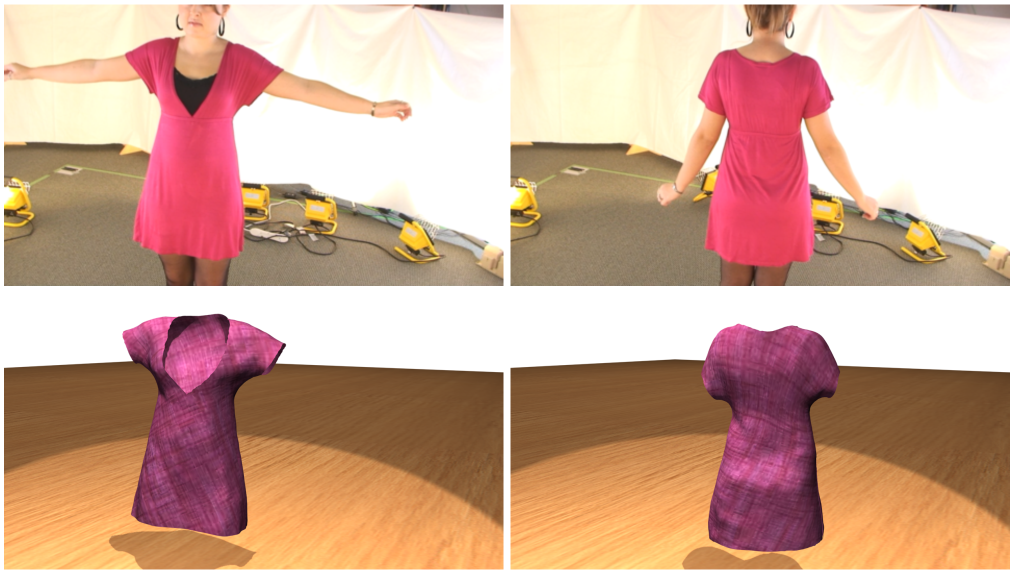


Figure 6.17: Capture results for two frames of a pink dress. Input images (top) and reconstructed geometry (bottom).

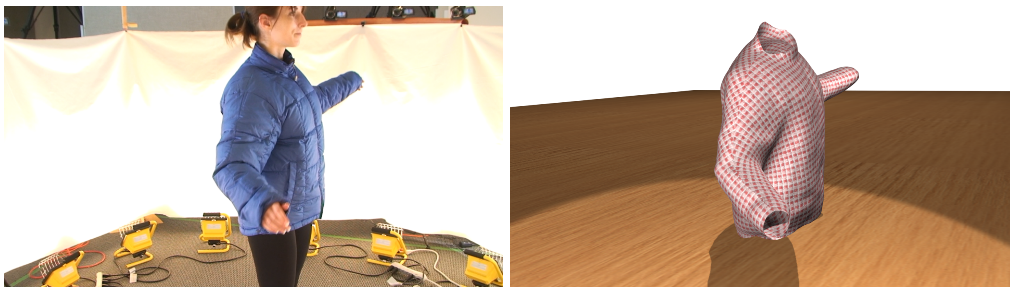


Figure 6.18: Capture result for a long-sleeve nylon-shell down jacket.

Chapter 7

High-Resolution Passive Facial Performance Capture

Facial performance capture is evolving into a major tool for creating realistic animations in both movie and game industries. A practical and versatile facial performance capture system should fulfill a number of requirements. First, it should generate sequences of detailed meshes with dynamic texture in order to capture both geometric deformations of the face, as well as the corresponding changes in skin appearance due to sweating or changes in blood circulation. Second, the triangulation of the meshes and their mapping between frames should be compatible over time, such that both the geometry and the texture can be edited by an artist (e.g. by applying virtual makeup), and these changes can be propagated over time. Finally, the capture process itself should be simple and automatic, and should not require separate geometry scans or excessively expensive hardware.

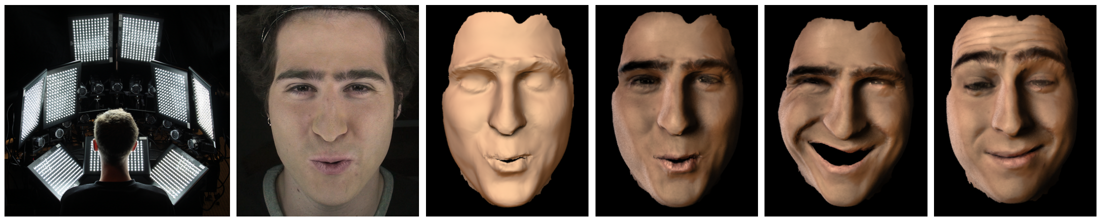


Figure 7.1: High resolution passive facial performance capture. From left to right: Acquisition setup; one reference frame; the reconstructed geometry (1 million polygons); final textured result; and two different frames of the sequence.

Reconstructing a human face is challenging, since most faces do not have sufficient medium-scale texture to establish dense correspondences between different viewpoints. For this reason, commonly-used methods typically involve either structured lighting [Zhan 04, Wang 04], special makeup [Furu 09c], or markers [Bick 07, Ma 08] to track the geometry. This makes it difficult to simultaneously capture both geometry and texture, thus requiring either inpainting of markers or sacrificing temporal resolution by staggering structured light with uniform light. Additionally, these active methods can be uncomfortable for the actors, affecting their performance. Many of these techniques also require an initial laser scan of the face, while the others suffer from low-resolution reconstructions.

In this Chapter, we present a fully passive method for facial performance capture that satisfies the criteria outlined above [Brad 10b]. By using only a camera array and uniform illumination, our setup is less intrusive to the actors. We require no markers, no face paint, no fluorescent makeup, no structured light, and no laser-scanned model, yet are still able to reconstruct high resolution time-varying meshes at 30 frames per second. Our passive setup also allows us to reconstruct high-resolution, time-varying texture maps that can capture potential changes in skin appearance due to, for example, blushing or sweating of the actor (see Figure 7.1). Our reconstruction is made possible by a number of contributions:

1. A novel high-resolution acquisition setup that allows us to use pores, blemishes and hair follicles as trackable surface features.
2. A high-quality stereo reconstruction algorithm, extending our method presented in Chapter 5 to include new disparity constraints that work with our camera setup.
3. Most significantly, an automatic surface tracking method based on optical flow. This method supports automatic drift correction, as well as an edge-based mouth tracking approach, which together yield realistic, time-varying, textured facial models.

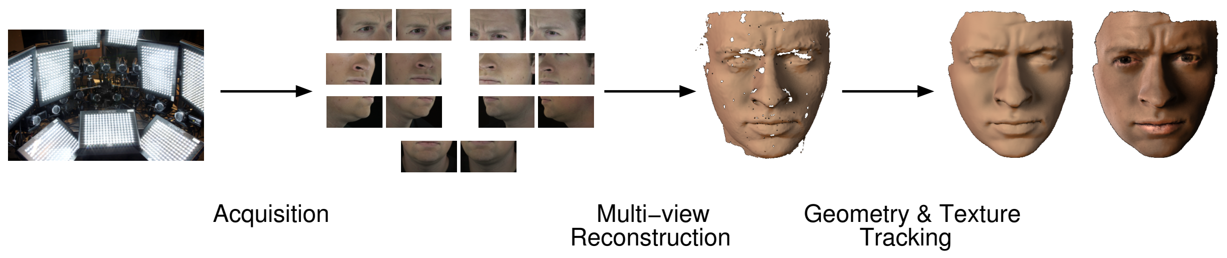


Figure 7.2: Overview of our facial performance capture algorithm.

Our facial performance capture system consists of the three main components shown in Figure 7.2:

- **Acquisition Setup:** Our setup consists of 14 high definition video cameras, arranged in seven binocular stereo pairs. Each pair is zoomed-in to capture a small patch of the face surface in high detail under bright ambient illumination (Section 7.1).
- **Multi-View Reconstruction:** We use an iterative binocular stereo method to reconstruct each of the seven surface patches independently, and then combine them into a single high-resolution mesh. The zoomed-in cameras allow us to use skin pores, hair follicles and blemishes as surface texture to guide the stereo algorithm, producing meshes with roughly 1 million polygons (Section 7.2).
- **Geometry and Texture Tracking:** In order to consistently track geometry and texture over time, we choose a single reference mesh from the sequence, and compute a mapping between it and every other frame by sequentially using optical flow. The observed pores and other surface details not only serve to provide accurate per-frame reconstructions, but also allow us to compute cross-frame flow. Drift caused by inevitable optical flow error is detected in the per-frame texture maps and corrected in the geometry. In order to account for the high-speed motion generated by talking, the mapping is guided by an edge-based mouth-tracking process (Section 7.3).

7.1 Acquisition Setup

Our acquisition setup consists of 14 high definition Sony HDR-SR7 cameras arranged in an array in front of the actor (see Figure 7.3). The cameras are geometrically calibrated in pairs using our approach from Chapter 4, and optically synchronized using our method described in Chapter 3. We use nine LED light fixtures, each with 192 LEDs, to provide both bright, uniform illumination, as well as the camera synchronization.

Our cameras are inexpensive consumer camcorders, which means that they are not connected to any computers for either synchronization or streaming of data. Although we solve the synchronization problem explicitly (see Chapter 3), we are left with a camera control problem, in that simple tasks such as turning the cameras on, recording, pausing, zooming, and downloading the data must be performed manually for each camera. This is a very tedious and time-consuming approach. Fortunately, the cameras support remote control operations via infra-red and wired remotes, using a standard protocol. We make use of this feature to control the entire array of cameras through a customized hardware design that interfaces with the cameras through the wired remote port. Using Arduino technology[4] and simple controlling software, basic camera operations can be automatically performed on the entire array via single commands sent from a laptop. A second Arduino hardware solution was designed for controlling the array of lights in a similar manner. This work was performed by Steven Stuber, an undergraduate student under my supervision.

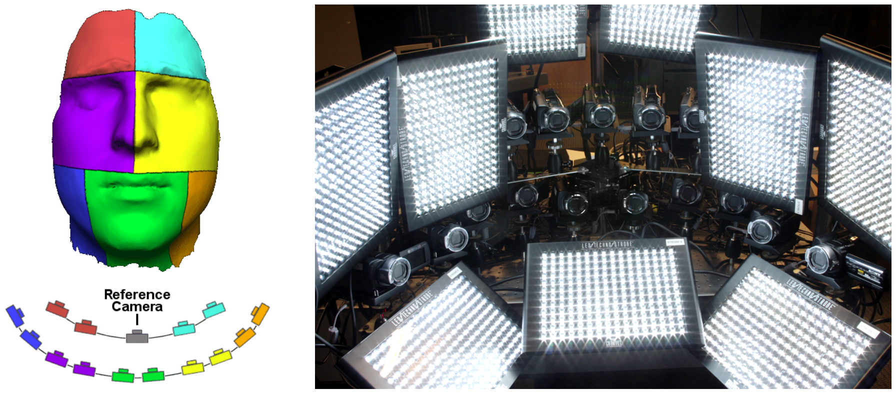


Figure 7.3: Schematic view (left) and photo (right) of our acquisition setup.

Reconstructing a human face is challenging because most faces do not have sufficient medium-scale surface texture. This is the main reason why previous work relies on hand-placed markers or structured light patterns. However, this assumption does not hold if we increase the resolution of the acquisition system and capture images of very small face details such as pores, freckles and wrinkles.

To this end, we arrange the cameras in seven stereo pairs and maximize the optical zoom level in order to observe fine-scale surface details, providing a natural surface texture for reconstruction. Figure 7.3 shows an illustration of the setup and the seven face regions observed by the stereo pairs, along with a photo of the actual setup. Note that there is significant overlap between the face regions, even though it is not shown in the illustration. Figure 7.4 shows a video frame captured from one camera, demonstrating the high resolution surface details. We also have an additional *reference* camera that is not zoomed-in and not used for processing. This camera provides a normal video for comparison with our results.

[4]www.arduino.cc

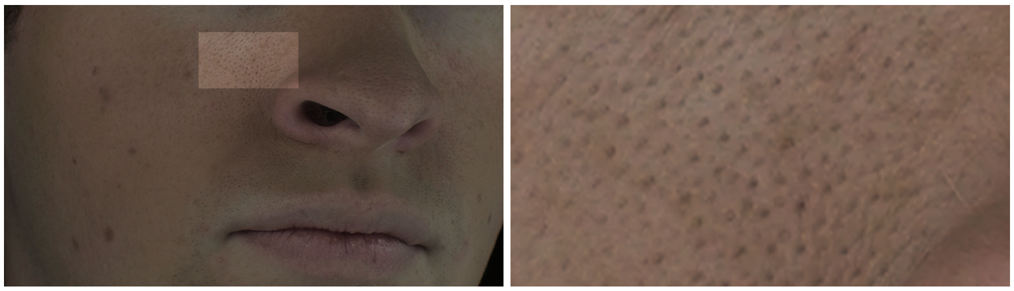


Figure 7.4: One HD video frame showing that pores can be used as natural surface texture for face reconstruction.

Preparing an actor for capture requires very little work, since we do not place markers on the face or require an initial scan. We do use off-the-shelf foundation makeup to reduce specularity in the case of oily skin, however this is common practice for preparing actors for any performance.

7.2 Multi-View Reconstruction

To reconstruct a mesh for each video frame, we perform binocular stereo for each of the seven facial regions, resulting in seven overlapping depth images. The natural surface texture obtained from our capture setup provides sufficient detail for stereo reconstruction. The seven depth images are then merged into a single dense point cloud and converted to a triangle mesh using the method that we described in Chapter 5. This system was successfully used for our garment capture work (Chapter 6), which has similar requirements on accuracy and efficiency for capturing deformable geometry.

However, the reconstruction method as described in Chapter 5 is designed for 360° reconstruction, where the visual hull of the object can be pre-computed and used to reduce outliers in the binocular reconstructions. Our current camera setup prevents us from applying the technique exactly as described, because we observe only patches of the face and do not have full 360° coverage. Thus we cannot compute a visual hull. Without the visual hull constraint, the resulting depth images can contain many outliers (see Figure 7.5, c). We resolve this problem by adopting an *iteratively constrained* binocular reconstruction approach, designed to iteratively remove outliers in the reconstruction. We apply this method to each of the seven camera pairs.

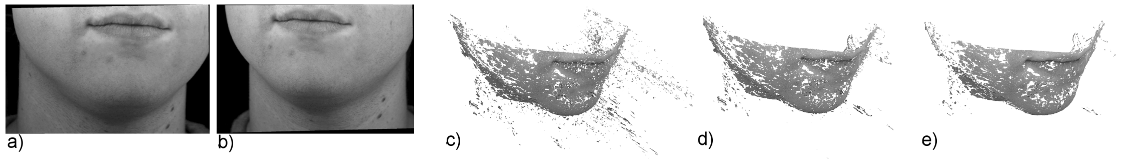


Figure 7.5: Iteratively constrained multi-view stereo removes outliers by iteratively tightening depth constraints.

Iteratively Constrained Binocular Reconstruction. In the first pass of binocular reconstruction, we enforce a loose depth constraint that corresponds to the maximum reconstruction volume for the face patch (measured by hand). The resulting depth image has many valid samples but also contains outliers. Figure 7.5 (a) and (b) shows an example stereo pair, and the depth outliers can be seen in the corresponding point cloud in Figure 7.5 (c). In order to reduce outliers, we tighten the initial depth constraints for the next iteration. To this end, we perform Gaussian smoothing on the current depth image using a large kernel size, creating an over-smooth approximation of the surface with fewer outliers. This smooth depth image is used to compute per-pixel constraints for a second pass of binocular stereo. During the second pass, we process only the pixels whose depth in the first reconstruction violates the new constraints. If the depth is within an acceptable distance from the smooth surface then we do not re-process the pixel. By tightening the depth constraints, the depth image computed in the second pass has fewer outliers (Figure 7.5, d). We repeat the smoothing process on the new depth image to establish new constraints for the next pass, iterating between stereo matching and constraint tightening until convergence. In practice, we found that three iterations was sufficient for all reconstructions (Figure 7.5, e).

In this approach, we assume that the true surface lies within a small distance of the over-smoothed one in each iteration. This holds true if the surface does not contain very high curvature or sharp features, which is the case for faces. We also assume that outliers have small local support, so that they are removed by the smoothing step.

Pair Merging. After applying iterative binocular reconstruction for each camera pair, we obtain seven corresponding point cloud reconstructions. The point clouds are then merged into a single dense point cloud. On average, we reconstruct approximately 8-10 million points per face. These points are downsampled, filtered and triangulated as described in Chapter 5. Our final meshes have approximately 500K vertices (1 million triangles). Each frame t of a capture sequence can be reconstructed independently, so we perform the reconstructions in parallel. In the following we refer to these initial meshes as G^t. The reconstruction of a selection of frames is shown in Figure 7.6.

7.3 Geometry and Texture Tracking

Given the per-frame reconstructions, the next step is to reconstruct the motion of the face by tracking the geometry and texture over time. We explicitly compute a sequence of compatible meshes without holes, which allows artists to later edit both the geometry and the texture, and to propagate these modifications consistently over time. Given the initial per-frame reconstructions G^t, we would like to generate a set of compatible meshes M^t that have the same connectivity as well as explicit vertex correspondence. That is to say, we desire one mesh that deforms over time. In order to create high-quality renderings, we also require per-frame texture maps T^t that capture appearance changes such as wrinkles and sweating of the actor.

We propose an optical flow based approach for motion reconstruction, as illustrated in Figure 7.7. The basic method works as follows: starting with a single reference mesh M^0, generated by manually cleaning up the first frame G^0 (Section 7.3.1), we compute dense optical flow on the video images and use it in combination with the initial geometric reconstructions G^t to automatically propagate M^0 through time (Section 7.3.2). At each time step, we compute a high-quality 2D face texture T^t from the video images (Section 7.3.3).

In theory this basic method can reconstruct the face motion and produce a temporally consistent animation. However, the practical limitations of optical flow can pose problems. For instance, optical flow is prone to

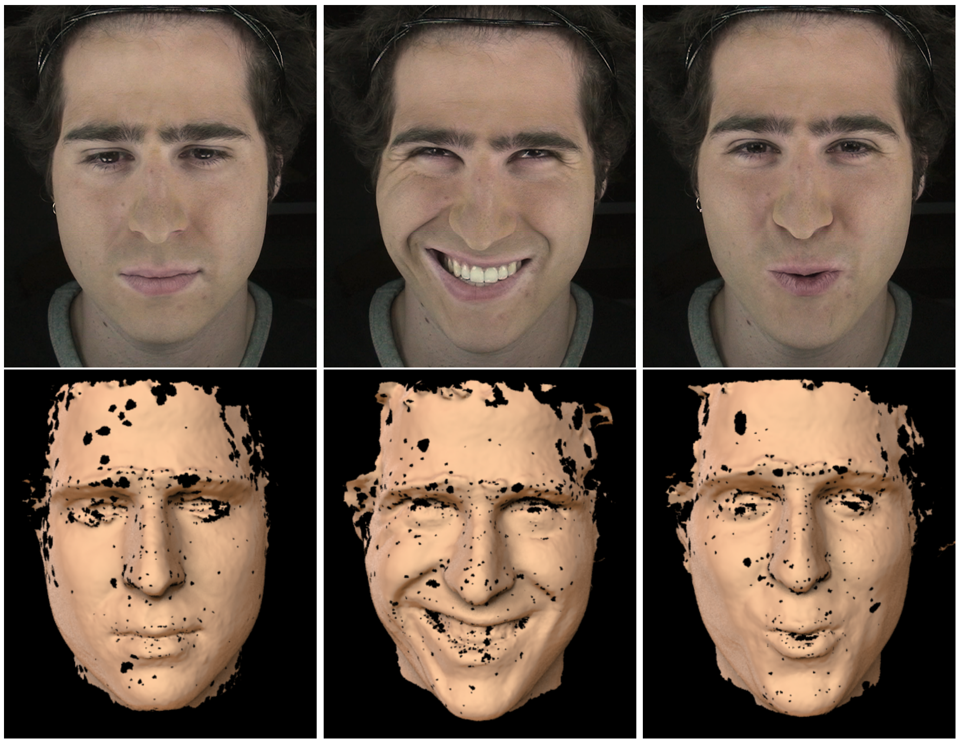


Figure 7.6: Multi-view reconstructions for three different frames.

errors during rapid deformations such as lip movement during speech, and can be inaccurate for a variety of other reasons including occlusions, insufficient image details, and appearance changes such as the formation of wrinkles. Furthermore, since the basic method processes the frames sequentially, even the smallest error will accumulate over time, causing the geometry to drift. Temporal drift is unacceptable, as it will destroy the consistent mapping between frames that we aim to reconstruct. We overcome these issues with two improvements to the basic method, which aim to stabilize the animation through explicit mouth tracking and texture-based drift correction (Section 7.3.4). As a final step, we perform smoothing to remove unwanted noise in the animation (Section 7.3.5).

7.3.1 Reference Mesh

We start each performance with a neutral facial expression, from which we create our reference mesh. The first reconstruction, G^0, (shown in Figure 7.6 left) is manually edited to remove any outliers caused by hair, and then a 2D parameterization of the geometry is computed. We use LSCM [Levy 02] to generate the parameterization because it successfully deals with the holes in our initial reconstruction. The holes are filled in 2D by creating small Delaunay triangulations [Shew 96], and the new 3D geometry is created as a membrane surface, C^1 continuous with the surrounding geometry. Finally, a slit is cut in the mesh for the mouth. The result is the first mesh M^0 of the final sequence. The 2D parameterization computed here will become the domain for the texture map. Note that this parameterization remains constant for the entire sequence. That is to say, as a vertex v_i of the mesh moves over time, its texture coordinates never change.

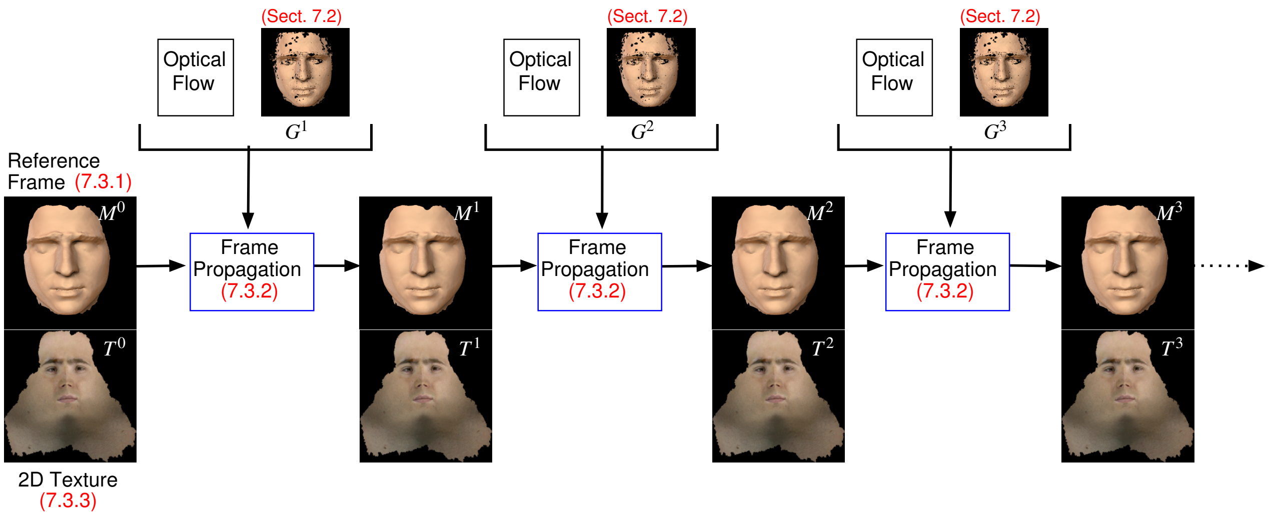


Figure 7.7: Overview of our geometry tracking algorithm.

This observation was made by Rav-Acha et al. [Rav 08] in a technique that uses 2D unwrap-mosaics as a domain for editing videos of deforming surfaces. We will further use the constant parameterization domain for surface tracking and drift-correction, as described in Section 7.3.4.

7.3.2 Frame Propagation (Basic Method)

We compute optical flow [Boug 99] over the whole sequence for each of the 14 video cameras individually. Using this flow and the initial reconstructions G^t, we can now propagate M^0 forward in time to produce our output sequence. The process is illustrated in Figure 7.8, and it proceeds as follows. For each vertex v_i^{t-1} of M^{t-1} we project the vertex onto each camera c in which it is visible (i.e. inside the field of view of and not occluded). Let $p_{i,c}$ be this projected pixel. We then look up the 2D flow vector that corresponds to $p_{i,c}$ and add the flow to get a new pixel location $p'_{i,c}$. Back-projecting from $p'_{i,c}$ onto G^t gives us a guess for the new vertex location, which we call $\vec{v}_{i,c}^t$. The illustration in Figure 7.8 has exaggerated inter-frame motion for better visualization.

We require at least two cameras to agree on the new vertex location. We say that cameras c_1 and c_2 agree if

$$\| \vec{v}_{i,c_1}^t - \vec{v}_{i,c_2}^t \| < 1mm. \tag{7.1}$$

The computed vertex location $\vec{v}_i^t$ is then a weighted average of the n per-camera guesses that agree:

$$\vec{v}_i^t = \sum_{c=1}^{n} w_{i,c}^t \cdot \vec{v}_{i,c}^t, \tag{7.2}$$

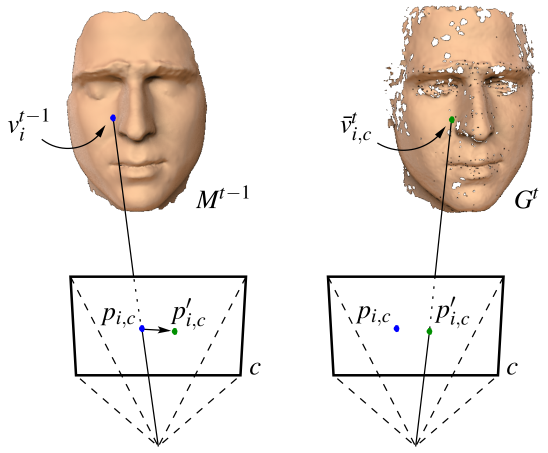


Figure 7.8: Computing vertex positions for the next frame, using per-camera optical flow (with exaggerated motion for visualization).

where $w_{i,c}^t$ is the dot product between the surface normal at $\bar{v}_{i,c}^t$ and the vector from there to c. The difference between the new vertex location and its previous location can be considered a 3D flow vector, which we denote δ_i^t. If not enough cameras agree on a new vertex location, for example if a vertex falls on a hole in the next frame, then the 3D flow from neighboring vertices is interpolated. We assume that the motion of the face is spatially smooth, so neighboring vertices have similar 3D flow. The interpolation is achieved by solving a simple least-squares Laplacian system on the surface for all vertices that were not updated (all updated vertices remain fixed):

$$\min \parallel \Delta\delta_i^t \parallel^2 . \tag{7.3}$$

Finally, we apply regularization to the mesh in order to avoid possible triangle-flips and remove any unwanted artifacts that may have been present in the initial reconstruction. Following the regularization method of de Aguiar et al. [Agui 08], we again solve a least-squares Laplacian system using cotangent weights and the current positional constraints $\bar{v}_i^t$. Thus, we generate the final mesh M^t by minimizing

$$\arg\min_{v^t}\{\parallel v_i^t - \bar{v}_i^t \parallel^2 + \alpha \parallel Lv^t - Lv^0 \parallel^2\}, \tag{7.4}$$

where L is the cotangent Laplacian matrix. The parameter α controls the amount of regularization, and is set to 100 for all reconstructions.

7.3.3 Computing 2D Texture

At each time step, in addition to reconstructing geometry, we also compute a high-resolution 2D texture T^t for rendering. Since we do not use markers or face paint, our texture images are rich in detail, containing per-frame appearance changes due to, for example, blushing or sweating of the actor. As pointed out by Borshukov et al. [Bors 03], these textural variations are very important for creating believable facial renderings.

All of our 14 HD cameras are used to compute a single texture that covers the entire surface. This allows us to create very high-resolution textures, in the order of 8-10 megapixels, for extreme zooms (see Figure 7.17). For other purposes, lower resolution textures may be sufficient (e.g. 900×900 as used in most of our examples). The domain of the texture image is given by the 2D parameterization of the mesh (Section 7.3.1). Every vertex of the mesh has unique 2D coordinates in the parameter domain, yielding a one-to-one mapping between 2D and 3D mesh triangles.

To compute the texture for frame t, we start by projecting each triangle of M^t onto the camera that observes it best, as determined by the dot product between the triangle normal and the camera direction. The camera pixels corresponding to the projection are then copied to the corresponding 2D triangle in the texture domain. Figure 7.9 shows the computation of a face texture. We visualize the contribution from each camera as a grayscale *patch* image in Figure 7.9 (middle-left). Figure 7.9 (middle-right) shows the initial texture result after copying the pixels from the camera images. Since the cameras are not radiometrically calibrated, different texture patches can have drastically different skin tones. To compute the final texture, shown in Figure 7.9 (far right), we apply Poisson image editing [Pere 03] similar to Mohammed et al. [Moha 09]. We start with the largest patch and iteratively add adjacent patches until the texture image is complete. For each new patch we compute x- and y-gradients inside the patch and solve a Poisson equation to find a new patch that matches the gradients as closely as possible, while also obeying the boundary conditions set by other completed patches. Finally, in order to have temporally consistent textures, we use the previous texture T^{t-1} as per-pixel soft constraints when solving the Poisson equation.

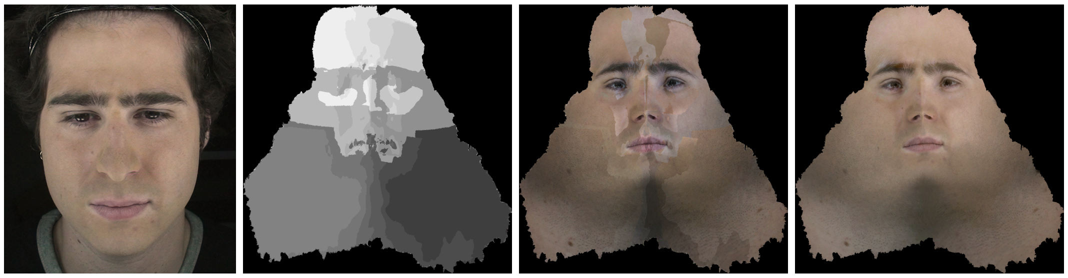


Figure 7.9: 2D texture generation. From left to right: reference image, camera contribution image, initial texture, final texture.

Two additional textures for the same sequence are shown in Figure 7.10. Notice how the texture domain remains fixed even though the 3D face undergoes substantial deformations, including opening of the mouth. The only differences in the textures is changes in skin appearance caused by wrinkles, blushing or sweating of the actor.

7.3.4 Tracking Enhancements

In general, the basic optical flow-based tracking technique described so far produces realistic animations of face deformation. For most of the face, optical flow vectors are both dense and accurate, since our capture setup provides natural high-resolution surface features which easily guide the flow computation. However,

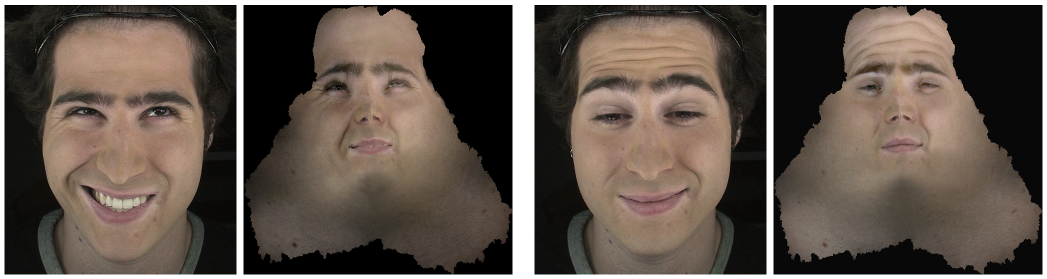


Figure 7.10: Texture results for two different frames, including reference images.

optical flow can fail during very fast motion such as rapid mouth deformations, and minor inaccuracies can accumulate over time leading to temporal drift. We resolve these problems automatically by enhancing the basic technique with an explicit mouth tracking algorithm, along with a method for detecting and correcting temporal drift.

Mouth Tracking. To perform automatic mouth tracking, we introduce positional constraints for a sparse set of points around the mouth at each time step. The positional constraints are computed in image-space, and thus do not map directly to mesh vertices. Instead, they are incorporated as barycentric constraints on mesh triangles.

The constraint points are determined by tracking the mouth in a single camera (either one of the two *green* cameras in Figure 7.3). We perform edge detection [Cann 86] on each frame within a user-specified region-of-interest (ROI). The region should contain the mouth throughout the sequence but avoid other edges caused by surrounding wrinkles or the silhouette of the face. If the sequence contains too much global face motion to contain the mouth in a single ROI then mouth tracking can be performed in temporal segments with different ROIs. For each frame we perform a simple analysis of the detected edges to locate the mouth. Figure 7.11 shows a few different frames of mouth tracking. Detected edges are shown in white, and the ROI is indicated by the blue rectangle. If we consider the image as a set of rows and columns, we start by choosing the two corners of the mouth (shown as red points) as the minimum and maximum columns that contain an edge pixel. We then detect the top and bottom lips by uniformly sampling a sparse number of columns between the mouth corners, and selecting the minimum and maximum rows at each column that contain edge pixels (shown as green points). Empirically we found that 30 sample columns were sufficient. These 62 pixels then become the mouth constraints for this frame. We process each frame in the same manner, yielding an explicit temporal correspondence between each of the individual constraints. Since edge detection is notoriously unstable, we smooth the constraints both spatially and temporally to remove outliers. Although our mouth tracking technique is rather simple, we found the results to be quite robust, as we show in the bottom row of Figure 7.11.

We back-project the constrained pixels into the reference frame M^0 to determine the set of constraint mesh triangles and the corresponding barycentric coordinates for each of the mouth constraint points. We encode the barycentric coordinates in a sparse matrix B which has similar structure to the Laplacian matrix, except that it contains rows only for vertices that are adjacent to a constraint triangle. Let P^0 be the set of 3D

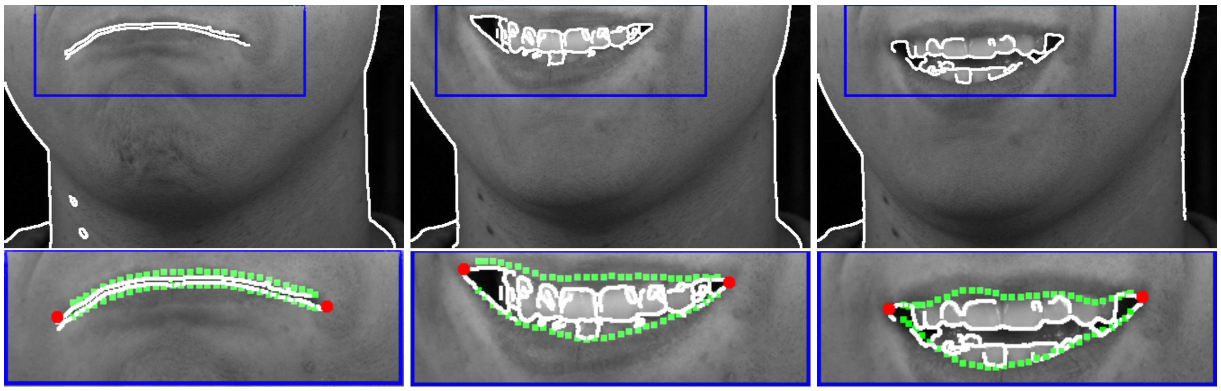


Figure 7.11: Mouth tracking through edge detection. The green and red points become constraints in geometry tracking.

constraint points determined from the back-projection, then

$$P^0 = Bv^0. \tag{7.5}$$

Throughout the sequence B remains fixed. The per-frame mouth constraints are used to compute 3D constraint points P^t by back-projecting onto the initial reconstructions G^t. The mouth constraints guide the regularization from Section 7.3.2, replacing Equation 7.4 with

$$\arg\min_{v^t}\{\| v_i^t - \tilde{v}_i^t \|^2 + \alpha \| Lv^t - Lv^0 \|^2 + \beta \| Bv^t - P^t \|^2\}, \tag{7.6}$$

where β controls how much we constrain the mouth. We use a high weight, $\beta = 10^4$, since the mouth can deform quite rapidly, causing large errors in the basic optical flow approach. Our mouth tracking procedure alleviates these errors, and thus is an essential part of our method for generating realistic facial animations.

Drift Correction. It is well-known that optical-flow based tracking methods suffer from accumulation of error, known as drift [DeCa 00, Bors 03]. DeCarlo and Metaxas [DeCa 96] solve this problem by combining optical flow with edge information, and Borshukov et al. [Bors 03] rely on manual intervention. A key feature of our method is that we are able to detect and correct drift in the 3D animation automatically, using the texture domain of the faces.

Drift typically occurs because optical flow is computed between successive video frames only. If it were possible to accurately compute flow between the first video image and every other frame, there would be no accumulation of error. Unfortunately, temporally distant video images in a capture sequence are usually too dissimilar to consider this option. In our case, however, the texture domain of the mesh remains constant over time, which means that the computed per-frame texture images are all very similar. Any temporal drift in the 3D geometry appears as a small 2D shift in the texture images, which can easily be detected, again by optical flow.

To incorporate drift correction, we employ a simple modification to the basic tracking method described in Section 7.3.2. After computing the geometry M^t and texture T^t for a given frame, we compute optical flow between the textures T^0 and T^t. This flow (if any is detected) is then used to update M^t on a per-vertex basis using the direct mapping between the geometry and the texture. Any shift in texture space becomes a 3D shift along the mesh surface. After updating the vertices to account for drift we apply regularization again (Equation 7.6), to avoid possible triangle flips.

The only problem that remains is that, if significant appearance changes such as wrinkles have occurred in the current frame, optical flow between T^0 and T^t can fail, resulting in large flow errors. However, since we expect drift to appear gradually, the flow between T^0 and T^t should never be more than a few pixels. Larger flow vectors are discarded as outliers. Still, face regions that contain these appearance changes may incur drift, as wrinkles can be present for a significant number of frames. In these regions, we detect drift more locally by computing the flow between T^{t-k} and T^t, and updating the geometry accordingly. We choose k to be small, so that both frames $(t-k)$ and t contain similar appearance, such as the same wrinkles, allowing flow to be computed accurately. On the other hand, k must be large enough so that drift can accumulate and be detected. In all reconstructions, we found that $k=5$ was an appropriate trade-off. By performing local drift correction we are, in effect, only slowing down the drift accumulation rather than removing it. However, this approach does stabilize the animation until the wrinkles disappear, at which time normal drift correction is automatically resumed.

7.3.5 Post-Processing

As a final step, we post-process the sequence to provide a smooth, realistic facial animation.

Saliency-Based Smoothing. As with any stereo reconstruction method, spatial noise can appear in the resulting geometry due to, for example, slight inaccuracies in camera calibration (see Figure 7.12-bottom left). We wish to smooth the face meshes to remove this noise, but avoid removing the spatial features and wrinkles that define the face. To accomplish this, we introduce saliency-based smoothing, a technique for smoothing less-salient regions of the face while preserving more-salient features. Saliency is computed in the texture image T^t through a simple analysis of local histograms. Kadir and Brady [Kadi 01] remark that areas of an image with high saliency tend to have flatter distributions in the local histogram of intensities. Following this principle, we mark a pixel in T^t as non-salient if its local histogram contains a single strong peak. All other pixels are salient. We compute histograms in local 15x15 windows, quantized to 8 intensity bins and use simple thresholding to determine if a peak exists. Figure 7.12 (top row) shows one of the texture images and the computed saliency mask, where salient pixels are white. The saliency mask is then used to constrain salient vertex positions in a Laplacian smoothing step. The result is shown in the bottom row of Figure 7.12. Notice that the eyebrows, mouth and cheek deformation were not affected by the smoothing.

Relaxing the Eye Geometry. As we saw in the initial reconstructions (Figure 7.6) and again in Figure 7.12, the eye regions are not reconstructed well. This is because eyes are too specular, and surrounding eye-lashes are thin hairs that only add noise to the surface. We alleviate this problem by performing localized smoothing in the eye regions, as shown in Figure 7.13. These eye regions are marked manually in the first frame and are then propagated automatically for the rest of the sequence.

Temporal Smoothing. We end with a single pass of Gaussian smoothing in the temporal domain to prevent temporal flicker. In practice we found that temporal noise was minimal, so we use a small smoothing

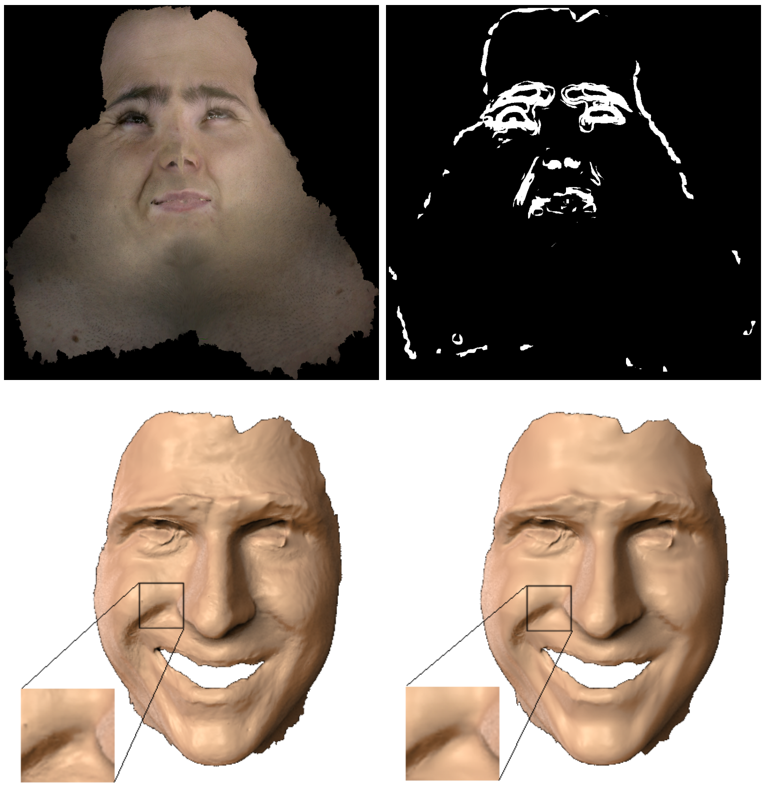


Figure 7.12: Saliency-based spatial smoothing. Top row: an input texture and the saliency map. Bottom row: before and after smoothing.

radius of only three frames.

After all post-processing, 2D textures are re-computed since the face geometry has changed.

7.4 Results

We now show results for a number of facial performances given by different people. All of our results were rendered with Renderman, using the MakeHuman skin shader in Pixie[5].

Figure 7.18 contains six frames from the first sequence, rendered in a number of different ways to highlight the results. The top row is the reference footage. In the second row we show the geometry without a texture map. Here we can see the exact geometric details that form each facial expression. The third row is rendered using a static checkerboard texture. This visualization shows how the skin stretches and compresses during deformation, for example when raising the eyebrows in the second image from the left. The stability of the checker pattern over time also indicates that we have achieved a drift-free reconstruction. The fourth row shows our high-quality rendering using the captured per-frame textures. Finally, the last row demonstrates

[5]www.makehuman.org, www.renderpixie.com

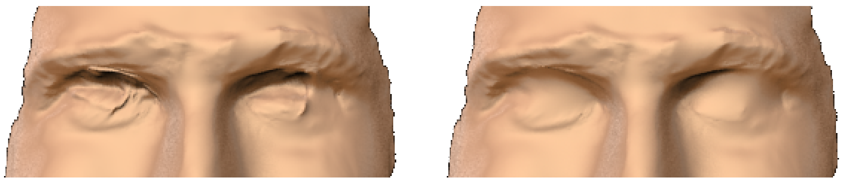


Figure 7.13: Local smoothing to improve the eye regions.

how virtual makeup can be applied to the sequence. Here, an artist would edit the makeup in the first frame, and the edits would be propagated to the rest of the sequence automatically, similar to the video editing technique of Rav-Acha et al. [Rav 08]. Once a facial performance has been captured, we can also render it from arbitrary viewpoints or lighting conditions, as we show in Figure 7.19.

In the fifth column of Figure 7.18, note that the mouth of the actor is open, resulting in a hole where the teeth should be. Like most methods for facial performance capture, we do not reconstruct the inside of the mouth. Although this effect can be distracting, making the entire face appear different from the reference image, we illustrate that the face reconstruction is still accurate by manually compositing the teeth from the reference frame into the result, as shown in Figure 7.14. The final result including the teeth now matches the reference frame and is very compelling, indicating an area of future work to simultaneously reconstruct time-varying teeth models with the rest of the face.

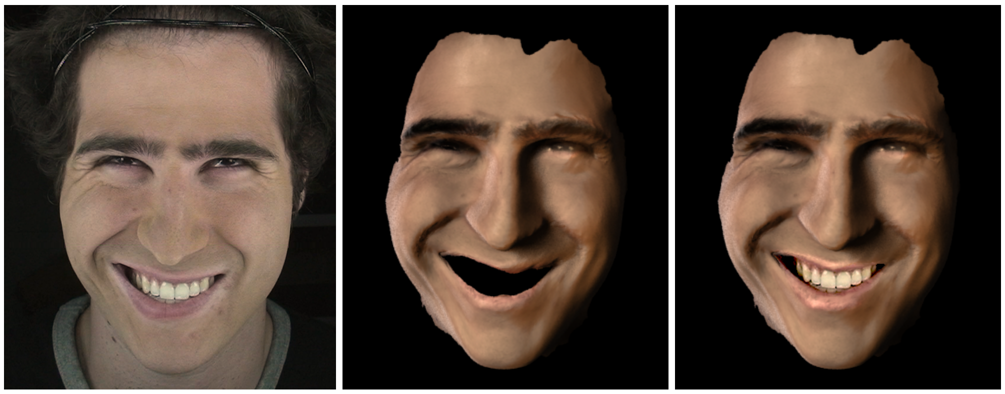


Figure 7.14: Adding the teeth creates a compelling result, indicating an area for future research.

We show the versatility of our approach by capturing two other actors, one male and one female. Results are shown in Figure 7.15 and Figure 7.16, including the pure geometry result and the high-quality textured rendering. Two extreme zoom renderings are shown in Figure 7.17(middle and right), using a 10 megapixel texture that we reconstructed from the video images.

7.5 Discussion

In this Chapter we present a purely passive method for facial performance capture. Unlike previous methods that require markers, face paint, structured light, or expensive hardware, we use only a camera array and uniform illumination. Nonetheless, we are able to reconstruct high resolution, time-varying meshes captured at 30 frames per second. The absence of markers and structured light allow us to capture detailed, per-frame textures and create high-quality renderings.

One of the keys to our approach is our novel high-resolution acquisition setup. We are able to use natural skin blemishes, hair follicles and pores both for establishing detailed geometric reconstructions, and also for tracking the face over time. Our geometry and texture tracking method is fully automatic, and includes robust temporal drift correction. While the small facial details are visible in the video images, spatial geometry regularization and temporal smoothing prevent us from reconstructing the finest pore-scale geometry. These smoothing steps are required to overcome inaccuracies in the initial reconstructions and in the optical flow vectors. However, we do provide much higher resolution than previous passive methods [Li 93, Essa 96, DeCa 96, Pigh 99]. Also note that fine wrinkles and pores can be added to the geometry in a post-process similar to Beeler et al. [Beel 10], and we consider this future work.

Our method is versatile, which we demonstrate by reconstructing three different facial performances given by different actors, and including very different deformations. To our knowledge, we present the first fully-automatic technique to reconstruct high-quality facial performances without the need for markers, face paint, or structured light.

The main limitation of our technique is that very fast motion can lead to incorrect geometry tracking due to motion blur and inaccurate optical flow. Furthermore, since our method processes frames sequentially, an error in tracking could cause an early termination of the face sequence. Additionally, our method is not designed to reconstruct facial hair, and we require manual processing of one frame of the sequence to build the reference mesh.

In the future, we plan to explore methods for automatically completing the face model by capturing the teeth. In addition, correctly capturing the eye geometry, including eyelashes and eyebrows, would produce even more realistic results, particularly for high-quality specular rendering of the eyes.

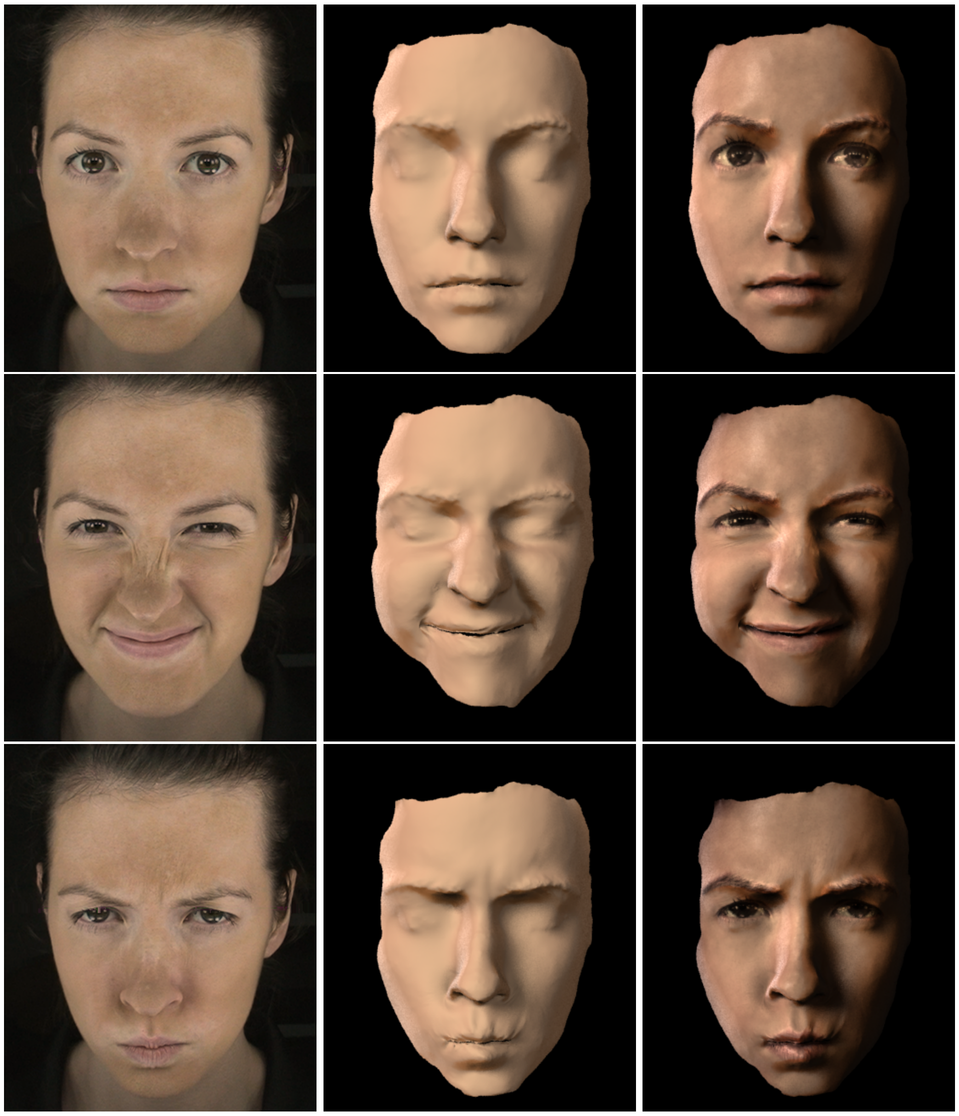


Figure 7.15: Capture results for another sequence, including the reference frames (left), pure geometry result (center), and high-quality rendering with texture (right).

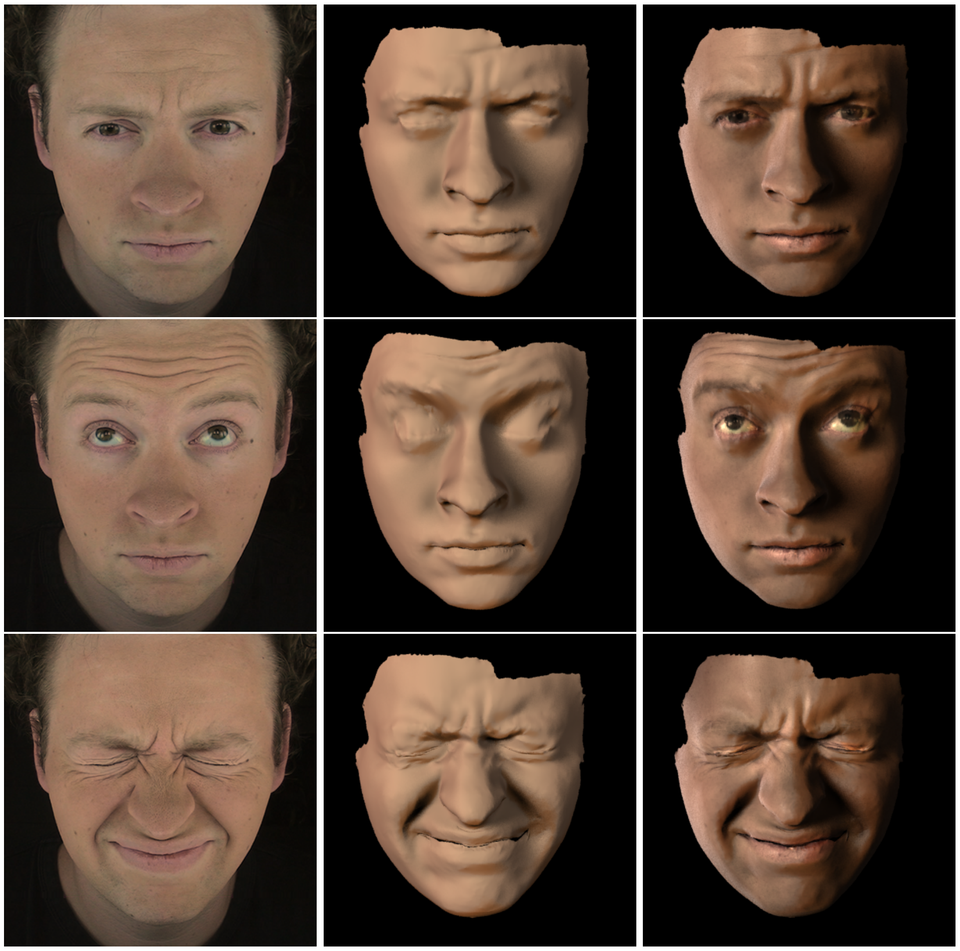


Figure 7.16: Capture results for yet another sequence, including the reference frames (left), pure geometry result (center), and high-quality rendering with texture (right).

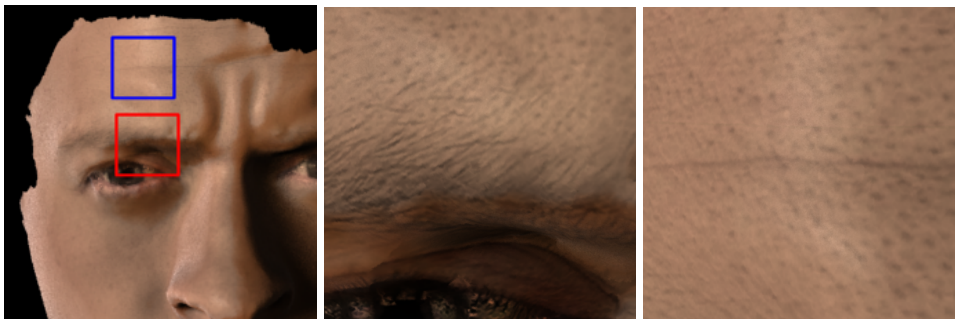


Figure 7.17: 10 megapixel textures allow extremely close zoom renderings (middle and right).

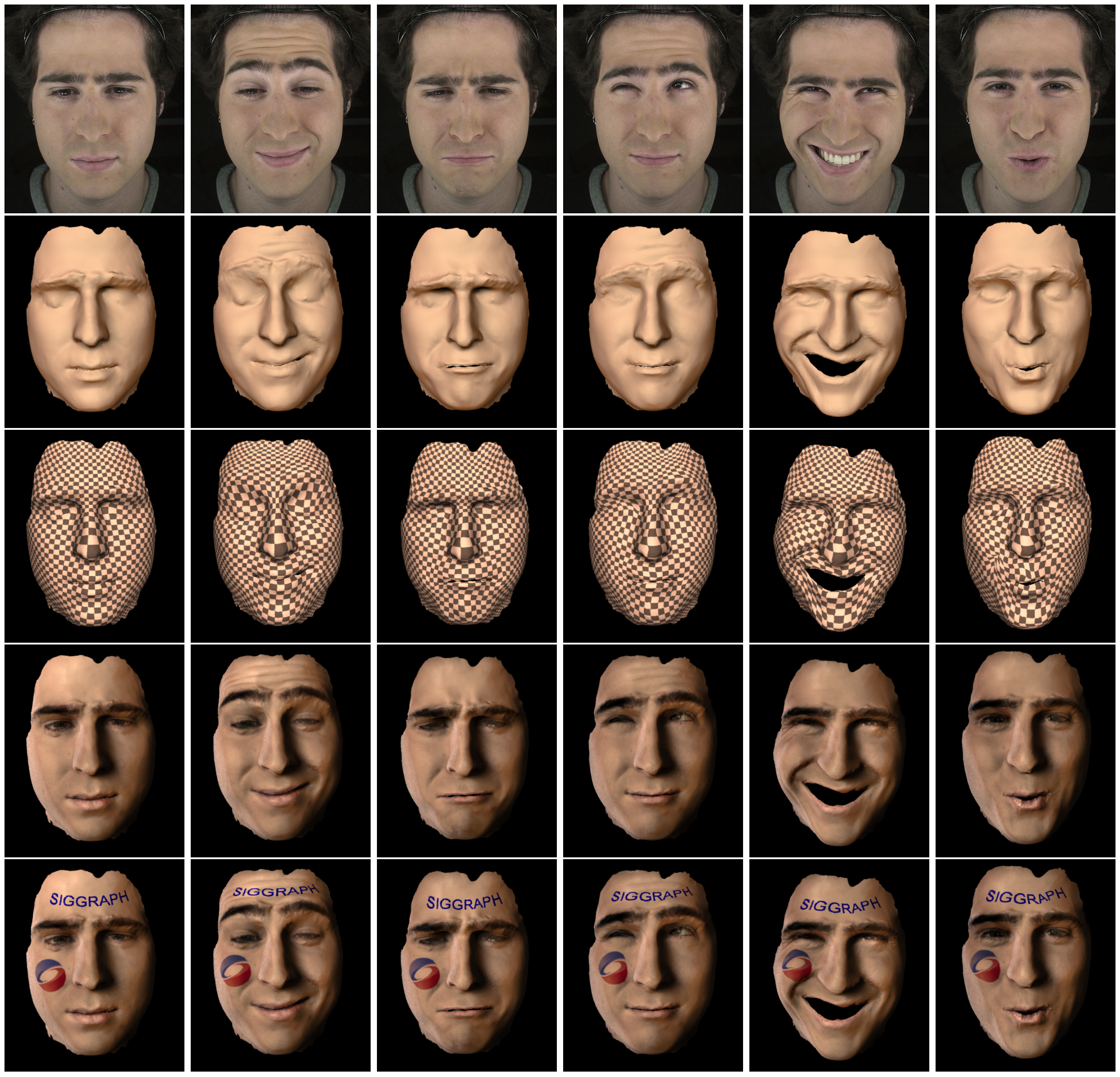


Figure 7.18: Capture results for one sequence including the reference footage (top row), pure geometry result (2nd row), skin stretch visualization (3rd row), high-quality rendering with texture (4th row), and virtual makeup (bottom row).

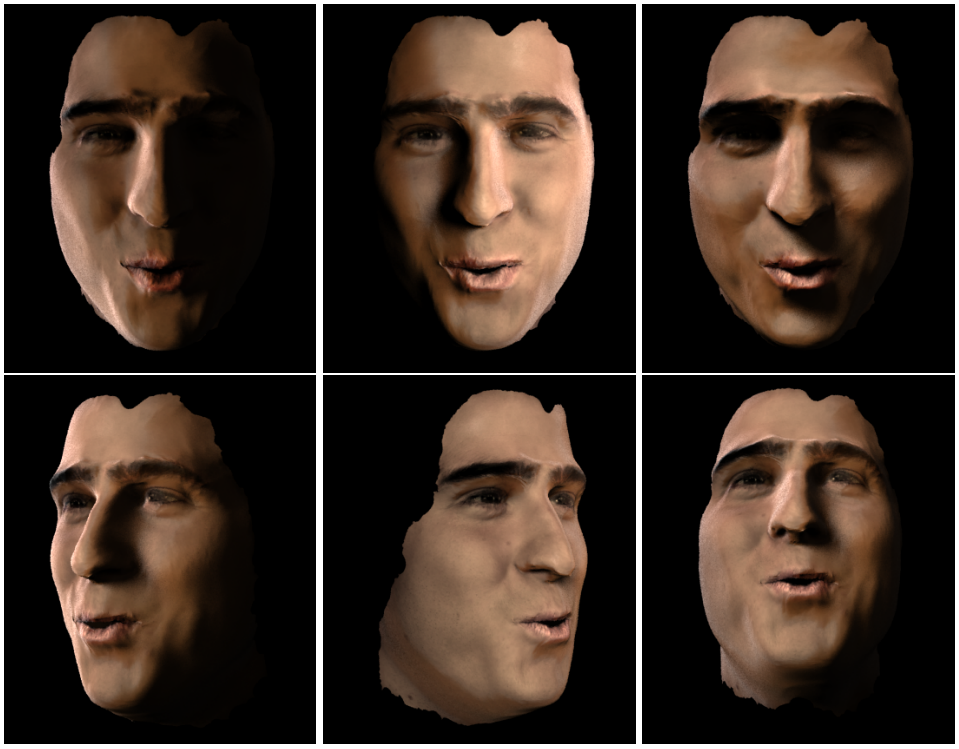


Figure 7.19: Realistic renderings under various different illuminations and viewpoints.

Chapter 8

Conclusion

In this thesis, we advance the state of the art in multi-view reconstruction of deforming surfaces by presenting markerless techniques for capturing the shape and motion of both garments and human facial performances. To make this possible, the contributions of this thesis also include new methods for multi-camera calibration and synchronization, and a robust and efficient method for multi-view stereo reconstruction. Together these algorithms form a hierarchy for time-varying deformable surface capture.

8.1 Summary of Contributions

Our main camera synchronization algorithm presented in Chapter 3 uses stroboscopic illumination to optically synchronize consumer camcorders and remove the rolling shutter distortions that are common with these cameras. This approach allows us to employ inexpensive, portable cameras for multi-view reconstruction.

In Chapter 4 we discussed how calibrating multiple cameras in pairs using epipolar geometry can yield more accurate camera calibrations, directly increasing the quality of stereo reconstructions.

We presented a new algorithm for multi-view stereo reconstruction that is both accurate and efficient in Chapter 5. By using a scaled-window matching technique between pairs of adjacent cameras, we can gather more surface point samples than using traditional square-window matching. The extra data points help with outlier removal and robust surface reconstruction. To our knowledge, our algorithm is the only technique to excel at both accuracy and efficiency, making it the ideal tool for multi-view reconstruction of deforming surfaces where hundreds or even thousands of video frames must be reconstructed. Our technique is already creating an impact in industry, as we are currently licensing this technology to a company that creates 3D reconstructions for visual effects, video games and reverse engineering.

The ability to capture and parameterize deforming surfaces without markers, special painted textures, or structured light patterns allows us to reconstruct the fabric-specific motion of off-the-shelf garments (Chapter 6), and also capture high-resolution facial geometry with dynamic texture maps that contain actor-specific performance details such as face wrinkling, blood flow and sweating (Chapter 7). This research has the potential for high impact on both the film and video game industries, as most current commercial approaches rely on marker-based methods. Markers are not only less comfortable for the actor, but they also prevent the capture of certain cloth fabrics where markers cannot be accurately printed (e.g. fleece), and they inhibit the reconstruction of accurate time-varying texture maps. We believe that the algorithms presented in this thesis have the potential for a large impact on the research community as well, since we are among the first researchers to present solutions for markerless motion capture, strengthening a new field of research that is now becoming more popular every year.

The hierarchy of algorithms that define this thesis form a robust base of techniques that can benefit other applications, even outside the scope of this thesis. In fact, I have been involved in four additional projects that all make use of one or more algorithms in the hierarchy. The first is a method for tomographic reconstruction of transparent objects [Trif 06], which required a unique solution for calibration of multiple camera viewpoints using a calibration grid. Our stroboscopic illumination technique was used to synchronize multiple cameras for 3D capture of non-stationary gas flows [Atch 08]. Our algorithm for markerless garment capture was extended to produce realistic cloth wrinkling using space-time data-driven deformation [Popa 09]. And finally, our multi-view stereo reconstruction pipeline has been used to generate data for an algorithm that produces globally consistent space-time reconstructions [Popa 10]. In addition to the techniques presented in this thesis, the data sets we have generated can also benefit others. Our data sets have been made publicly available for researchers to evaluate their own algorithms. As an example, Sharf et al. have employed our garment capture point clouds to validate their volumetric technique for reconstructing deforming objects using incompressible flow [Shar 08]. To date, the algorithms and data sets presented in this thesis are continuously being employed in new research projects that push the state of the art in 3D reconstruction.

8.2 Short-term Future Work

This thesis opens up a number of areas for future work. Here we discuss some improvements that can be made to the current algorithms in a short-term time frame.

Low Level Algorithms. At the lowest level, it would be interesting to further explore pair-wise and multi-camera calibration using epipolar geometry. Currently we use the calibration method of Zhang [Zhan 00], which minimizes the reprojection error to produce multiple external parameter choices, and then we select the best calibration using our rectification error measure. An alternative would be to modify the method of Zhang to minimize the rectification error directly.

Multi-View Stereo. Future work for our multi-view stereo algorithm includes performance enhancements by optimizing the window matching technique. Although our algorithm is among the most efficient already, we believe further speed increases are possible. Currently, given two images, each pixel in primary image is matched independently to the reference image, including some redundant computations. As an alternative, we can compute correlation scores for all pixels at once, testing each disparity offset by shifting the reference image over the primary image and performing efficient local filtering. This technique would also lead itself to a GPU implementation, which would even further increase efficiency.

Garment Capture. There are two main drawbacks of our markerless garment capture method that could be explored in future work. First, since spatial noise in the reconstruction requires us to smooth the models, the very fine-scale surface details and wrinkles tend to be lost. Although we have already started to examine this problem and have proposed a post-process to wrinkle the captured garments [Popa 09], an alternative would be to modify the initial reconstruction so that noise is removed in a more elegant, perhaps spatio-temporal, manner.

The second limitation is that garments must not come into contact with themselves during capture, e.g. if the sleeve of a jacket were to touch the torso then the resulting model would become connected at the contact point. This drawback has also been examined and one solution has been developed that uses temporal information to maintain globally consistent mesh topology [Popa 10].

Facial Performance Capture. Finally, with respect to facial performance capture, we plan to explore methods for automatically completing the face model by capturing the teeth. In addition, correctly capturing the eye geometry, including eyelashes and eyebrows, would produce even more realistic results, particularly for high-quality specular rendering of the eyes.

The face wrinkle geometry could also be improved. By using the fact that the texture domain remains constant throughout a capture sequence, the difference between the texture images for two frames would primarily consist of shading changes between the frames. Since we use uniform illumination, the shading changes will almost directly correspond to the skin wrinkles, which can be localized in the texture images and mapped directly to the face model. Analysis of the difference in texture images can lead to the required difference in local geometry between frames, using a variant of shape-from-shading [Horn 89], and the shape of the wrinkles can be improved.

A final area for future work would be to “learn” where wrinkles form on a specific actor through analysis of one or more sequences. This knowledge could be used for adaptively downsampling the geometry where high-resolution is never required. Understanding how a particular person’s face deforms would also be useful for tasks such as performance transfer (i.e. deforming one face to match the performance of another person) or performance editing by an artist, all while keeping the learned wrinkles consistent.

8.3 Long-term Future Work

One of the main limitations of space-time capture methods is that it is challenging to evaluate the resulting reconstructions. This is typically due to a lack of ground truth data. Currently, it is not possible to recover ground truth models for deforming surfaces such as garments and human faces. In the future, high-accuracy real-time laser scanners could be one option. In the mean time, validation could be performed using synthetic datasets. A deforming 3D ground truth model can be rendered from multiple viewpoints, and the rendered images could be used as input for reconstruction. However, this does not solve the problem of comparing results between different capture methods. Currently, each method tends to rely on a specific hardware setup, often with different sensor types, and different lighting environments. This makes the comparison of different space-time capture methods very difficult.

A second limitation of most capture methods is the inability to re-animate the motion sequence. For example, performance capture techniques would become even more attractive if there was a simple way to change the performance post-capture. As we have shown, one can often vary certain parameters of the animation such as texture and reflectance, or even small geometry changes are possible [Vlas 08]. However, it remains a challenge to automatically generate new motions from the capture data while deforming the object in a realistic way. In order to allow re-animation of captured geometry, the capture process could be coupled with physical simulation in an attempt to learn the physical parameters and forces that caused the observed deformation. In this way, new animations could be obtained by applying new forces to the geometry while using the learned physical parameters, obtaining new physically correct deformations. This type of approach has already been explored in the context of cloth by Bhat et al. [Bhat 03] for small swatches of fabric, but it has yet to be extended to full garments, human faces or other deforming surfaces. I believe that the future of performance capture will include techniques for facilitating re-animation of the capture sequences.

While the work presented in this thesis aims to advance the current state of markerless reconstruction techniques for deforming surfaces, a general solution for fully-automatic capture of *any* scene remains an unsolved problem. Most current reconstruction methods target well-defined surfaces that exhibit Lambertian reflectance and smooth motion over time. However, our world is filled with complex shapes and objects with complex reflectance properties, and so new methods must be explored. Some initial research has already been performed, yielding specific reconstruction techniques some of these objects. Examples include plants [Quan 06], trees [Tan 07], hair [Pari 08], surfaces with anisotropic [Holr 08] or specular reflectance [Chen 06], transparent objects [Hull 08], fluids [Wang 09], fire [Ihrk 04] and gases [Atch 08]. However there is much work yet to be done, since no single unified solution exists for reconstructing an arbitrary time-varying scene. In the future, we may approach a solution to this problem with a meaningful combination of existing object-specific techniques, new acquisition hardware, multiple shape and motion priors, and perhaps the fusion from numerous different sensor types. Looking even further, a universal capture method could be extended to include semantics about the scene and object recognition, which in turn could be used to improve the quality and accuracy of the reconstructed geometry and scene motion in a cyclic fashion.

To conclude, this dissertation considers many aspects of the pipeline for markerless 3D reconstruction of temporally deforming surfaces from multiple cameras. We propose a number of different algorithms ranging from low-level camera calibration and synchronization, to multi-view stereo reconstruction and high-level capture applications, demonstrating acquisition results for deforming garments and human facial performances. We hope that this research will inspire further innovative ideas in the field of 3D reconstruction and performance capture.

Bibliography

[Agui 07] E. de Aguiar, C. Theobalt, C. Stoll, and H.-P. Seidel. “Marker-less Deformable Mesh Tracking for Human Shape and Motion Capture”. In: *Proc. CVPR*, pp. 1–8, June 2007.

[Agui 08] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. “Performance Capture from Sparse Multi-view Video”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 98, 2008.

[Ahme 08a] N. Ahmed, C. Theobalt, P. Dobrev, H.-P. Seidel, and S. Thrun. “Robust Fusion of Dynamic Shape and Normal Capture for High-quality Reconstruction of Time-varying Geometry”. In: *Proc. CVPR*, p. , 2008.

[Ahme 08b] N. Ahmed, C. Theobalt, C. Roessl, S. Thrun, and H.-P. Seidel. “Dense Correspondence Finding for Parameterization-free Animation Reconstruction from Video”. In: *Proc. CVPR*, p. , 2008.

[Ait 07] O. Ait-Aider, A. Bartoli, and N. Andreff. “Kinematics from lines in a single rolling shutter image”. In: *Proc. of CVPR*, pp. 1–6, 2007.

[Alex 09] O. Alexander, M. Rogers, W. Lambeth, M. Chiang, and P. Debevec. “The Digital Emily project: photoreal facial modeling and animation”. In: *ACM SIGGRAPH Courses*, pp. 1–15, 2009.

[Angu 05] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. “SCAPE: shape completion and animation of people”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, pp. 408–416, 2005.

[Atch 08] B. Atcheson, I. Ihrke, W. Heidrich, A. Tevs, D. Bradley, M. Magnor, and H.-P. Seidel. “Time-resolved 3D Capture of Non-stationary Gas Flows”. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, Vol. 27, No. 5, p. 132, 2008.

[Aucl 08] A. Auclair, L. Cohen, and N. Vincent. “Using Point Correspondences Without Projective Deformation For Multi-View Stereo Reconstruction”. In: *ICIP*, 2008.

[Bake 81] H. Baker and T. Binford. “Depth from Edge and Intensity Based Stereo”. In: *IJCAI*, pp. 631–636, 1981.

[Beel 10] T. Beeler, B. Bickel, R. Sumner, P. Beardsley, and M. Gross. “High-Quality Single-Shot Capture of Facial Geometry”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. , 2010.

[Besl 92] P. J. Besl and N. D. McKay. “A method for registration of 3-d shapes”. *IEEE Trans. on PAMI*, Vol. 14, No. 2, pp. 239–256, 1992.

[Bhat 03] K. Bhat, C. Twigg, J. Hodgins, P. Khosla, Z. Popovic, and S. Seitz. "Estimating Cloth Simulation Parameters from Video". In: *Proc. SCA*, pp. 37–51, 2003.

[Bick 07] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, and M. Gross. "Multi-scale capture of facial geometry and motion". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 33, 2007.

[Blan 03] V. Blanz, C. Basso, T. Vetter, and T. Poggio. "Reanimating Faces in Images and Video". *Computer Graphics Forum (Proc. Eurographics)*, Vol. 22, No. 3, pp. 641–650, 2003.

[Bors 03] G. Borshukov, D. Piponi, O. Larsen, J. P. Lewis, and C. Tempelaar-Lietz. "Universal capture: image-based facial animation for "The Matrix Reloaded"". In: *ACM SIGGRAPH Sketches & Applications*, 2003.

[Boub 05] T. Boubekeur, P. Reuter, and C. Schlick. "Visualization of Point-Based Surfaces with Locally Reconstructed Subdivision Surfaces". In: *Shape Modeling International*, June 2005.

[Boub 06] T. Boubekeur, W. Heidrich, X. Granier, and C. Schlick. "Volume-Surface Trees". *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2006)*, Vol. 25, No. 3, pp. 399–406, 2006.

[Boug 99] J.-Y. Bouguet. "Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the Algorithm". Tech. Rep., Intel Corporation, Microprocessor Research Labs, 1999.

[Brad 08a] D. Bradley, T. Boubekeur, and W. Heidrich. "Accurate Multi-View Reconstruction Using Robust Binocular Stereo and Surface Meshing". In: *Proc. CVPR*, p. , 2008.

[Brad 08b] D. Bradley, T. Popa, A. Sheffer, W. Heidrich, and T. Boubekeur. "Markerless Garment Capture". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 99, 2008.

[Brad 09] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. "Synchronization and Rolling Shutter Compensation for Consumer Video Camera Arrays". In: *International Workshop on Projector-Camera Systems (PROCAMS 2009)*, 2009.

[Brad 10a] D. Bradley and W. Heidrich. "Binocular Camera Calibration Using Rectification Error". *IEEE Conference on Computer and Robot Vision (CRV)*, p. , 2010.

[Brad 10b] D. Bradley, W. Heidrich, T. Popa, and A. Sheffer. "High Resolution Passive Facial Performance Capture". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. , 2010.

[Camp 08] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. "Using Multiple Hypotheses to Improve Depth-Maps for Multi-View Stereo". In: *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pp. 766–779, 2008.

[Cann 86] J. Canny. "A computational approach to edge detection". *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 8, No. 6, pp. 679–698, 1986.

[Carc 04] R. L. Carceroni, F. L. C. Padua, G. A. M. R. Santos, and K. Kutulakos. "Linear sequence-to-sequence alignment". In: *Proc. of CVPR*, pp. 746–753, 2004.

[Casp 06] Y. Caspi, D. Simakov, and M. Irani. "Feature-based sequence-to-sequence matching". *IJCV*, Vol. 68, No. 1, pp. 53–64, 2006.

[Chen 06] T. Chen, M. Goesele, and H.-P. Seidel. "Mesostructure from Specularity". In: *CVPR*, pp. 17–22, 2006.

[Curl 96] B. Curless and M. Levoy. "A Volumetric Method for Building Complex Models from Range Images". In: *SIGGRAPH*, 1996.

[Dai 06] C. Dai, Y. Zheng, and X. Li. "Subframe video synchronization via 3D phase correlation". In: *Proc. of ICIP*, pp. 501–504, 2006.

[DeCa 00] D. DeCarlo and D. Metaxas. "Optical Flow Constraints on Deformable Models with Applications to Face Tracking". *Int. Journal of Computer Vision*, Vol. 38, No. 2, pp. 99–127, 2000.

[DeCa 96] D. DeCarlo and D. Metaxas. "The Integration of Optical Flow and Deformable Models with Applications to Human Face Shape and Motion Estimation". In: *CVPR*, p. 231, 1996.

[Dela 08] A. Delaunoy, E. Prados, P. Gargallo, J.-P. Pons, and P. Sturm. "Minimizing the Multi-view Stereo Reprojection Error for Triangular Surface Meshes". In: *Proceedings of the 19th British Machine Vision Conference, Leeds, UK*, 2008.

[Essa 96] I. Essa, S. Basu, T. Darrell, and A. Pentland. "Modeling, Tracking and Interactive Animation of Faces and Heads Using Input from Video". In: *Proceedings of Computer Animation*, p. 68, 1996.

[Este 04] C. H. Esteban and F. Schmitt. "Silhouette and stereo fusion for 3D object modeling". *Comput. Vis. Image Underst.*, Vol. 96, No. 3, pp. 367–392, 2004.

[Faug 98] O. Faugeras and R. Keriven. "Variational principles, surface evolution, PDE's, level set methods and the stereo problem". *IEEE Trans. on Image Processing*, Vol. 7, No. 3, pp. 336–344, 1998.

[Fial 05] M. Fiala and C. Shu. "Fully automatic camera calibration using self-identifying calibration targets". Tech. Rep. NRC-48306 ERB-1130, National Research Council of Canada, 2005.

[Fial 08] M. Fiala and C. Shu. "Self-identifying patterns for plane-based camera calibration". *Machine Vision and Applications*, Vol. 19, No. 4, pp. 209–216, 2008.

[Fisc 81] M. Fischler and R. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". *Commun. ACM*, Vol. 24, No. 6, pp. 381–395, 1981.

[Floa 03] M. Floater. "Mean value coordinates". *Comput. Aided Geom. Des.*, Vol. 20, No. 1, pp. 19–27, 2003.

[Fors 02] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2002.

[Furu 07] Y. Furukawa and J. Ponce. "Accurate, Dense, and Robust Multi-View Stereopsis". In: *CVPR*, 2007.

[Furu 08] Y. Furukawa and J. Ponce. "Dense 3D Motion Capture from Synchronized Video Streams". In: *Proc. CVPR*, p. , 2008.

[Furu 09a] Y. Furukawa and J. Ponce. "Accurate Camera Calibration from Multi-View Stereo and Bundle Adjustment". *Int. J. Comput. Vision*, Vol. 84, No. 3, pp. 257–268, 2009.

[Furu 09b] Y. Furukawa and J. Ponce. “Accurate, Dense, and Robust Multi-View Stereopsis”. In: *PAMI*, 2009.

[Furu 09c] Y. Furukawa and J. Ponce. “Dense 3D Motion Capture for Human Faces”. In: *CVPR*, 2009.

[Fusi 00] A. Fusiello, E. Trucco, and A. Verri. “A compact algorithm for rectification of stereo pairs”. *Mach. Vision Appl.*, Vol. 12, No. 1, pp. 16–22, 2000.

[Gall 07] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. “Real-time Plane-sweeping Stereo with Multiple Sweeping Directions”. In: *CVPR*, 2007.

[Garg 07] P. Gargallo, E. Prados, and P. Sturm. “Minimizing the Reprojection Error in Surface Reconstruction from Images”. In: *ICCV*, 2007.

[Goes 06] M. Goesele, B. Curless, and S. M. Seitz. “Multi-View Stereo Revisited”. In: *CVPR*, 2006.

[Goes 07] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. “Multi-View Stereo for Community Photo Collections”. In: *ICCV*, 2007.

[Gold 07] R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, and E. Grinspun. “Efficient simulation of inextensible cloth”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 49, 2007.

[Gopi 00] M. Gopi, S. Krishnan, and C. Silva. “Surface Reconstruction based on Lower Dimensional Localized Delaunay Triangulation”. In: *Eurographics*, 2000.

[Guen 98] B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin. “Making faces”. In: *SIGGRAPH ’98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 55–66, 1998.

[Gusk 03] I. Guskov, S. Klibanov, and B. Bryant. “Trackable Surfaces”. In: *Proc. SCA*, pp. 251–257, 2003.

[Habb 07] M. Habbecke and L. Kobbelt. “A Surface-Growing Approach to Multi-View Stereo Reconstruction”. In: *CVPR*, 2007.

[Hart 94] R. I. Hartley. “Euclidean Reconstruction from Uncalibrated Views”. In: *Proceedings of the Second Joint European - US Workshop on Applications of Invariance in Computer Vision*, pp. 237–256, 1994.

[Hasl 06] N. Hasler, M. Asbach, B. Rosenhahn, J.-R. Ohm, and H.-P. Seidel. “Physically Based Tracking of Cloth”. In: *Proc. Workshop on Vision, Modeling, and Visualization*, pp. 49–56, 2006.

[Hern 07] C. Hernandez, G. Vogiatzis, G. J. Brostow, B. Stenger, and R. Cipolla. “Non-Rigid Photometric Stereo with Colored Lights”. In: *Proc. ICCV*, 2007.

[Hiep 09] V. H. Hiep, R. Keriven, P. Labatut, and J.-P. Pons. “Towards high-resolution large-scale multi-view stereo”. In: *CVPR*, pp. 1430–1437, 2009.

[Holr 08] M. Holroyd, J. Lawrence, G. Humphreys, and T. Zickler. “A Photometric Approach for Estimating Normals and Tangents”. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, Vol. 27, No. 5, p. 133, 2008.

[Horn 06] A. Hornung and L. Kobbelt. "Hierarchical Volumetric Multi-view Stereo Reconstruction of Manifold Surfaces based on Dual Graph Embedding". In: *CVPR*, 2006.

[Horn 89] B. K. P. Horn. "Obtaining shape from shading information". *Shape from shading*, pp. 123–171, 1989.

[Hull 08] M. B. Hullin, M. Fuchs, I. Ihrke, H.-P. Seidel, and H. P. A. Lensch. "Fluorescent Immersion Range Scanning". *ACM Trans. Graphics (Proc. SIGGRAPH)*, Vol. 27, No. 3, p. 87, 2008.

[Hung 06] Y. S. Hung and W. K. Tang. "Projective Reconstruction from Multiple Views with Minimization of 2D Reprojection Error". *Int. J. Comput. Vision*, Vol. 66, No. 3, pp. 305–317, 2006.

[Ihrk 04] I. Ihrke and M. Magnor. "Image-Based Tomographic Reconstruction of Flames". In: *Eurographics Symposium on Computer Animation*, pp. 367–375, 2004.

[ITU 07] ITU. "ITU-R BT.601-6, Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios". Standard Recommendation 601, International Telecommunication Union, 2007.

[ITU 90] ITU. "ITU-R BT.709, Basic Parameter Values for the HDTV Standard for the Studio and for International Programme Exchange". Standard Recommendation 709, International Telecommunication Union, 1990.

[Janc 09] M. Jancosek, A. Shekhovtsov, and T. Pajdla. "Scalable Multi-View Stereo". In: *3DIM (ICCV Workshop)*, 2009.

[Kadi 01] T. Kadir and M. Brady. "Saliency, Scale and Image Description". *Int. J. Comput. Vision*, Vol. 45, No. 2, pp. 83–105, 2001.

[Kazh 06] M. Kazhdan, M. Bolitho, , and H. Hoppe. "Poisson Surface Reconstruction". In: *Symposium on Geometry Processing*, 2006.

[Kole 09] K. Kolev, M. Klodt, T. Brox, and D. Cremers. "Continuous Global Optimization in Multiview 3D Reconstruction". *International Journal of Computer Vision*, Vol. 84, No. 1, pp. 80–96, 2009.

[Krae 04] V. Kraevoy and A. Sheffer. "Cross-parameterization and compatible remeshing of 3D models". *ACM Trans. Graphics (Proc. SIGGRAPH)*, pp. 861–869, 2004.

[Krae 05] V. Kraevoy and A. Sheffer. "Template-based mesh completion". In: *Proc. SGP*, p. 13, 2005.

[Laba 07] P. Labatut, J.-P. Pons, and R. Keriven. "Efficient Multi-View Reconstruction of Large-Scale Scenes using Interest Points, Delaunay Triangulation and Graph Cuts". In: *ICCV*, 2007.

[Ladi 08] A. Ladikos, S. Benhimane, and N. Navab. "Multi-View Reconstruction using Narrow-Band Graph-Cuts and Surface Normal Optimization". In: *British Machine Vision Conference (BMVC)*, 2008.

[Lamb 09] P. Lambert and P. Hébert. "Robust Multi-View Stereo without Matching". In: *3-D Digital Imaging and Modeling, 2009. 3DIM '09. Seventh International Conference on*, pp. 1614–1621, 2009.

[Lei 06] C. Lei and Y.-H. Yang. “Tri-focal tensor-based multiple video synchronization with subframe optimization”. *IEEE Transactions on Image Processing*, Vol. 15, No. 9, pp. 2473–2480, 2006.

[Levy 02] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. “Least squares conformal maps for automatic texture atlas generation”. *ACM Trans. Graph.*, Vol. 21, No. 3, pp. 362–371, 2002.

[Li 93] H. Li, P. Roivainen, and R. Forcheimer. “3-D Motion Estimation in Model-Based Facial Image Coding”. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 15, No. 6, pp. 545–555, 1993.

[Lian 05] C.-K. Liang, Y.-C. Peng, and H. H. Chen. “Rolling shutter distortion correction”. In: *Proc. of VCIP*, pp. 1315–1322, 2005.

[Lian 08] C.-K. Liang, L.-W. Chang, and H. H. Chen. “Analysis and compensation of rolling shutter effect”. *IEEE Transactions on Image Processing*, Vol. 17, No. 8, pp. 1323–1330, 2008.

[Lin 05] I.-C. Lin and M. Ouhyoung. “Mirror MoCap: Automatic and efficient capture of dense 3D facial motion parameters from video”. *The Visual Computer*, Vol. 21, No. 6, pp. 355–372, 2005.

[Linz 08] C. Linz, T. Stich, and M. Magnor. “High-speed motion analysis with multi-exposure images”. In: *Proc. of Vision, Modeling, and Visualization*, pp. 273–281, 10 2008.

[Liu 09a] Y. Liu, X. Cao, Q. Dai, and W. Xu. “Continuous depth estimation for multi-view stereo.”. In: *CVPR*, pp. 2121–2128, 2009.

[Liu 09b] Y. Liu, Q. Dai, and W. Xu. “A Point-Cloud-Based Multiview Stereo Algorithm for Free-Viewpoint Video”. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 99, 2009.

[Ma 07] W.-C. Ma, T. Hawkins, P. Peers, C.-F. Chabert, M. Weiss, and P. Debevec. “Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination”. In: *Eurographics Symposium on Rendering*, pp. 183–194, 2007.

[Ma 08] W.-C. Ma, A. Jones, J.-Y. Chiang, T. Hawkins, S. Frederiksen, P. Peers, M. Vukovic, M. Ouhyoung, and P. Debevec. “Facial Performance Synthesis using Deformation-Driven Polynomial Displacement Maps”. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, Vol. 27, No. 5, p. 121, 2008.

[Mein 05] M. Meingast, C. Geyer, and S. Sastry. “Geometric models of rolling-shutter cameras”. In: *Proc. of Int. Workshop on Omnidirectional Vision, Camera Networks, and Non-classical Cameras*, p. , 2005.

[Merr 07] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nister, and M. Pollefeys. “Real-Time Visibility-Based Fusion of Depth Maps”. In: *ICCV*, 2007.

[Moha 09] U. Mohammed, S. J. D. Prince, and J. Kautz. “Visio-lization: generating novel facial images”. *ACM Trans. Graph.*, Vol. 28, No. 3, p. 57, 2009.

[Mori 07] Y. Mori and T. Igarashi. “Plushie: an interactive design system for plush toys”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 45, 2007.

[mvie] mview. “http://vision.middlebury.edu/mview/”.

[Nick 07] S. P. Nicklin, R. D. Fisher, and R. H. Middleton. “Rolling shutter image compensation”. In: *RoboCup 2006*, pp. 402–409, 2007.

[Ogal 05] A. S. Ogale and Y. Aloimonos. “Shape and the Stereo Correspondence Problem”. *Int. J. Comp. Vis.*, Vol. 65, No. 3, pp. 147–162, 2005.

[Ogal 07] A. S. Ogale and Y. Aloimonos. “A Roadmap to the Integration of Early Visual Modules”. *Int. J. Comp. Vis.*, Vol. 72, No. 1, pp. 9–25, 2007.

[Ohta 03] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H. Seidel. “Multi-level partition of unity implicits”. In: *SIGGRAPH*, 2003.

[Open] “Open source computer vision library”.

[Ouya 05] C. Ouyang, G. Wang, Q. Zhang, W. Kang, and H. Ding. “Evaluating Harris method in camera calibration”. In: *IEEE Eng. in Medicine & Biology*, pp. 6383–6386, 2005.

[Pari 08] S. Paris, W. Chang, O. I. Kozhushnyan, W. Jarosz, W. Matusik, M. Zwicker, and F. Durand. “Hair photobooth: geometric and photometric acquisition of real hairstyles”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 30, 2008.

[Pere 03] P. Pérez, M. Gangnet, and A. Blake. “Poisson image editing”. *ACM Trans. Graph.*, Vol. 22, No. 3, pp. 313–318, 2003.

[Pigh 99] F. H. Pighin, R. Szeliski, and D. Salesin. “Resynthesizing Facial Animation through 3D Model-based Tracking”. In: *ICCV*, pp. 143–150, 1999.

[Pons 05] J.-P. Pons, R. Keriven, and O. Faugeras. “Modelling Dynamic Scenes by Registering Multi-View Image Sequences”. In: *CVPR*, 2005.

[Popa 09] T. Popa, Q. Zhou, D. Bradley, V. Kraevoy, H. Fu, A. Sheffer, and W. Heidrich. “Wrinkling Captured Garments Using Space-Time Data-Driven Deformation”. *Computer Graphics Forum (Proc. Eurographics)*, Vol. 28, No. 2, pp. 427–435, 2009.

[Popa 10] T. Popa, I. South-Dickinson, D. Bradley, A. Sheffer, and W. Heidrich. “Globally Consistent Space-Time Reconstruction”. In: *Eurographics Symposium on Geometry Processing*, p. , 2010.

[Prau 01] E. Praun, W. Sweldens, and P. Schröder. “Consistent Mesh Parameterizations”. In: *ACM Trans. Graphics (Proc. SIGGRAPH)*, pp. 179–184, 2001.

[Prit 03] D. Pritchard and W. Heidrich. “Cloth Motion Capture”. In: *Proc. Eurographics*, pp. 263–271, Sep. 2003.

[Quan 06] L. Quan, P. Tan, G. Zeng, L. Yuan, J. Wang, and S. B. Kang. “Image-based plant modeling”. *ACM Trans. Graph.*, Vol. 25, No. 3, pp. 599–604, 2006.

[Rav 08] A. Rav-Acha, P. Kohli, C. Rother, and A. Fitzgibbon. “Unwrap Mosaics: A new representation for video editing”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 2008.

[Rose 07] B. Rosenhahn, U. Kersting, K. Powell, R. Klette, G. Klette, and H.-P. Seidel. “A system for articulated tracking incorporating a clothing model”. *Machine Vision and Applications*, Vol. 18, No. 1, pp. 25–40, 2007.

[Rose 08] B. Rosenhahn, C. Schmaltz, T. Brox, J. Weickert, D. Cremers, and H.-P. Seidel. “Markerless Motion Capture of Man-Machine Interaction”. In: *Proc. CVPR*, p. , 2008.

[Scha 02] D. Scharstein and R. Szeliski. “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms”. *International Journal of Computer Vision*, Vol. 47, No. 1/2/3, pp. 7–42, 2002.

[Scho 04] V. Scholz and M. A. Magnor. “Cloth Motion from Optical Flow”. In: *Proc. Vision, Modeling and Visualization 2004*, pp. 117–124, November 2004.

[Scho 05] V. Scholz, T. Stich, M. Keckeisen, M. Wacker, and M. Magnor. “Garment Motion Capture Using Color-Coded Patterns”. In: *Proc. Eurographics*, pp. 439–448, 2005.

[Schr 04] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe. “Inter-surface mapping”. *ACM Trans. Graphics (Proc. SIGGRAPH)*, pp. 870–877, 2004.

[Seit 06] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. “A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms”. In: *CVPR*, 2006.

[Shar 08] A. Sharf, D. A. Alcantara, T. Lewiner, C. Greif, A. Sheffer, N. Amenta, and D. Cohen-Or. “Space-time surface reconstruction using incompressible flow”. *ACM Trans. Graph.*, Vol. 27, No. 5, pp. 1–10, 2008.

[Shef 04] A. Sheffer and V. Kraevoy. “Pyramid Coordinates for Morphing and Deformation”. In: *Proc. 3DPVT*, pp. 68–75, 2004.

[Shew 96] J. Shewchuk. “Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator”. In: M. C. Lin and D. Manocha, Eds., *Applied Computational Geometry: Towards Geometric Engineering*, pp. 203–222, Springer-Verlag, 1996.

[Shre 06] P. Shrestha, H. Weda, M. Barbieri, and D. Sekulovski. “Synchronization of multiple video recordings based on still camera flashes”. In: *Proc. of ACM Multimedia*, pp. 137–140, 2006.

[Sinh 04a] S. Sinha and M. Pollefeys. “Synchronization and calibration of camera networks from silhouettes”. In: *Proc. of ICPR*, pp. 1:116–119, 2004.

[Sinh 04b] S. N. Sinha, M. Pollefeys, and L. McMillan. “Camera Network Calibration from Dynamic Silhouettes”. *CVPR*, Vol. 1, pp. 195–202, 2004.

[Sinh 07] S. N. Sinha, P. Mordohai, and M. Pollefeys. “Multi-View Stereo via Graph Cuts on the Dual of an Adaptive Tetrahedral Mesh”. In: *ICCV*, 2007.

[Sorm 07] M. Sormann, C. Zach, J. Bauer, K. Karner, and H. Bischof. “Watertight Multi-View Reconstruction Based On Volumetric Graph-Cuts”. In: *SCIA*, 2007.

[Stei 98] G. P. Stein. “Tracking from multiple view points: Self-calibration of space and time”. In: *DARPA IU Workshop*, pp. 1037–1042, 1998.

[Stre 06] C. Strecha, R. Fransens, and L. V. Gool. “Combined Depth and Outlier Estimation in Multi-View Stereo”. In: *CVPR*, 2006.

[Sun 06] W. Sun and J. R. Cooperstock. "An empirical evaluation of factors influencing camera calibration accuracy using three publicly available techniques". *Machine Vision and Applications*, Vol. 17, No. 1, pp. 51–67, 2006.

[Szel 99] R. Szeliski. "A multi-view approach to motion and stereo". In: *CVPR*, 1999.

[Tan 07] P. Tan, G. Zeng, J. Wang, S. B. Kang, and L. Quan. "Image-based tree modeling". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 87, 2007.

[Theo 04] C. Theobalt, I. Albrecht, J. Haber, M. Magnor, and H.-P. Seidel. "Pitching a baseball – tracking high-speed motion with multi-exposure images". In: *Proc. of SIGGRAPH*, pp. 540–547, ACM SIGGRAPH, 2004.

[Toma 98] C. Tomasi and R. Manduchi. "Bilateral Filtering for Gray and Color Images". In: *ICCV*, pp. 839–846, 1998.

[Tran 06] S. Tran and L. Davis. "3D Surface Reconstruction Using Graph Cuts with Surface Constraints". In: *ECCV*, 2006.

[Trif 06] B. Trifonov, D. Bradley, and W. Heidrich. "Tomographic Reconstruction of Transparent Objects". In: *Eurographics Symposium on Rendering*, p. , 2006.

[Trig 99] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. "Bundle Adjustment - A Modern Synthesis". *Lecture Notes in Computer Science*, Vol. 1883, pp. 298–372, 1999.

[Tsai 86] R. Tsai. "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision". In: *CVPR*, pp. 364–374, 1986.

[Vlas 08] D. Vlasic, I. Baran, W. Matusik, and J. Popović. "Articulated Mesh Animation from Multi-view Silhouettes". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 97, 2008.

[Vlas 09] D. Vlasic, P. Peers, I. Baran, P. Debevec, J. Popović, S. Rusinkiewicz, and W. Matusik. "Dynamic Shape Capture using Multi-View Photometric Stereo". *ACM Transactions on Graphics*, Vol. 28, No. 5, p. 174, 2009.

[Vogi 07] G. Vogiatzis, C. H. Esteban, P. H. S. Torr, and R. Cipolla. "Multi-view Stereo via Volumetric Graph-cuts and Occlusion Robust Photo-Consistency". In: *PAMI*, 2007.

[Wand 09] M. Wand, B. Adams, M. Ovsjanikov, A. Berner, M. Bokeloh, P. Jenke, L. Guibas, H.-P. Seidel, and A. Schilling. "Efficient reconstruction of nonrigid shape and motion from real-time 3D scanner data". *ACM Trans. Graph.*, Vol. 28, No. 2, pp. 1–15, 2009.

[Wang 04] Y. Wang, X. Huang, C.-S. Lee, S. Zhang, Z. Li, D. Samaras, D. Metaxas, A. Elgammal, and P. Huang. "High Resolution Acquisition, Learning and Transfer of Dyanmic 3-D Facial Expressions". In: *Computer Graphics Forum*, pp. 677–686, 2004.

[Wang 05] H. Wang and R. Yang. "Towards space-time light field rendering". In: *Proc. of I3D*, pp. 125–132, 2005.

[Wang 09] H. Wang, M. Liao, Q. Zhang, R. Yang, and G. Turk. "Physically guided liquid surface modeling from videos". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 90, 2009.

[Wany 03] M. Wäny and G. P. Israel. "CMOS Image sensor with NMOS-only global shutter and enhanced responsivity". *IEEE Transactions on Electronic Devices*, Vol. 50, No. 1, pp. 57–62, 2003.

[Weng 92] J. Weng, P. Cohen, and M. Herniou. "Camera Calibration with Distortion Models and Accuracy Evaluation". *IEEE Transactions on PAMI*, Vol. 14, No. 10, pp. 965–980, 1992.

[Whit 07] R. White, K. Crane, and D. Forsyth. "Capturing and Animating Occluded Cloth". *ACM Trans. Graphics (Proc. SIGGRAPH)*, p. 34, 2007.

[Wilb 04] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz. "High-speed videography using a dense camera array". In: *Proc. of CVPR*, pp. 294–301, 2004.

[Will 90] L. Williams. "Performance-driven facial animation". *SIGGRAPH Comput. Graph.*, Vol. 24, No. 4, pp. 235–242, 1990.

[Zach 07] C. Zach, T. Pock, and H. Bischof. "A Globally Optimal Algorithm for Robust Tv-L^1 Range Image Integration". In: *ICCV*, 2007.

[Zach 08] C. Zach. "Fast and high quality fusion of depth maps". In: *3DPVT*, 2008.

[Zaha 06] A. Zaharescu, R. Horaud, R. Ronfard, and L. Lefort. "Multiple Camera Calibration using Robust Perspective Factorization". In: *3DPVT*, pp. 504–511, 2006.

[Zaha 07] A. Zaharescu, E. Boyer, and R. Horaud. "TransforMesh: A Topology-Adaptive Mesh-Based Approach to Surface Evolution". In: *ACCV*, 2007.

[Zaha 08] A. Zaharescu, C. Cagniart, S. Ilic, E. Boyer, and R. P. Horaud. "Camera Clustering for Multi-Resolution 3-D Surface Reconstruction". In: *ECCV 2008 Workshop on Multi Camera and Multi-modal Sensor Fusion Algorithms and Applications*, 2008.

[Zhan 00] Z. Zhang. "A flexible new technique for camera calibration". *IEEE Trans. on PAMI*, Vol. 22, No. 11, pp. 1330–1334, 2000.

[Zhan 04] L. Zhang, N. Snavely, B. Curless, and S. M. Seitz. "Spacetime faces: high resolution capture for modeling and animation". *ACM Trans. Graphics (Proc. SIGGRAPH)*, pp. 548–558, 2004.

[Zhan 99] Z. Zhang. "Flexible Camera Calibration By Viewing a Plane From Unknown Orientations". In: *ICCV*, pp. 666–673, 1999.