3D FACIAL PERFORMANCE CAPTURE FROM A SINGLE RGBD CAMERA

A Dissertation

by

YEN-LIN CHEN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Jinxiang Chai
Committee Members,	John Keyser
	Scott Schaefer
	Ricardo Gutierrez-Osuna
	Suhasini Subba Rao
Department Head,	Hank Walker

May 2013

Major Subject: Computer Science and Engineering

ABSTRACT

Realistic facial animation remains one of the most challenging problems in computer graphics, where facial performance capture of real people has been a key component. The current state-of-the-art technologies used to capture facial performances are far too expensive and cumbersome for general users, which limits the potential applications of performance capture. The primary contribution of this dissertation is to propose two systems that are suitable for common users to capture facial performance using a single low-cost device.

Our first system focuses on large-scale facial performance reconstruction from a single RGBD image. Our goal is to accurately reconstruct global transformation, as well as large-scale deformations from the images provided by a single shot of a Microsoft *Kinect* camera. With the combination of a robust facial feature detector and an image-based registration method, our system is automatic, robust and accurate to reconstruct facial movements. The result face meshes are topology consistent and with dense correspondences. Since people are natural experts of native human expressions and can distinguish subtle differences, *e.g.* dynamic facial wrinkles, we propose a second system combining our performance capture with a 3D scanning system to add person-specific high-resolution details in an efficient and effective way.

We demonstrate the power of our proposed systems by testing on both real and synthetic data, as well as a commercially available motion capture system. Results show that the proposed systems generate believable and comparable results. We believe the proposed systems should be useful and applicable for general as well as professional users.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deep gratitude to my advisor Prof. Jinxiang Chai for giving me the opportunity to perform research in this exciting and interesting field of computer vision and graphics. I greatly appreciate his expertise, invaluable guidance, and continuous support throughout my PhD study. His advice and detailed comments allowed me to achieve results that I would not have thought of. I would also like to thank my committee members, Prof. John Keyser, Prof. Scott Schaefer, Prof. Ricardo Gutierrez-Osuna, and Prof. Suhasini Subba Rao, for their precious advices and encouragements.

My special thanks to Dr. Xin Tong and Hsiang-Tao Wu at Microsoft Research Asia for continuous collaboration and invaluable discussion on the two projects of this thesis. Moreover, I would like to acknowledge my lab mate: Fuhao Shi for working on the facial performance capture project with me. I thank you all for the great partnership and support. Furthermore, I would like to thank all my friends and colleagues within the lab, Xiaolin Wei, Jianyuan Min, Hui Lou, Peizhao Zhang, and Jianjie Zhang, for the helps, ideas, and joy they brought into both my work and my life.

Finally, I would like to take the opportunity to thank my husband. He is always the one devoting himself for supporting me, sharing my happiness and sadness, and taking care of our family through the past few years. I would like to thank my parents as well for their continues love and support. I thank all who love me and helped me, for everything I have now would not have been real without having you in my life.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Facial performance is a crucial component in many fields, including computer science and engineering, medical science and psychology. It is also becoming a larger principle component in many applications, such as movies, video games, online virtual reporters, plastic surgery simulators, behavior monitoring, and other interactive human-computer interfaces. The recent most notable examples are found in the success of movies, which apply facial capture techniques to bring digital characters to life such as in the movies Beowulf and Avatar. Capturing facial performances of real people has been crucial and can be partially solved by using commercially available marker-based motion capture equipment such as a Vicon system [26]. However, this solution is too expensive for common use. Additionally, it is cumbersome, requiring the user to wear or draw markers on the face and track their movements with cameras. Most importantly, the marker-based motion capture system has low spatial resolution (usually 100 to 200 markers), which is not capable of capturing subtle local changes and details on face, especially the complex deformable surface of the human face. Capturing detailed 3D facial performances, however, is difficult because it requires capturing complex facial movements at different scales, including large-scale motion deformations and fine-scale geomantic details, such as pores and expression wrinkles.

Other than marker-based motion capture systems, researcher have also been exploring a number of approaches to capture 3D facial performances, including 3D scanning, structured light systems, and image-based stereo reconstruction systems. 3D face scanning systems, such as XYZ RGB [39], are capable of acquiring fine detail facial geometric details, but only for static poses. Structure light systems [40, 21, 2]

and multi-view stereo reconstruction systems [10] have made it possible to capture 3D dynamic faces with moderate fidelity, resolution, and consistency but their results still cannot match the spatial resolution of static face scans or the acquisition speed of marker-based motion capture systems. Most importantly, they use more expensive capturing equipments, such as multiple professional digital cameras with or without projectors, and the equipments need to be placed in a specific arrangement, which makes the systems not portable and not suitable for common users.

In this dissertation, we propose systems to capture and reconstruct facial performance by using a single cheap device as capturing equipment, and we demonstrate that our results are comparable to that of commercial capturing systems at similar levels of details. Our goal is to design and build automatic facial performance acquisition systems that are suitable for common users. We focus on image-based facial capture because it is intuitive. People can easily take facial pictures or videos by using a phone, any commercial camera, or even a cheap gaming device, for example, a Microsoft *Kinect* camera, and use it to reconstruct facial performances. We choose *Kinect* cameras for facial performance acquisition because they are low-cost, easy to use, require no markers, and moreover, provide an additional channel depth map, which is more stable than RGB images under different light conditions. A *Kinect* camera simultaneously captures depth maps with a resolution of 320×240 and color images with a resolution of 640×480 at 30 frames per second based on infrared projection (Figure 1.1).

Choosing to use a single *Kinect* camera as the input device and use image-based motion capture technique as our main framework bring about two main challenges. First, low-cost devices often provide low quality data, such as noisy low-resolution images, which would directly affect both the accuracy of the results and the robustness of the system. Figure 1.2 shows some sample frames of the observed images and

Figure 1.1: The Kinect camera simultaneously captures a 320×240 pixels depth map and a 640×480 pixels color image at 30 Hz.

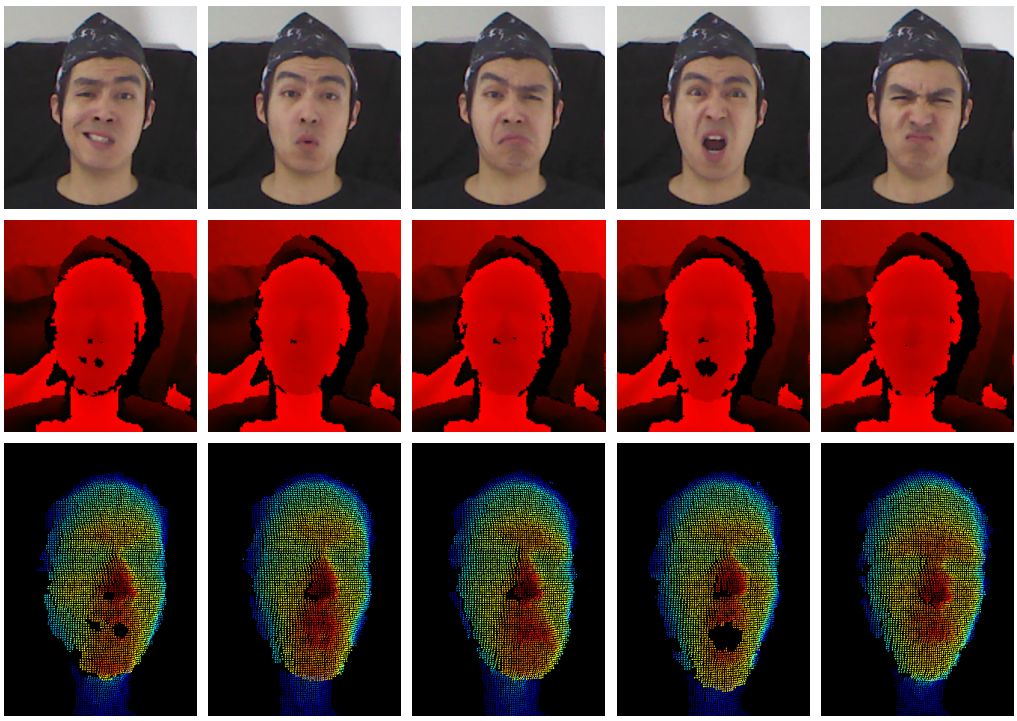


Figure 1.2: Sample frames of observed images and point clouds of a Kinect camera: row (1) shows the video images, row (2) shows the depth images, and row (3) shows the depth point clouds, where warmer color is used for visualizing points that are closer to the depth camera.

point clouds. Second, people are naturally experts of native human expressions and can distinguish subtle differences from global movements, large-scale deformations and even fine-scale facial geometry details. How to apply low-quality data to reconstruct accurate and quality results becomes the main issue, and addressing it would make a major contribution to this field of study.

We propose two systems that provide different needs. The first system focuses on large-scale facial performance capture from a single RGBD image. Our goal is to accurately reconstruct global transformation, as well as large-scale deformations from the images provided by a single shot from a Microsoft *Kinect* camera. We set several requirements for this system, including accuracy, generality, robustness, and automation. The system needs to best utilize the input images, down to sub-pixel accuracy, to generate a result that closely matched the observed data. The system needs to be general for common users, meaning no extra prior knowledge would be needed to reconstruct facial performance, such as person-specific 3D facial models or blendshapes. The system should be robust to process any single frame independency without knowing temporal coherence and should be fully automatic without any manual assistance through out the process.

Due to the aforementioned challenges, low-cost devices typically do not provide enough information about facial details, such as pores and dynamic wrinkles, which are vital components of a quality facial performance. Hence, our second system focuses on reconstructing such details. Considering the feasibility of common users, we extend the first system by incorporating additional prior information to help in reconstruction of high-fidelity facial details. The person-specific priors need be easy to identify, require minimal efforts to retrieve, and be able to be reused. We herein incorporate a 3D scanning system into the performance capture system, and provide an automatic facial analysis technique to determine a minimal set of face scans

Figure 1.3: The capturing equipments and environment settings for our proposed systems. (left) a Kinect camera is used for capturing large-scale movements; (right) a Minolta VIVID 910 laser scanner is used for scanning static facial geometry of an actor to help in reconstruction of high-fidelity facial details.

required for accurate facial performance reconstruction. In other words, we maximize the use of any single face scan to cover similar expressions under different scales and rigid transformations. The face scans could be reused afterwards, and therefore, the user could still use a single *Kinect* camera to reconstruct high-detailed facial performances. More face scans used to reconstruct the performance from a sequence generate closer quality to per-frame face scans. Figure 1.3 shows the capturing equipments and environment settings for both systems.

The main contribution of our performance capture system is a novel 3D facial modeling process that automatically reconstructs 3D facial expression from a single RGBD image without using any 3D facial prior information. Thus, it is not restricted to capture predefined facial expressions. We focus on per-frame reconstruction because it ensures the process is fully automatic and does not suffer from drifting error. At the core of our system lies a 3D facial deformation registration process which iteratively deforms a template face model to best match a single depth image. To give good initialization, we incorporate a facial features detection process into the regis-

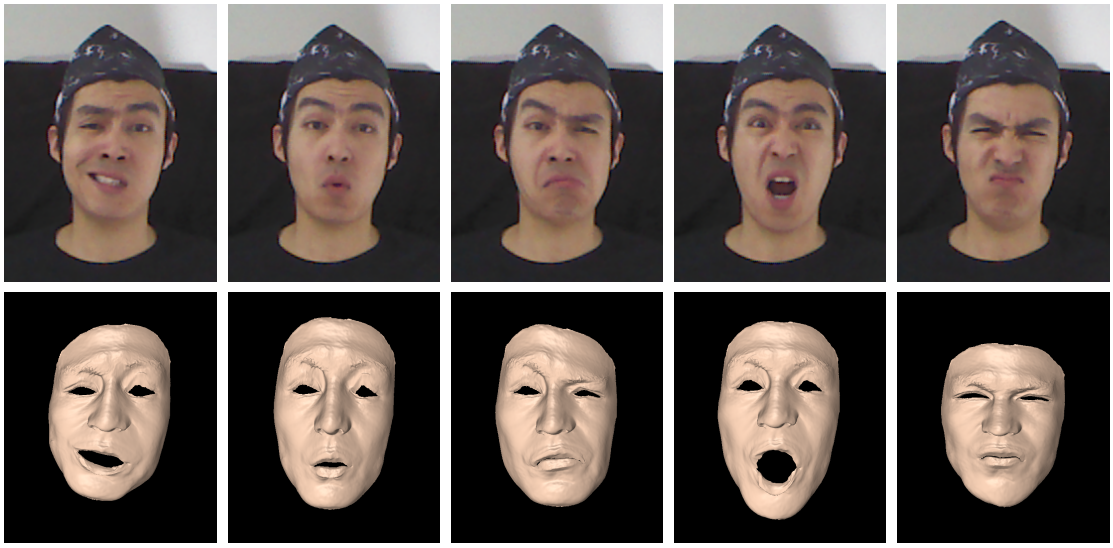


Figure 1.4: Reconstructed faces from our performance capture system using a single *Kinect* RGBD image: (top) observed images; (bottom) the reconstructed facial performances.

tration process that significantly improves the accuracy and robustness of the facial capture system. The results obtained from our system are quantitatively comparable in quality to those obtained from a commercial motion capture system with a full set of markers. Figure 1.4 shows some reconstruction results from our system.

The second system is an extension of the first system that incorporates a high-resolution 3D face scan system into the system to make it possible for reconstructing high-fidelity facial performances with realistic dynamic wrinkles and fine-scale facial details. We start the process by recording facial performances of an actor using a *Kinect* camera. We track the whole sequence of images and depth data by deforming a 3D template mesh model to match observed data at each frame. We then perform automatic facial analysis of the facial tracking data and thereby obtain a minimal set of face scans required for accurate facial reconstruction. We introduce a novel registration process that utilizes image and depth data across the entire sequence to

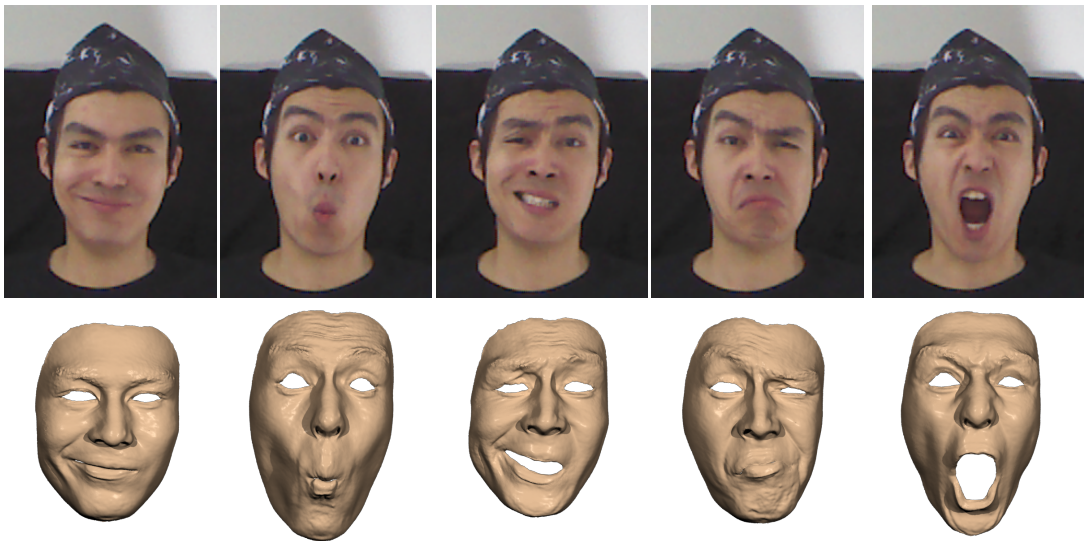


Figure 1.5: Sample frames of the reconstructed high-fidelity facial performances from our system: (top) observed images; (bottom) the reconstructed facial performances.

automatically build dense and consistent surface correspondences between all the face scans. Lastly, we combine depth and image data with the minimal set of registered face scans to reconstruct high-fidelity facial performances in a keyframe interpolation framework. Figure 1.5 shows some sample frames from our reconstruction results.

We demonstrate the effectiveness of our systems by capturing a wide range of facial expressions from multiple subjects. Moreover, we evaluate the performance of our systems by testing on synthetic data and comparing against alternative methods and a commercially available motion capture system. Results show that our proposed systems generate believable and comparable results. We believe both systems could be useful and feasible for both general and professional users.

1.1 Contributions

Our marker-less facial performance capture process, by using a single low-cost device, is made possible by the following two systems:

- An automatic, general, robust, and accurate 3D facial performance capture system that combines the power of facial feature detection and facial deformation registration.

- A high-quality facial performance capture system that combines the power of large-scale facial motion tracking and 3D high-resolution priors acquisition with minimum efforts using a 3D scanning system.

Our low-cost facial capture process is made possible and feasible for common users by a number of technical contributions:

- A low-cost, ease of use, and non-intrusive system that uses only a single Kinect camera and does not need to put either markers or makeups on the face.

- A general system that requires no 3D facial database (priors) or blendshapes, but just a template.

- A robust framework that reconstruct performance with single-frame estimation.

- An accurate model-based depth flow algorithm that iteratively register a template face model with observed facial data in a single depth image via linear system solvers, which achieves sub-pixel accuracy.

- A combination of a facial feature detector and an image-based registration framework that guarantees consistent correspondences of all result meshes on both important features and dense surface. This makes the reconstructed models easy to use for multiple applications, such as video editing.

Our high-fidelity facial performance acquisition is made possible by a number of technical contributions:

- An efficient facial reconstruction method that utilize observed image and depth data as well as a minimal set of registered face scans to accurately reconstruct facial performances in a keyframe interpolation framework. This framework also enables the possibility of reconstructing high detailed facial performance by reusing the face scans if performing similar performances.

- A novel facial tracking framework that automatically translates, rotates and deforms a template mesh model to match image and depth data obtained from a single *Kinect* camera. This also guarantees the result meshes are topology consistent and with dense correspondences.

- An automatic facial analysis technique that determines a minimal set of face scans required for accurate facial performance reconstruction. This not only improves the accuracy of 3D facial reconstruction but also significantly reduces the time and effort required for 3D face scanning.

- A new registration process that automatically builds dense, consistent surface correspondences across all the face scans using image and depth data obtained from a *Kinect* camera. This is nontrivial because face scans often contain high resolution facial details such as pores and wrinkles, and a small misalignment between any two scans will result in unpleasant visual artifacts in the captured facial performance.

In the next chapter, we describe how to capture large-scale 3D facial performance from a single RGBD image. We then describe our approach for reconstructing fine-scale facial geometric details in Chapter 3.

2. AUTOMATIC CAPTURE OF 3D FACIAL PERFORMANCES USING A SINGLE RGBD IMAGE

The ability to accurately reconstruct 3D facial performances would allow the intuitive control of characters in computer games, the control of avatars for virtual reality or electronically mediated communication, and the rapid prototyping of facial animations. This problem has been partially solved by commercially available marker-based motion capture equipment (*e.g.* [26]), but this solution is far too expensive for common use. It is also cumbersome, requiring the user to wear more than 90 carefully positioned retro-reflective markers on face. In this system, we present a different approach to solving this problem: reconstructing the user's 3D facial performances using a single RGBD camera (Figure 2.1). The results obtained from our system are quantitatively comparable in quality to those obtained from a commercial motion capture system with a full set of markers. The cost is much lower because only a single *Kinect* camera is required. The system is also portable, easy to set up, and non-intrusive because it requires no markers or no controlled illumination.

The main contribution of this system is a novel 3D facial modeling process that automatically reconstructs 3D facial expression from a single RGBD image. We focus on per-frame reconstruction because it ensures the process is fully automatic and does not suffer from drifting error. Unlike similar facial capture systems using a *Kinect* [36], our process does not require any 3D facial priors and therefore is not restricted to capture predefined facial expressions. At the core of our system lies a 3D facial deformation registration process which iteratively deforms a template face model to best match a single depth image. We build the registration algorithm on Lucas-Kanade registration framework [4] by extending model-based optical flow to

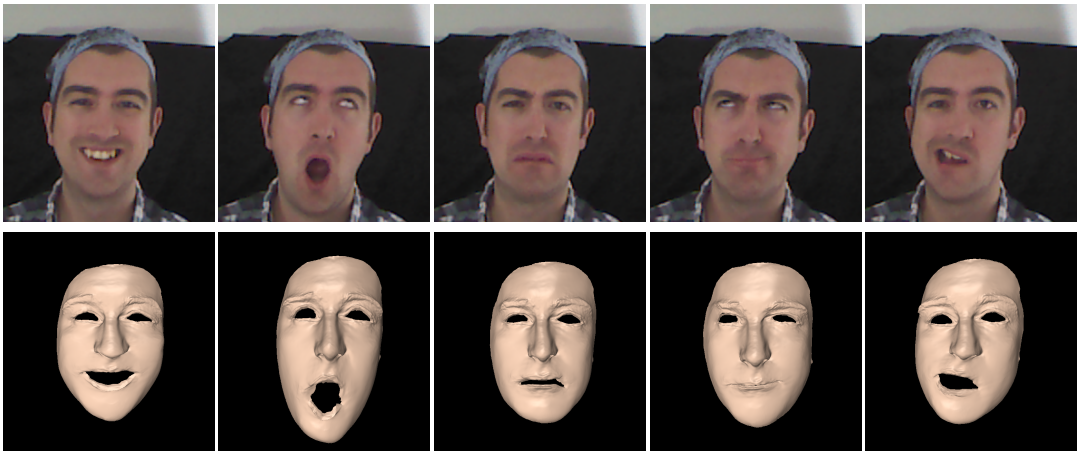


Figure 2.1: Our system captures a variety of facial performances with a single *Kinect* Camera: (top) image data; (bottom) the reconstructed facial performances.

depth data. This allows us to use linear system solvers to incrementally deform a template face model using a single depth image.

Our registration process, like any other iterative registration process, requires a good initialization. The system often produces poor reconstruction results when facial performances are far from the template face model. It also does not take into account perceptually significant facial features such as nose tip and mouth corners, thereby resulting in misalignments in those perceptually important facial regions. We address the challenges by complementing our registration process with facial feature detection process. Incorporating facial features into our registration process significantly improves the accuracy and robustness of the facial capture system.

Our final marker-less facial capture process is made possible by a number of technical contributions:

- A novel facial feature detector that accurately and reliably locates important facial features in a single RGBD image. The detection process is robust to variation in human subjects, facial expressions, head poses, and illuminations.

- A model-based depth flow algorithm that iteratively register a template face model with observed facial data in a single depth image via linear system solvers.

- An automatic, accurate 3D facial performance capture system that combines the power of facial feature detection and facial deformation registration. The system does not suffer from drifting error because it builds on single-frame modeling process. It is also flexible because it does not rely on any 3D facial priors or predefined blendshape models.

We demonstrate the power of our facial performance capture system by capturing a wide range of facial expressions from multiple subjects. We evaluate the performance of our system by comparing against alternative methods such as marker-based motion capture and nonrigid registration using Iterative Closest Points (ICP). We have also explored application of our captured facial performances to edit photo-realistic facial video data.

2.1 Background

Our system reconstructs 3D facial expression using a single RGBD image. Therefore, we will focus our discussion on methods and systems developed for acquiring 3D facial performances.

One of most successful approaches for 3D facial performance capture is to use marker-based motion capture systems [38, 17], which robustly and accurately track a sparse set of markers attached on face. Recent effort in this area has been focused on complementing marker-based systems with other capturing types of devices such as video cameras and/or 3D scanners to improve the resolution and details of captured facial geometry. Bickel and his colleges [7] augmented the marker-based motion capture system with face paints and two synchronized video cameras for tracking

medium-scale expression wrinkles. More recently, Huang and his colleagues [18] proposed a system that combines the power of marker-based motion capture and 3D scanning for acquiring fine-scale geometric details in facial performances. Marker-based motion capture, however, is not practical for random users targeted by this system as they are expensive, cumbersome, and intrusive for 3D facial performance capture.

Marker-less motion capture provides an appealing alternative to facial performance capture because it is non-intrusive and does not impede the subject's ability to perform facial expressions. One solution to marker-less facial capture is the use of depth and/or color data obtained from structured light systems [40, 23, 21, 37]. For example, Zhang and his colleagues [40] captured 3D facial geometry and texture over time and built the correspondences across all the facial geometries by deforming a generic face template to fit the acquired depth data using optical flow computed from image sequences. Ma et al. [23] achieved high-resolution facial reconstructions by interleaving structured light with spherical gradient photometric stereo using the USC Light Stage. Recently, Li and his colleagues [21] captured dynamic depth maps with their realtime structured light system and fit a smooth template to the captured depth maps using embedded deformation techniques [32].

Reconstructing high-quality face models directly from multiview images offers another possibility for marker-less motion capture [28, 9, 10, 5, 6]. Borshukov and his colleagues [9] developed the *Universal Capture* system to recreate actors for *The Matrix Reloaded*. Their system deformed a scanned 3D face template by using optical flow fields computed from multiple image sequences. Bradley and his colleagues [10] used multi-view stereo reconstruction techniques to obtain initial facial geometry, which is then used to capture 3D facial movement by tracking the geometry and texture over time. Beeler et al. [5] presented an impressive multi-view stereo re-

construction system for capturing the 3D geometry of a face in a single shot and late extended it to acquiring dynamic facial expressions using multiple synchronized cameras [6]. More recently, Valgaerts et al. [34] combined image-based tracking with shading-based geometry refinement to reconstruct facial performances from stereo image sequences.

The minimal requirement of a single camera for facial performance capture is particularly appealing, as it offers the lowest cost and a simplified setup. However, previous single camera systems for facial capture [15, 14, 29] are often vulnerable to ambiguity caused by a lack of distinctive features on face and uncontrolled lighting environments. One way to address the issue is to use 3D prior models to reduce the ambiguity of image-based facial deformations(*e.g.*, [8, 35]). Another possibility is to extract a small number of 2D facial features from monocular image sequences and use them to interpolate prerecorded facial data [11].

Among all the systems, our work is most closely related to Weise et al. [36], which uses RGBD images from a *Kinect* and a template, along with a set of predefined blend shape models, to track facial performances over time. Our system shares a similar perspective as theirs because both are targeting low-cost and portable facial capture accessible to random users. Our goal, however, is different from theirs in that we focus on authentic reconstruction of 3D facial performances while they are mainly focused on performance-based avatar animation and control. Our method for facial capture is also distinct from theirs. Their approach utilizes a set of predefined blend shape models and closest points measurement to sequentially track facial performances in a Maximum A Posteriori(MAP) framework. Our approach does not require any 3D facial model priors. Our facial capture builds on model-based depth flow that iteratively registers a template face model with observed facial data in a single depth image. In addition, our system automatically locates important facial features in

a single RGBD image and uses them to initialize and guide our facial registration process.

2.2 Overview

Our system acquires high-quality 3D facial models using a single kinect camera. We choose a *Kinect* for facial performance capture because it is low cost, portable, and non-intrusive. A *Kinect* can simultaneously capture depth maps with a resolution of 320×240 and color images with a resolution of 640×480 at 30 frames per second based on infrared projection. Our facial modeling process leverages facial detection and nonrigid facial registration for automatic 3D facial modeling. In the following, we highlight the issues that are critical for the success of this endeavor and summarize our approach for addressing them.

We start the process by automatically identifying important facial features such as nose tip, eye and mouth corners in a single RGBD image (see Figure 2.2(d)). The problem is challenging because the algorithm needs to be robust to variation in human subjects, head movements, facial expressions, and illumination conditions. We formulate feature detection as a per-pixel classification problem and apply randomized trees to associate each pixel with probability scores of being a particular feature. The detected features are often noisy and frequently corrupted by outliers due to classification errors. To handle this challenge, we robustly search closest examples in a training set of labeled images, where all the key facial features are labeled, and remove misclassified features that are inconsistent with the closest example. In the final step, we refine feature locations by utilizing active appearance models (AAM) and 2D facial priors embedded in K closest examples of detection features.

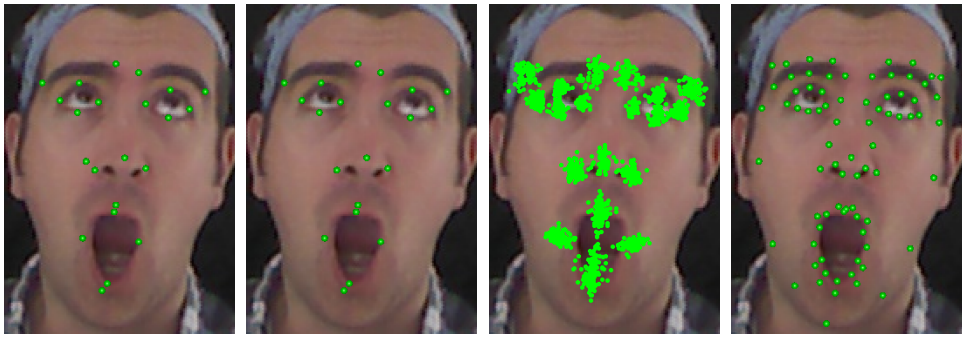
The next challenge is how to accurately reconstruct 3D facial expression from a

single RGBD image. We model nonrigid facial deformation using embedded deformation [33] and this allows us to constrain the solution to be in a reduced subspace. We introduce a model-based depth flow algorithm to incrementally deform a face template to match observed depth data via linear system solvers. In addition, we incorporate *facial feature* term and *boundary constrain* term into the registration framework to improve the robustness and accuracy of our system.

We demonstrate the power and effectiveness of our 3D facial modeling process in two applications, including marker-less facial performance capture and photo-realistic video editing. In particular, our video editing process allows the user to edit underlying geometry and/or texture data of facial subjects in any frame across the entire video sequence and propagating the effects of editing to the whole image sequences. We describe these components in more detail in the next sections.

2.3 Robust Facial Feature Detection

This section introduces a robust feature detection algorithm that accurately locates a set of predefined facial features (*e.g.* nose tip and the mouth corners) in a single RGBD image. The whole detection process consists of three main steps. The system first detects feature locations by using local information of pixels. The features from local detection process are often noisy and frequently corrupted by outliers due to detection errors (see Figure 2.2(a)). We automatically remove misclassified pixels by robustly searching closest examples in a training image data set (Figure 2.2(b) and (c)). In the final step, we refine feature locations and extract locations of secondary facial features by utilizing Active Appearance Models (AAMs) [24] and 2D facial priors embedded in closest examples (Figure 2.2(d)). We discuss each step in detail in the rest of this section.



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

Figure 2.2: Robust detection of facial features: (a) candidate features after local detection and multi-mode extraction; (b) detected features after outlier removal; (c) closest examples of detected features found by robust KNN; (d) final detection output.

2.3.1 Local Feature Detection

Our local feature detection process focuses on utilizing local information of a pixel (*i.e.* an input patch centered at a pixel) to detect locations of facial features. We formulate the local feature detection process as a per-pixel classification problem. During training, we construct a set of $N = 21$ classes of keypoints. Each class corresponds a prominent facial feature such as nose tip and left corner of the mouth. Figure 2.3 shows sample RGBD images of training data set, and Figure 2.2(b) shows the salient facial features considered by local detection process. At runtime, given an input patch centered at a RGBD pixel $\mathbf{x}$, we want to decide the likelihood that a desired feature $c \in \{1, ..., N\}$ is located at point $\mathbf{x}$ in the image.

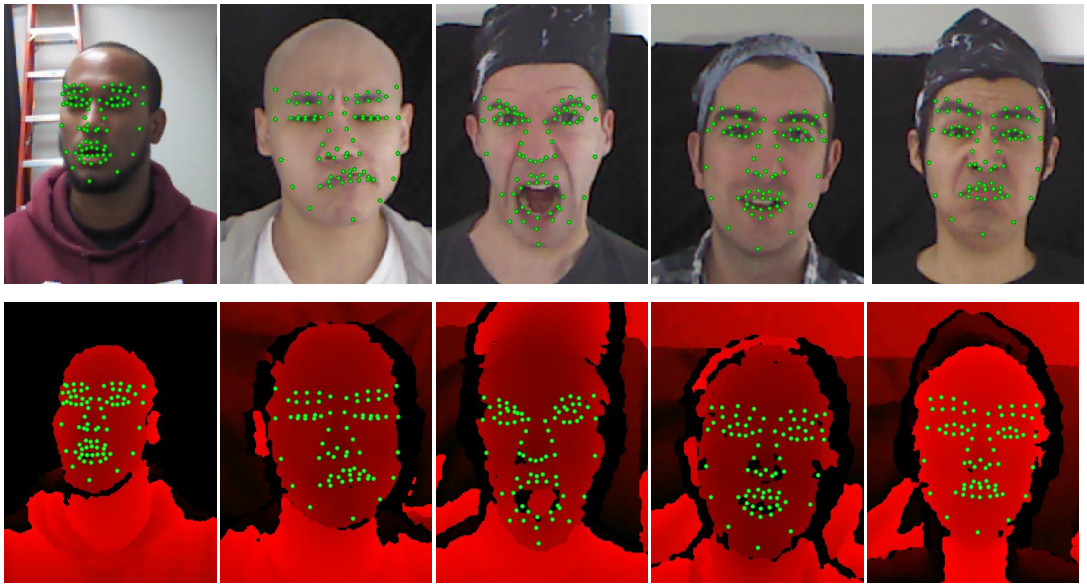


Figure 2.3: Sample training images with manually annotated facial features: (top) RGB images; (bottom) depth images.

We use randomized decision trees [3, 20] to train a classifier for automatic labeling of pixels. Randomized trees are an ensemble of L decision trees $T_1, ..., T_L$. Each node in the tree contains a simple test that splits the space of data to be classified, in our case the space of image patches. Each leaf contains an estimate based on training data of the posterior distribution over the classes. A new patch is classified by dropping it down the tree and performing an elementary test at each node that sends it to one side or the other. When it reaches a leaf, it is assigned probabilities of belonging to a class depending on the distribution stored in the leaf. Once the trees $T_1, ..., T_L$ are built, their responses are combined during classification to achieve a better recognition rate than a single tree could.

Specifically, for an input patch centered at pixel $\mathbf{x}$, each tree $T_1, ..., T_L$ outputs posterior probabilities $Pr_{\lambda(l,\mathbf{x})}(c|\mathbf{x})$, where c is a label in $C = \{1, ..., N\}$ and $\lambda(l, \mathbf{x})$ is the leaf of tree T_l reached by the patch $\mathbf{x}$. The probability of the patch $\mathbf{x}$ associated

with each feature c is the average of the class distributions over the leaf nodes reached for all L trees:

$$Pr(c|\mathbf{x}) = \frac{1}{L} \sum_{l=1,\ldots,L} Pr_{\lambda(l,\mathbf{x})}(c|\mathbf{x}) \tag{2.1}$$

The probability score $Pr(c|\mathbf{x})$ indicates the likelihood that the desired feature c is located at point $\mathbf{x}$ in the image.

In our application, the tests performed at the nodes are simple binary tests based on simple functions of raw pixels taken in the neighborhood of the classification pixel. Our feature function calculates the difference of depth or intensity values of a pair of pixels taken in the neighborhood of the classification pixel. We normalize the offset of each depth pixel by its depth value to ensure the features are depth invariant. If the value of a splitting function is larger than a threshold, go to the left node and otherwise go to the right node. The choice of which pair of pixels and which type of pixels (depth or color) to be selected as well as the optimal threshold for splitting the node is automatically determined by maximizing the information gain for particular features during the training phase.

Our next task is to infer feature locations from their probability maps in the whole detection region (or in some smaller region around the face as inferred from an earlier face detection step). One way to achieve this is to extract the location of the highest detector score. In practice, no detector is perfect, so the correct location will not always be at the location with the highest detector score. Instead of choosing feature locations corresponding to the highest detection score, we extract a set of candidate locations by detecting peaks of all the important modes in probability maps and then rely on robust KNN search to remove misclassified features (see next subsection). This idea is motivated by an observation on local feature detection results. We have observed that actual locations of features are always correlated

to the peaks of important modes. We adopt a simple yet very effective method for detecting peaks of important modes. We first process the probability map by marking all the pixels with a cutoff threshold. In particular, a pixel is marked if its score is lower than a threshold. We search the highest detection score among all the unmarked pixels to extract the peak of the first mode. Once the peak of the first mode is extracted, we mark the corresponding pixel as well as its connected regions via flood fill algorithm. We repeat the same process until all the pixels in the detection window are marked.

2.3.2 Robust KNN Search

In the following, we describe how to use a training image data set to build global detectors to better handle the cases when the local detectors are likely to go astray.

Due to classification errors, feature candidates inevitably contain "outlier" features. Our idea is to robustly search closest examples in a training set of labeled images, where all the key facial features are labeled, and remove misclassified features that are inconsistent with the closest example. KNN search, however, requires computing similarity transformations for aligning the detection image with every training image. We formulate the problem as a robust fitting problem and apply random sampling techniques to automatically detect and remove misclassified features. We choose random sampling techniques because it allows for a robust estimate of the similarity transformation even in the presence of a high percentage of misclassified features.

We randomly sample a pair of feature points obtained from the local detection process and compute similarity transformations T to align the detection image with every database image $m = 1, ..., M$. We measure the similarity between detection image and a database image by counting the number of "inlier" features. A feature

c is considered as an "inlier" when the distance between a feature in detection image $T(\mathbf{x}_c)$ and a corresponding feature $\mathbf{z}_c$ in a database image is smaller than a threshold. Some features might have more than one candidate location caused by multiple modes in probability maps. When this occurs, we choose the feature location corresponding to the smallest feature distance $T(\mathbf{x}_c) - \mathbf{e}_c$.

We select the best sample $\tilde{r}$ by maximize the similarity between detection image and its closest example:

$$\{\tilde{r}, \tilde{m}\} = \arg\max_{r,m} S(m, r), \tag{2.2}$$

where r and m are sample index and database image index, respectively. $S(r, m)$ is a similarity score between detection image and m-th database image for r-th sample. Since our detection process only contains a small number of outliers, we do not need a large number of samples. We find *15* samples are often sufficient to generate satisfactory results for all the experiments in the system.

Meanwhile, we can find K closest examples of extracted "inlier" features based on the following metric:

$$dist = \sum_{c \in inlier} w_c \left\| T(\mathbf{x}_c) - \mathbf{z}_c \right\|^2, \tag{2.3}$$

where the function T is the 2D similarity transformation that aligns the detected features $\mathbf{x}_c$ with the corresponding database features $\mathbf{z}_c$. The weight $w_c = exp(-\frac{\|1 - Pr(\mathbf{x}_c)\|^2}{2\sigma^2})$ is the probability score of an "inlier" feature. A high probability score $Pr(\mathbf{x}_c)$ results in a low weight w_c. In our experiment, σ is set to 0.3.

2.3.3 Feature Location Refinement

This step is necessary for high-quality 3D facial modeling because of two reasons. First, even with outlier removal, results obtained from feature detection are still noisy.

For important facial features such as nose tip and mouth corners, several pixels of error might result in significant visual artifacts in output animation. Second, high-quality 3D facial modeling often requires accurate location of entire facial regions such as the lip and eyebrows. The set of facial features obtained from detection process is often not sufficient to accurately model such important regions. We refine the feature detection results by utilizing Active Appearance Models (AAM) [24] and K closest example of detected features. Figure 2.2(b) and (d) shows the improvement of feature locations as well as detected secondary facial features via the refinement step.

We formulate the refinement process in an optimization framework. The whole cost function consists of three terms:

$$E = w_1 E_{AAM} + w_2 E_{detection} + w_3 E_{prior}, \tag{2.4}$$

where the first term E_{AAM} is *Active Appearance Models* (AAM) term, which measures the inconsistency between detection image and the AAM model instance (for details, refer to [24]). The second term is the *detection* term which penalizes the deviation of new feature points from detected feature points from Section 2.3.2. The third term is the *prior* term which ensures the new feature points are consistent with 2D facial priors embedded in K closest examples. In this work, we fit a Gaussian prior based on K closest examples and obtain this term by applying the negative log to the Gaussian distribution. The local priors avoid the problem of finding an appropriate structure for global priors, which would necessarily be high-dimensional and nonlinear.

We minimize the cost function by simultaneously optimizing the shape and appearance parameters of the AAM model instance, as well as the similarity trans-

formation for aligning the the detection image with the AAM model instance. We initialize the shape parameter using feature locations of the closest example. The initialization of the similarity transformation is obtained from detection process via KNN search. The appearance parameter is set to the base appearance image of AAM models. We optimize the function in Lucas-Kanade registration framework via iterative linear system solvers [24]. We experimentally set the weights of w_1, w_2 and w_3 to 1, 2, and 0.001 respectively. The optimization typically converges in 10 iterations because of very good initializations. During the iterations, we gradually decrease the weight for the second term (w_2) from 2 to 0.0001 in order to ensure that the final feature locations can achieve a better accuracy via AAM fitting.

2.4 Image-based Nonrigid Facial Registration

This section focuses on automatic facial modeling using a single RGBD image. This is achieved by deforming a template mesh model, $\mathbf{s}_0$, to match depth/color image data as well as facial features from Section 2.3. We obtain the template mesh model $\mathbf{s}_0$ by scanning facial geometry of the subject under a neutral expression (Figure 2.4). We formulate the problem in a Lucas-Kanade image registration framework and incrementally estimates both rigid transformations (ρ) and nonrigid deformation ($\mathbf{g}$) via linear system solvers.

We represent rigid transformations of face using a 6-by-1 vector ρ. We model nonrigid deformation $\mathbf{g}$ using embedded deformation representation developed by Sumner et al. [32]. Embedded deformation builds a space deformation represented by a collection of affine transformations organized in a graph structure. One affine transformation is associated with each node and induces a deformation on the nearby space. The influence of nearby nodes is blended by the embedded deformation algorithm in order to deform the vertices or the graph nodes themselves. We choose

Figure 2.4: Initial template model obtained from a Minolta VIVID 910 laser scanner under a neutral expression. (left) scan mesh; (middle) textured scan mesh; (right) embedded deformation representation using 374 graph nodes represented by cubes.

embedded deformation because it allows us to model the deformation in a reduced subspace, thereby significantly reducing the ambiguity for facial modeling.

In embedded deformation, the affine transformation for individual node is defined by a 3-by-3 matrix $\mathbf{A}_i$ and a 3-by-1 translation vector $\mathbf{t}_i$. In this way, the collection of all per-node affine transformations, denoted as $\mathbf{g} = \{\mathbf{A}_i, \mathbf{t}_i\}_{i=1,\ldots,M}$, expresses a non-rigid deformation of the template mesh model in a reduced deformation space. In our experiment, graph nodes are chosen by uniformly sampling vertices of the template mesh model in the frontal facial region. We have found 300 graph nodes are often sufficient to model facial details captured by a *Kinect*. Let $\mathbf{q} = [\rho, \mathbf{g}]$ denote the state of our modeling process. The 3D representation of our facial model can be defined as $\mathbf{s} = \mathbf{s}_0 \oplus \mathbf{q}$.

Let $\bar{\mathbf{p}} = [\bar{x}, \bar{y}, \bar{z}]$ and $\mathbf{p} = [x, y, z]$ be the local and global 3D coordinates of a point located on the 3D mesh, respectively. Let $\mathbf{x} = [u, v]$ be the 2D coordinates of the corresponding pixel in depth image. We model the mapping from local coordinates to global coordinates using the forward kinematic function for mesh deformation: $\mathbf{p} =$

$\mathbf{h}(\mathbf{q}; \mathbf{p}_0, \mathbf{s}_0)$, which computes the global position ($\mathbf{p}$) of a surface point from the model state ($\mathbf{q}$) given the local coordinates ($\mathbf{p}_0$) of a surface point on the template mesh ($\mathbf{s}_0$). The relationship between the global coordinates of a point and its corresponding pixel on image plane is defined by a projection transformation $\mathbf{x} = \mathbf{f}(\mathbf{p})$.

2.4.1 Objective Function

We adopt an "analysis-by-synthesis" strategy to measure how well the transformed and deformed face model fits observed data. Our image-based nonrigid facial registration process aims to minimize the following objective function:

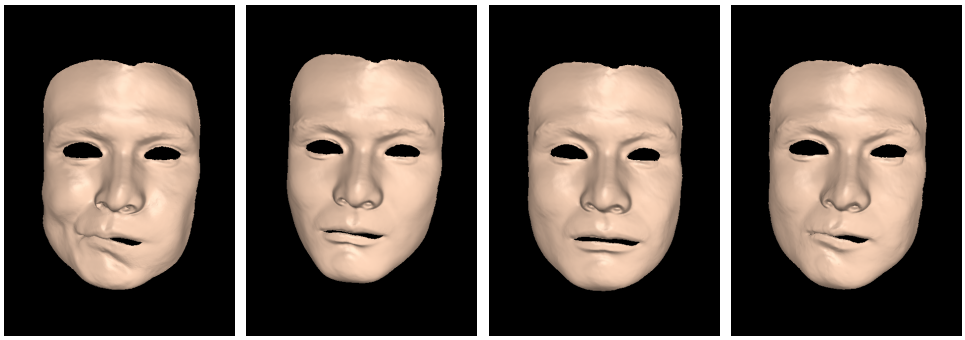
$$\min_{\mathbf{q}} E_{data} + \alpha_1 E_{rot} + \alpha_2 E_{reg}, \tag{2.5}$$

where the first term is the *data fitting* term that measures how well the reconstructed facial model matches the observed data. The second term E_{rot} ensures that local graph nodes deform as rigidly as possible. The third term E_{reg} serves as a regularizer for the deformation by indicating that the affine transformations of adjacent graph nodes should agree with one another.

We define the data fitting term as a weighted combination of three terms:

$$\alpha_{depth} E_{depth} + \alpha_{feature} E_{feature} + \alpha_{boundary} E_{boundary}, \tag{2.6}$$

where the first term E_{depth} is *depth image* term which minimizes the difference between the observed and the rendered depth data. The second term $E_{feature}$ is *facial feature* term which ensures the synthesized facial features are consistent with detected facial features in observed data. Figure 2.5 shows the results of using only facial term, only depth image term, or the combination of the two terms. The third term is *boundary constraint* term which stabilizes the registration process by



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

Figure 2.5: The importance of facial feature detection (on synthetic data): (a) ground truth mesh; (b) reconstructed result using only facial feature term; (c) reconstructed result using only depth image term; (d) final result.

penalizing the misalignments of boundary points between the "hypothesized" and "observed" face models. In our experiment, we set α_1, α_2, $\alpha_{feature}$, α_{depth}, $\alpha_{boundary}$ to 1.0, 0.5, 0.001, 0.1 and 5.

This requires minimizing a sum of squared nonlinear function values. Our idea is to extend the Lucas-Kanade algorithm [4] to solve the above non-linear least squares problem. Lucas-Kanade algorithm, which is a Gauss-Newton gradient descent non-linear optimization algorithm, assumes that a current estimate of $\mathbf{q}$ is known and then iteratively solves for increments to the parameters $\delta\mathbf{q}$ using linear system solvers.

2.4.1.1 Depth Image Term

This section introduces a model-based depth flow algorithm for incrementally estimating rigid transformation (ρ) and nonrigid transformation ($\mathbf{g}$) of the template mesh ($\mathbf{s}_0$) to best match a single depth image D. Assume the movement ($\delta\mathbf{p}$) between the two frames to be small, the depth image constraint at $D(\mathbf{x}, t)$ is defined as follows:

$$D(\mathbf{x}(\mathbf{p}), t) + \delta z = D(\mathbf{x}(\mathbf{p} + \delta\mathbf{p}), t + 1). \tag{2.7}$$

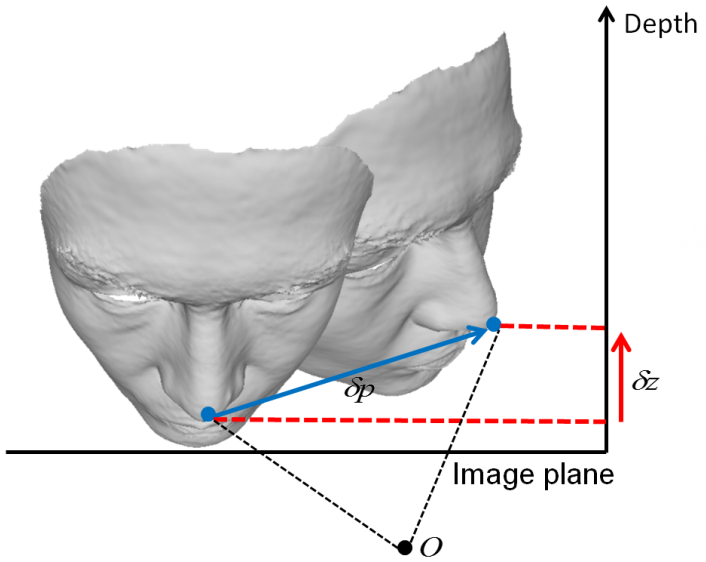


Figure 2.6: Reparameterization of a 3D point produces a corresponding small change along the depth axis.

When a 3D point ($\mathbf{p}$) has a delta movement ($\delta\mathbf{p}$) in 3D space, its projected pixel $\mathbf{x}(\mathbf{p})$ on image plane will have the corresponding movement $\delta\mathbf{x} = (\delta u, \delta v)$. However, unlike color image registration via optical flow, the depth value of a pixel is not constant. Instead, it will produce a corresponding small change δz along the depth axis (Figure 2.6). This is due to reparameterization of 3D point $\mathbf{p}$ on 2D image space.

Similar to optical flow formulation, we approximate the right side of Equation (2.7) with a Taylor series expansion. We have

$$(\frac{\partial D}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial \mathbf{p}} - \frac{\partial z}{\partial \mathbf{p}})\delta\mathbf{p} + \frac{\partial D}{\partial t} = 0, \tag{2.8}$$

where partial derivatives $\partial D/\partial \mathbf{x}$ are gradients of the depth image at pixel $\mathbf{x}$. The derivatives $\partial \mathbf{x}/\partial \mathbf{p}$ can be evaluated by the projection function $\mathbf{f}$. The temporal

derivative $\partial D/\partial t$ simply measures the pixel difference of two depth images. The partial derivatives $\frac{\partial z}{\partial \mathbf{p}}$ is simply a row vector $[0, 0, 1]$.

We adopt an "analysis-by-synthesis" strategy to incrementally register the template mesh with observed depth image via depth flow. More specifically, we render a depth image based on the current model state and estimate an optimal update of the model state by minimizing the inconsistency between the observed and rendered depth image. To register the template face with an observed depth image via depth flow, we associate the delta movement of a point ($\delta\mathbf{p}$) with the delta change of the model state ($\delta\mathbf{q}$):

$$\delta\mathbf{p} = \frac{\partial \mathbf{h}(\mathbf{q}; \mathbf{p}_0, \mathbf{s}_0)}{\partial \mathbf{q}} \delta\mathbf{q}, \tag{2.9}$$

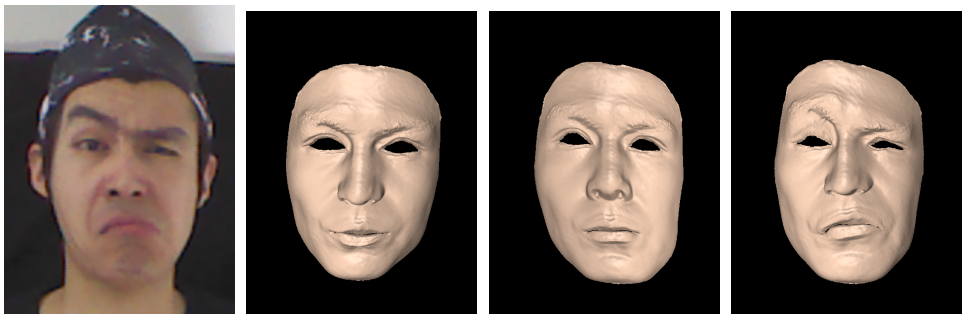
where the vector-valued function $\mathbf{h}$ is the forward kinematics function for mesh deformation.

After combing Equation (2.8) with Equation (2.9) using chain rules, we have

$$(\frac{\partial D}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{p}} - \frac{\partial z}{\partial \mathbf{p}}) \frac{\partial \mathbf{h}}{\partial \mathbf{q}} \delta\mathbf{q} + \frac{\partial D}{\partial t} = 0. \tag{2.10}$$

The above equation shows how to optimally update the model state ($\delta\mathbf{q}$) based on partial derivatives with respect to the spatial and temporal coordinates of the rendered depth image $D(u, v)$. In model-based depth flow, we evaluate the spatial derivative based on the rendered depth image. The temporal derivative is evaluated by the difference between the observed depth image and rendered depth image. An optimal update of the model state can be achieved by summing over the contributions of individual depth pixels associated with the template face.

One remaining issue for the *depth image* term evaluation is to determine which pixels in the "rendered" image should be included for evaluation. We stabilize the



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

Figure 2.7: The importance of facial feature detection: (a) reference image; (b) the face template; (c) result without facial feature term; (d) result with facial feature term.

model-based depth flow estimation process by excluding the pixels located on the border of outer boundary of the face. Corresponding vertices on the template mesh are automatically marked by back projecting the border pixels of the rendered depth image. Similarly, we also remove the pixels that are close to the border of inner boundary of the face, in particular the mouth and eyes.

2.4.1.2 Facial Feature Term

Depth data alone is often not sufficient to model accurate facial deformation because it does not take into account perceptually significant facial features such as nose tip and mouth corners, thereby resulting in misalignments in those perceptually important facial regions. We address this issue by including the *facial feature* term into the objective function. Figure 2.7 shows the importance of facial feature term. In our implementation, we choose to define the *facial feature* term based on a combination of 2D and 3D facial points obtained from detection process.

We annotate facial features on the template mesh model by identifying the local coordinates ($\bar{\mathbf{p}}_i$) of facial features on the template face. The facial feature term

minimizes the inconsistency between rendered features and observed features in either 2D or 3D space:

$$\sum_i w_i \|\mathbf{f}(\mathbf{h}(\mathbf{q}; \bar{\mathbf{p}}_i, \mathbf{s}_0)) - \mathbf{x}_i\|^2 + (1 - w_i)\|\mathbf{h}(\mathbf{q}; \bar{\mathbf{p}}_i, \mathbf{s}_0) - \mathbf{p}_i\|^2, \tag{2.11}$$

where the vector $\mathbf{x}_i$ and $\mathbf{f}_i$ are 2D and 3D coordinates of the i-th detected facial feature. The weight w is a binary value, which returns "1" if depth information is missing, otherwise "0". Note that only facial features around important regions, including the mouth and nose, eyes, and eyebrows, are used for feature term evaluation. This is because facial features located on outer contour are often not very stable.

2.4.1.3 Boundary Term

Depth data from a *Kinect* is often very noisy and frequently contains missing data along the face boundary. This inevitably results in noisy geometry reconstruction around the face boundary. We introduce the boundary term to stabilize the registration along the boundary.

To handle noisy depth data around the outer boundary of the face, we first estimate the rigid-body transformation ρ that aligns the template mesh with observed data (see Section 2.4.2). During nonrigid registration process, we stabilize the outer boundary of the deforming face by penalizing the deviation from the transformed template $\mathbf{s}_0 \oplus \rho$. Vertices on the outer boundary of the template/deforming mesh are automatically marked by back projecting outer boundary pixels of the "rendered" depth image. We define the *boundary* term in 3D position space by minimizing the sum of the squares of the distances between the boundary vertices of the deforming mesh ($\mathbf{s}_0 \oplus \rho \oplus \mathbf{g}$) and their target 3D positions obtained from the transformed mesh ($\mathbf{s}_0 \oplus \rho$).

2.4.2 Registration Optimization

Our 3D facial modeling requires minimizing a sum of squared nonlinear function values defined in Equation (2.5). Our idea is to extend the Lucas-Kanade framework [4] to solve the non-linear least squares problem. Lucas-Kanade algorithm [4], which is a Gauss Newton gradient descent nonlinear optimization algorithm, assumes that a current estimate of $\mathbf{q}$ is known and then iteratively solves for increments to the parameters $\delta\mathbf{q}$ using linear system solvers. In our implementation, we start with the template mesh and iteratively transform and deform the template mesh until the change of the state $\mathbf{q}$ is smaller than a specified threshold.

We have observed that a direct estimation of rigid transformations and embedded deformation is prone to local minima and often produces poor results. We thus decouple rigid transformations from nonrigid deformation and solve them in two sequential steps. In the first step, we drop off the boundary term from the objective function defined in Equation (2.6) and estimate the rigid transformation ρ using iterative linear solvers. We stabilize the rigid alignment by using a pre-segmented template that excludes the chin region from the registration as this part of the face typically exhibits the strongest nonrigid deformations. In the second step, we keep the computed rigid transformation constant and iteratively estimate embedded deformation g based on the objective function defined in Equation (2.5).

This requires minimizing a sum of squared nonlinear function values. Our idea is to extend the Lucas-Kanade algorithm [4] to solve the above non-linear least squares problem using iterative linear system solvers. In our implementation, we analytically evaluate the Jacobian terms of the objective function. The fact that each step in the registration algorithm can be executed in parallel allows implementing a fast solver on modern graphics hardware. By using CUDA to implement our per-frame facial

registration algorithm, the current system runs at an interactive frame rate on a machine with Intel Core i7 3.40GHz CPU and GeForce GTX 580 graphics card.

2.5 Experimental Results and Evaluation

We have evaluated the effectiveness of our system, including both facial feature detection and image-based nonrigid facial registration. We achieve interactive performance (about 15 fps) by executing both facial detection and 3D reconstruction in CUDA.

2.5.1 Evaluation on Facial Feature Detection

The training database for AAMs consists of 750 color images, each of which was annotated manually with 78 facial features (Figure 2.3). In total, there are 45 subjects in the database under different head poses, expressions, illuminations and skin colors. The database used for learning randomized trees is based on 470 RGBD images recorded by a *Kinect*, including 10 subjects with different races and genders. During capture, subjects were instructed to perform six basic expressions in frontal and side views (left and right).

We evaluate the performance of facial detection process on six subjects performing a wide range of facial expressions, which were not in the training database. The whole testing data sets consist of six image sequences (in total, 5616 frames). We measure the accuracy of facial detection by computing the root mean square error (RMSE) between detected feature locations and ground truth locations. In our test, we considered 21 facial features shown in Figure 2.2(b) for evaluation. We excluded secondary facial features because even with manual labeling the ground truth locations of secondary facial features are with large variance and less reliable.

Figure 2.8 shows the comparison results of three methods. We compare our method against AAM registration and facial feature detection. Note that AAM

registration often requires a good initialization for global alignment. In our experiment, we initialized the shape parameter of AAMs with the mean shape; the global transformation of AAMs was initialized using ground truth data. The vertical axis shows the percentage of "perfectly" detected features. Since it is hard to accurately quantify "perfectly" detected features due to inevitable errors caused by manual labeling of ground truth data, we show the percentage of "perfectly" detected features against different thresholds. The experimental results clearly show the robustness and accuracy of our detection method, as well as the necessity of combining detection method with AAMs. Table 2.1 shows the average detection accuracy of each method.

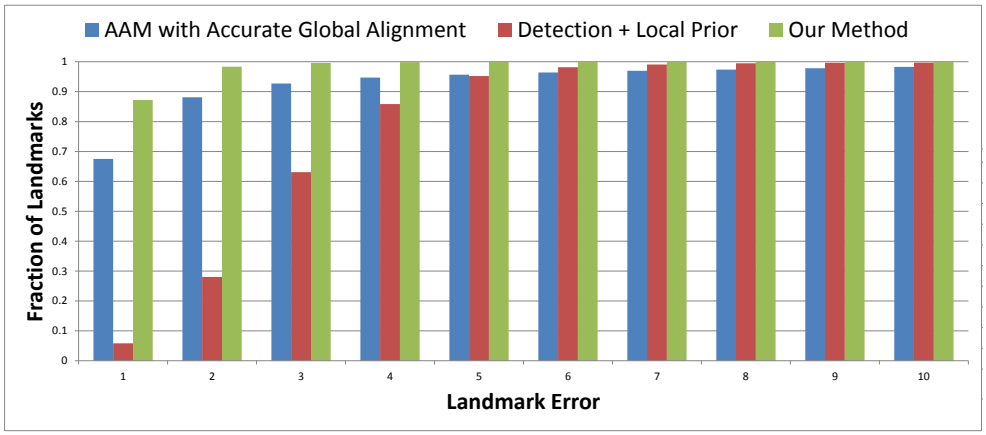


Figure 2.8: Comparisons between AAM with accurate global alignment, detection with local prior optimization and our method.

2.5.2 Evaluation on Image-based Nonrigid Facial Registration

The evaluation consists of two parts. The first part compares our approach with alternative methods. The second part validates the proposed approach by

Table 2.1: Accuracy comparisons (pixel) between AAM with accurate global alignment, detection with local prior optimization and our method.

AAM	Detection + Local Prior	Our Method
1.3061±2.3922	2.7656± 1.3791	0.4337±0.5416

evaluating the importance of each key component of our process. We have evaluated the effectiveness of our algorithm based on both synthetic and real data.

We evaluate our system on synthetic RGBD image data generated by high-fidelity 3D facial performance data captured by Huang and his colleagues [18]. The whole testing sequence consists of 1388 frames. We first synthesize a sequence of color and depth images based on a camera setting similar to what occurs in the real world. The resolutions of image and depth data, therefore, are set to 640×480 and 320×240, respectively, with 24-bit RGB color values and 13-bit integer depth values in millimeters. The face models are placed at a distance to approximate real world capturing scenarios. Figure 2.9 shows sample frames of the synthetic data. We then apply our facial reconstruction process to reconstruct the facial performances across the entire sequence. In this experiments, 236 graph nodes are used to model the template mesh. Sample frames of our constructed results are shown in Figure 2.10 and Figure 2.11.

We test our algorithm as well as alternative methods on synthetic RGBD images and obtain the quantitative error of the algorithm by comparing its reconstruction data against ground truth data. In particular, we compute the average correspondence/reconstruction error between our reconstructed models and the ground truth data across the entire sequence. We evaluate the reconstruction error by measuring the sum of distances between vertex position on the reconstruction mesh and its corresponding position on the ground truth mesh. Note that we know the corre-

spondences between the two meshes because the reconstruction meshes are deformed from the first mesh of ground truth data.

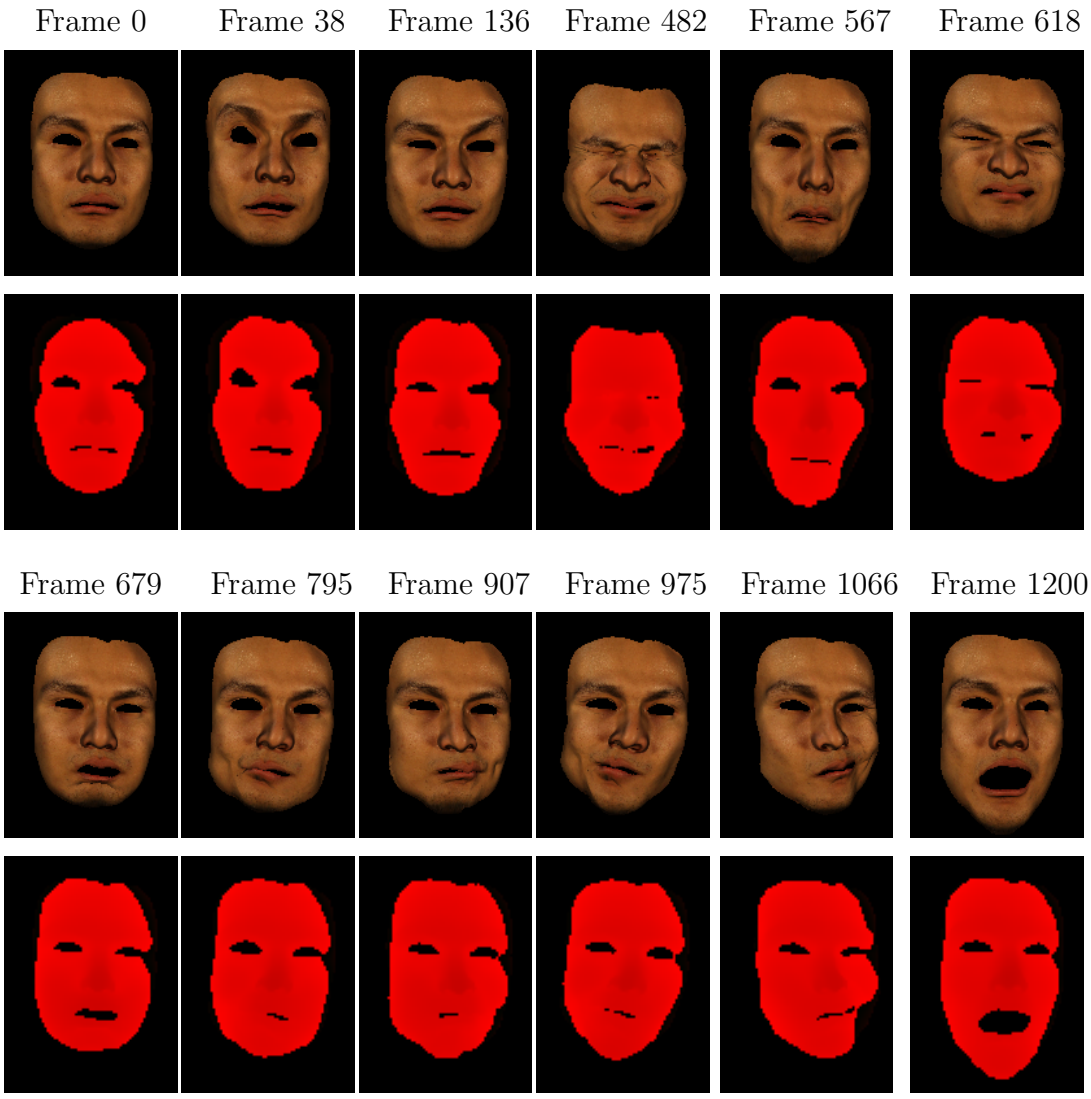
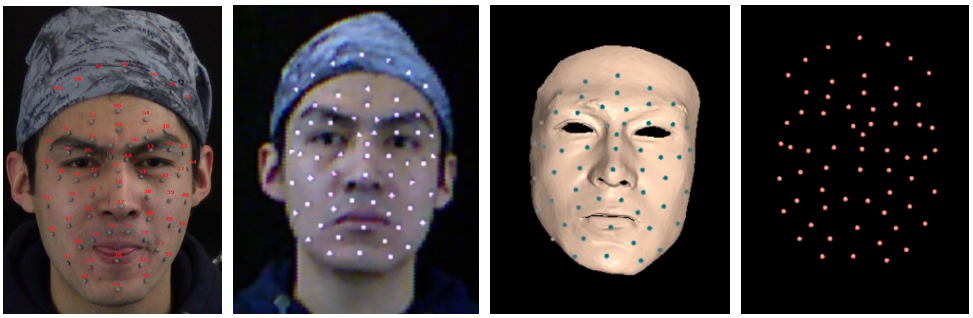


Figure 2.9: Sample frames of the synthetic RGBD images from the data captured by Huang and his colleagues [18] .

Figure 2.10: Our reconstructed results on synthetic data: row (1) shows ground truth facial performances; row (2) shows our reconstructed results; row (3) shows textured result meshes; row (4) shows result meshes with chessboard pattern.

Figure 2.11: Our reconstructed results on synthetic data (continued): row (1) shows ground truth facial performances; row (2) shows our reconstructed results; row (3) shows textured result meshes; row (4) shows result meshes with chessboard pattern.

We further do a real data evaluation by comparing against ground truth facial data acquired by an optical motion capture system [26]. We placed 62 retro-reflective markers (4 mm diameter hemispheres) on the subject's face and set up a twelve-camera Vicon motion capture system [26] to record dynamic facial movements at 240 frames per second acquisition rate. We synchronize a Kinect camera with the optical



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

Figure 2.12: Evaluation of real data with Vicon data: (a) markers on the subject; (b) aligned markers on the observed data from Kinect; (c) corresponding marker locations on the reconstructed model; (d) corresponding marker locations.

motion capture system to record the corresponding RGBD image data. The whole test sequence consists of 838 frames corresponding a wide range of facial expressions. We test our algorithm as well as alternative methods on recorded RGBD image data. We use the 3D trajectories of 62 markers captured by Vicon as ground truth data to evaluate the performance of the algorithm. The captured marker positions and observed RGBD image data are from different coordinate systems and therefore we need to transform them into the same coordinate system. We use the video images from Kinect as a reference to manually label marker positions in Kinect coordinate system and use them to estimate an optimal 4×4 transformation matrix to transform markers from motion capture coordinate system to Kinect coordinate system. Figure 2.12 shows the alignment results.

2.5.2.1 Comparisons Against Alternative Methods

We have evaluated the performance of our system by comparing against alternative methods.

We quantitatively assess the quality of the captured motion by comparing with

Table 2.2: Quantitative evaluation of our algorithm and nonrigid ICP registration (in sequential tracking and per-frame registration) on ground truth data obtained by Vicon system.

	Our Method (per-frame)	Nonrigid ICP (temporal)	Nonrigid ICP (per-frame)
x (2D image plane)	0.64997 pixel	2.2539 pixel	2.5890 pixel
y (2D image plane)	0.85933 pixel	4.5861 pixel	4.0202 pixel
x (3D space)	1.2112 mm	4.5118 mm	4.7829 mm
y (3D space)	1.5817 mm	8.7281 mm	7.5349 mm
z (3D space)	2.1866 mm	12.3468 mm	4.8383 mm

ground truth motion data captured with a full marker set in a twelve-camera *Vicon* system [26]. The average reconstruction error, which is computed as average 3D marker position discrepancy between the reconstructed facial models and the ground truth mocap data, was reported in Table 2.2. The computed quantitative errors provide us an upper bound on the actual errors because of reconstruction errors from the Vicon mocap system and imperfect alignments of the Vicon markers with the Kinect data. Note that we attached the mocap markers on the subject's face, which makes the markers deviate from the actual surface of the face.

We compared our method against non-rigid ICP method on ground truth data obtained from Vicon system. The nonrigid ICP process is similar to [21]. Both our method and nonrigid ICP are built on embedded deformation with the same settings. The only difference is the way to define the data term (*i.e.* E_{data} in Equation (2.5)). Nonrigid ICP process estimates both rigid transformations and embedded deformation with standard iterative closest point techniques (ICP). In each iteration, the corresponding point of each vertex on the deforming mesh is found by its closest point on the observed depth data. The corresponding points of all the vertices are then used to update the deforming mesh. Both methods estimate

a rigid transformation, followed by the non-rigid deformation. Figure 2.13 shows that our system produces much better results than non-rigid ICP either with sequential tracking or per-frame registration. This is because ICP is often sensitive to initial values and prone to local minimum, particularly involving tracking high-dimensional facial mesh model from noisy depth data.

In addition, we have compared our system against nonrigid ICP registration on synthetic data generated by high-quality facial performance data from Huang et al. [18]. We compared them in two different ways: per-frame registration and sequential tracking. Sequential tracking incorporates temporal coherence into facial reconstruction process. More specifically, we include a smoothness term into the objective function to penalize the change of the reconstructed meshes in two consecutive frames. In addition, we utilize the result from previous frame to initialize the current frame. As shown in Figure 2.14, our facial reconstruction produces more accurate results over nonrigid ICP registration, with and without temporal coherence. Sample frames with high errors are shown in Figure 2.15 and Figure 2.16.

We compare our results against Weise et al [36]. We downloaded their tracking software "faceshift" [16]. We started their tracking process by building a personal profile using their system. More specifically, we instructed the user to sit in front of the Kinect camera and recorded a small number of facial expressions to retarget a set of predefined blendshape models to the user. With retargetted blendshape models, we can use their software to sequentially track the facial expression of the user. The accompanying video clearly shows our system produces much more accurate results than their system. Figure 2.17 clearly shows our system produces much more accurate results than their system.

We evaluate the performance of our system by doing a side-by-side comparison against Microsoft Kinect facial SDK. Kinect facial SDK tracks a number of facial

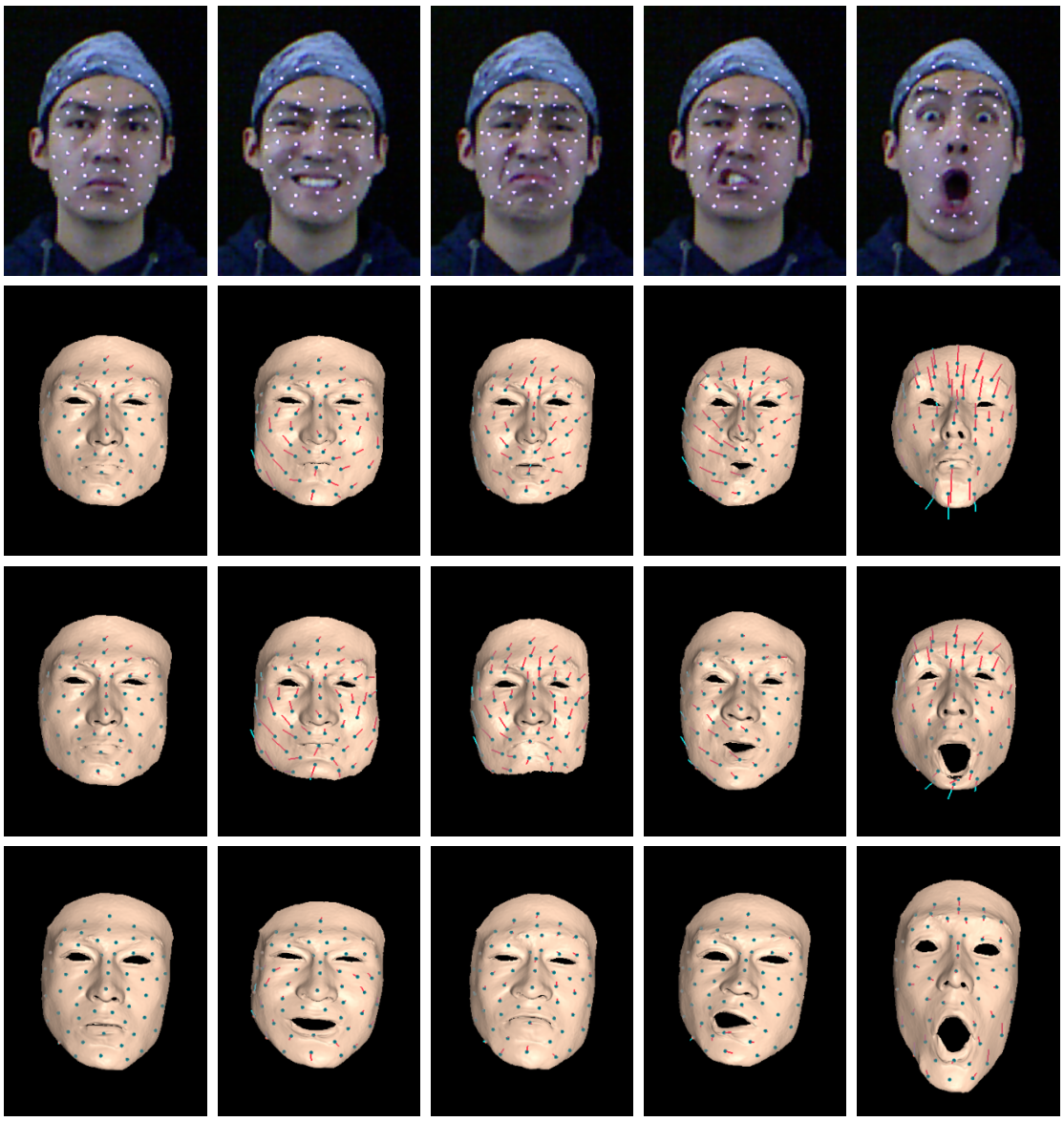


Figure 2.13: Comparison against nonrigid ICP registration techniques in per-frame registration and sequential tracking. Results are compared with ground truth motion data captured with a full marker set in a twelve-camera Vicon system [18]. Green dots denote the corresponding marker locations on the meshes, and the red lines visualize the distance to the aligned Vicon markers. Row (1) shows the reference images and the aligned Vicon markers, row (2) shows the results of nonrigid ICP registration in per-frame registration, row (3) shows the results of nonrigid ICP registration in sequential tracking, , and row (4) shows our results in per-frame registration.

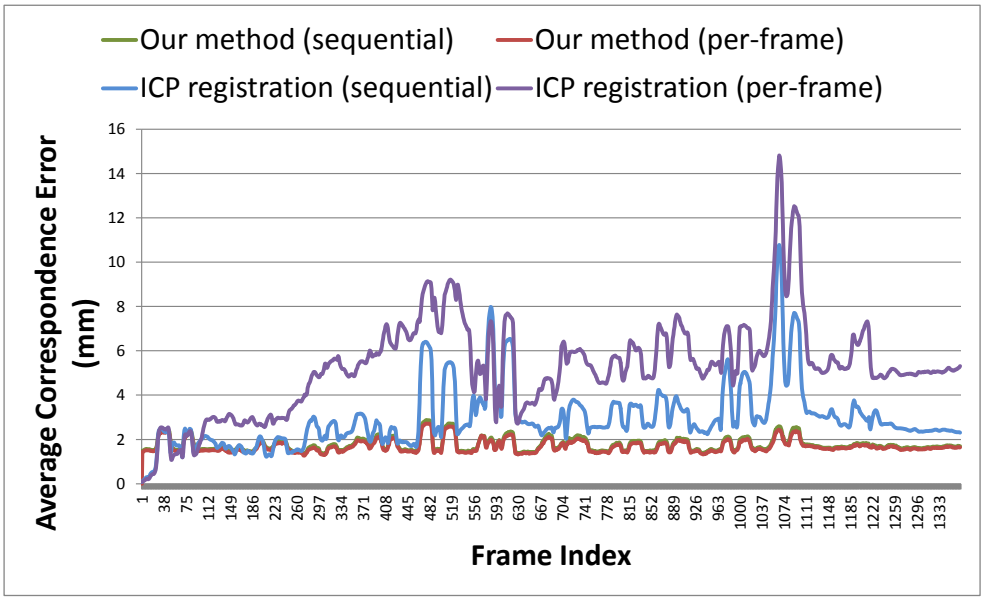


Figure 2.14: Average correspondence/reconstruction errors across the entire sequence for nonrigid ICP registration and our method with/without temporal coherence. The evaluation is based on synthetic RGBD image data.

features (about 100 features) from RGBD image data and combine the tracked facial features with a small number of predefined blendshape models in the Candide3 model, including six AUs (Animation Units) and 11 SUs (Shape Units), to reconstruct facial deformation. Figure 2.17 shows our system produces much better results. This is because we model detailed deformation of the whole face using both facial features and per-pixel depth information.

2.5.2.2 Evaluation on Facial Reconstruction Process

We have evaluated the performance of our tracking process by dropping off each term of the cost function described in Equation 2.6. The evaluation is based on synthetic RGBD data. The average correspondence error of our result to the ground truth meshes is 0.68 mm.

We compare results obtained by the facial registration process with or without

the facial feature term. The accompanying video shows that tracking without the facial feature term often results in misalignments of perceptually important facial features. More importantly, when the facial performances are very different or far away from the template model, the optimization often gets stuck in local minima, thereby producing inaccurate results. With the facial feature term, the algorithm can accurately reconstruct facial performances across the entire sequence. Without the facial feature term, the average correspondence/reconstruction error is increased from 0.68 mm per vertex to 1.7123 mm per vertex. Results are shown in Figure 2.18.

Our evaluation shows that incorporating the depth image term into the objective function can significantly improve the reconstruction accuracy. This is because the facial feature term only constrains facial deformation at locations of a sparse set of facial features rather than detailed per-vertex constraints. With the depth image term, the average error of our facial reconstruction process is reduced from 5.4621 mm per vertex to 0.68 mm per vertex. Results are shown in Figure 2.18.

We compare results of with and without the boundary term on both synthetic data and real data. The result shows that adding the boundary term does stabilize the facial deformation along the border of face boundary, which is more obvious in the real data. Results are shown in Figure 2.18.

2.6 Applications

This section discusses the applications of per-frame facial registration algorithm to high-quality 3D facial performance capture and photo-realistic facial video editing.

We have tested our system on acquiring 3D facial performances of four subjects. We use a Minolta VIVID 910 laser scanner to record high resolution static facial geometry of an actor as the template mesh. In our implementation, we utilize the temporal coherence to speed up the facial tracking process. The algorithm usually

converges quickly as we initialize the solution using the result from the previous frames. Figure 2.19 shows some performance capture results.

Our reconstructed 3D facial models allow the user to edit underlying geometry and/or texture data of facial subjects at any frame across the entire video sequence and propagate the effects of editing to the whole sequence. Figure 2.20 shows our video editing results on several facial performance sequences. For one sequence, we edit its geometry of neutral expression and transfer the reconstructed deformation data to animate the a different geometry model via deformation transfer technique described by [31]. We further texture map the registered image at each frame on the deformed target model throughout the whole motion (Figure 2.21). We also edit the registered texture at the first image by adding a beard onto the texture image; and the texture editing result is then automatically propagated to every frame of the entire sequence, by utilizing the correspondences obtained from our reconstruction facial data.

2.7 Conclusion and Future Work

We present an end-to-end system for acquiring 3D facial performances using a single *Kinect* camera. The proposed system combines the power of automatic facial feature detection and image-based nonrigid facial registration. Our results show that the system can capture a variety of 3D facial performances using low-resolution image and depth data obtained from a single *Kinect* camera. The combination of facial feature detection and facial deformation registration makes our system capable of capturing accurate 3D facial performance automatically. The two components also benefit from each other. On the one hand, we can utilize facial feature detection data to provide good initialization of the face pose and locations and utilize it as effective constraints. On the other hand, the use of image registration reconstruct

dense and consistent correspondences between meshes across frames. Therefore, our reconstructed mesh sequence can be easily edited by the user or artists.

Our system is genera because it requires no prior 3D faces or blendshapes and can easily be used for common users. All it needs is a template model, which in our case, is a face scan under neutral expression. Building a template model directly by using a single Kinect camera is also possible [36]. The template mesh doesn't need to have exactly the same position, orientation, or pose with the first frame. Our system will register the template by doing rigid transformation and non-rigid deformation to fit the mesh to the observed data.

Our system is Robust for its single-frame analysis framework. It does not suffer from drifting error and also ensure the process is fully automatic, together with our proposed robust facial detection method. Last but not the least, our result is accurate because facial feature detection provides locations for significant facial features to avoid misalignments, and our image-based registration method achieves down to sub-pixel accuracy.

The quality of the reconstructed facial performances also depends on the quality of the observed data. The depth images from Kinect camera is noisy, in which case we adjust the weight of E_{reg} to ensure smooth and stable result of facial geometry, which also constrain the freedom of deformation and the model may not fit the data perfectly. One possibility is to filter the noise before fetching into the system. Another solution is to encode temporal coherence to refine the geometry.

While this work focuses on capturing facial performance generally and robustly with accurate rigid transformation and large-scale facial geometry, in the future we are interested in reconstructing realistic dynamic wrinkles and fine-scale facial details.

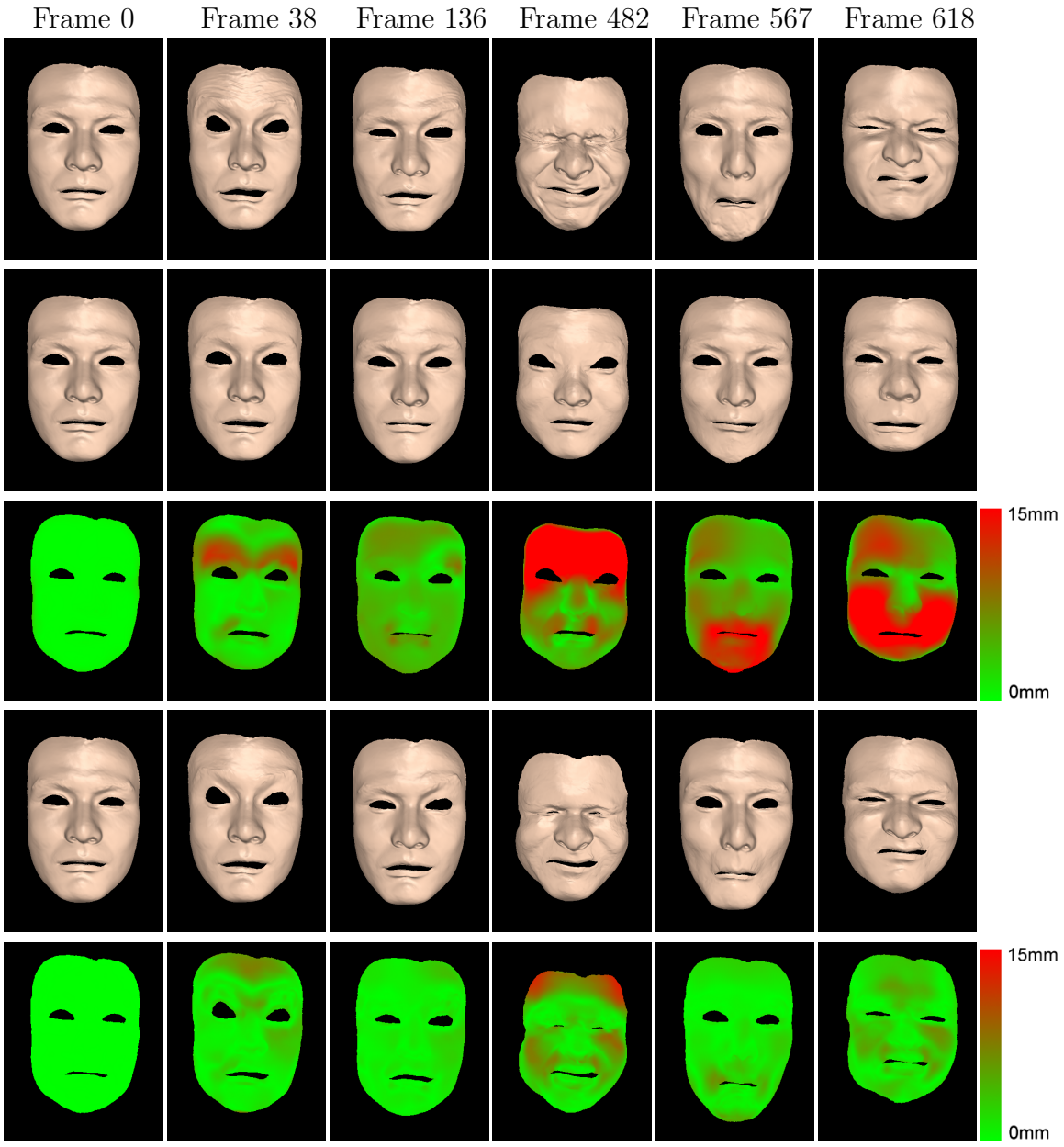


Figure 2.15: Comparison against nonrigid ICP registration on synthetic data generated by high quality facial performance data from Huang et al. [18]. Row (1) shows ground truth facial performances, row (2) shows ICP registration results, row (3) shows correspondence error maps, row (4) shows our reconstruction results, and row (5) shows correspondence error maps of our reconstruction results.

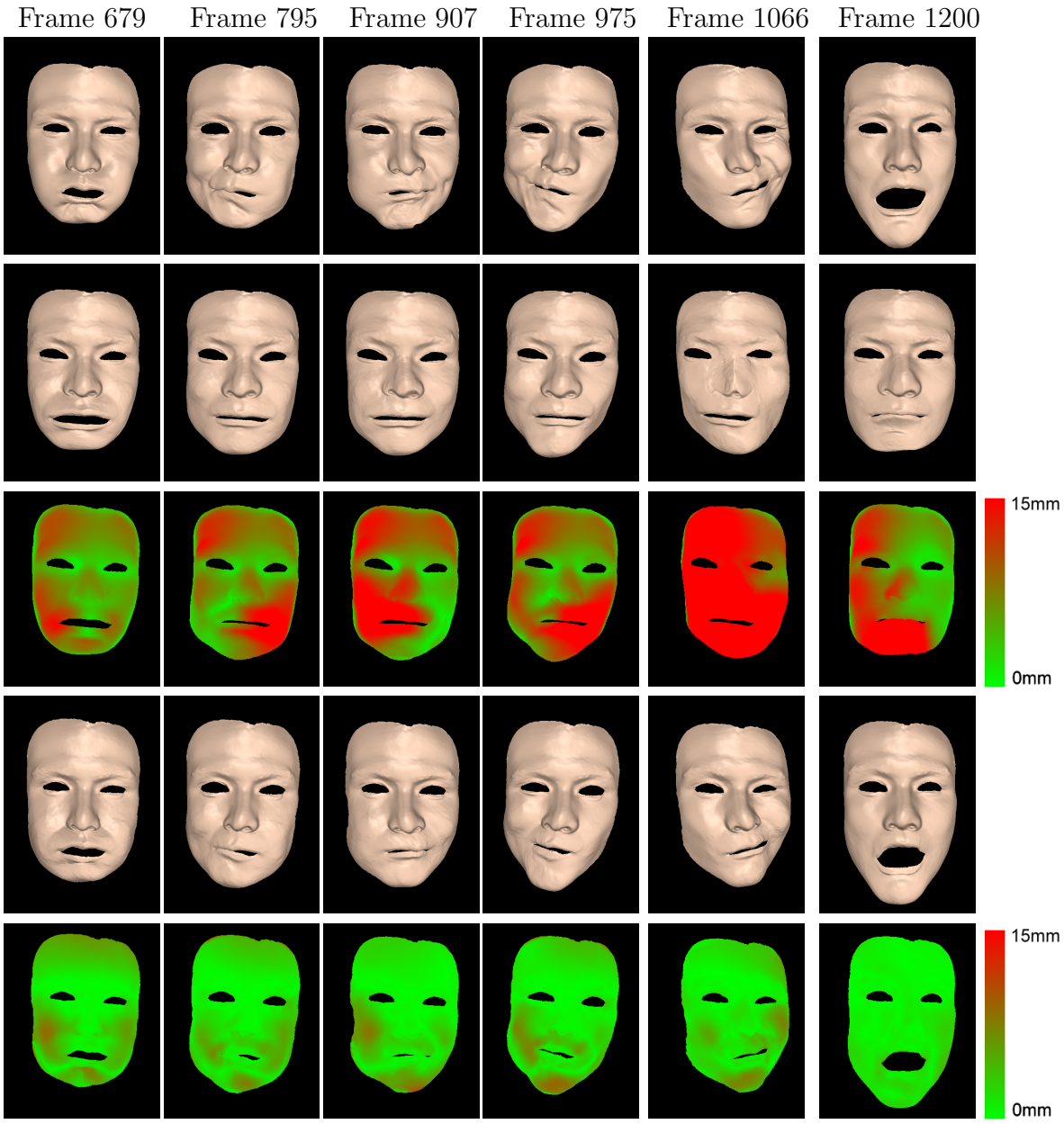


Figure 2.16: Comparison against nonrigid ICP registration on synthetic data generated by high quality facial performance data from Huang et al. [18] (continued). Row (1) shows ground truth facial performances, row (2) shows ICP registration results, row (3) shows correspondence error maps, row (4) shows our reconstruction results, and row (5) shows correspondence error maps of our reconstruction results.

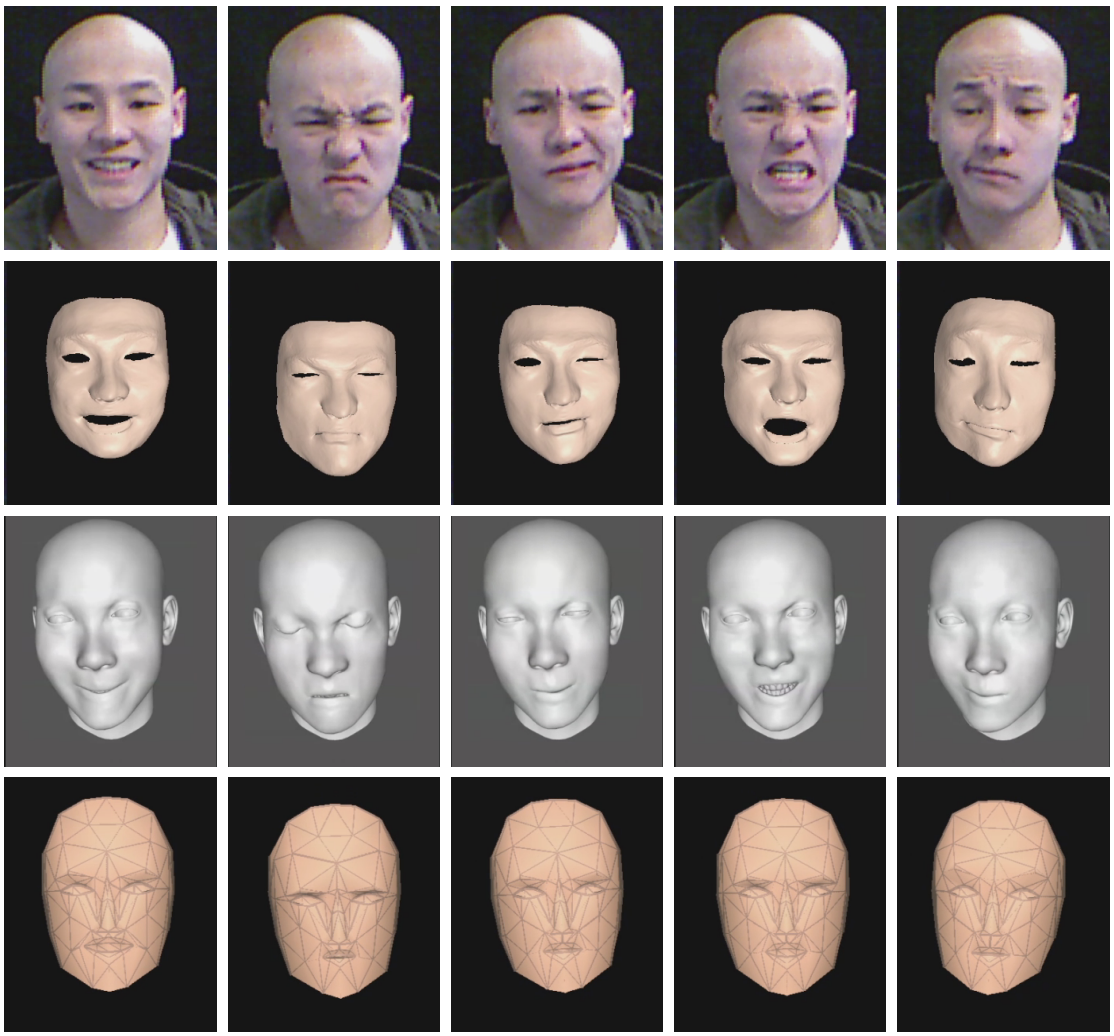


Figure 2.17: Comparisons against Weise et al. [36] and Microsoft Kinect Facial SDK [25]. Row (1) shows the reference images, row (2) shows our results, row (3) shows the results from Weise et al. [36], and row (4) shows the results from Microsoft Kinect Facial SDK [25].

Figure 2.18: Evaluation of our per-frame facial registration process by dropping off each term on synthetic data. Row (1) shows ground truth mesh, row (2) shows reconstructed result without depth term, row (3) shows reconstructed result without facial feature term, row (4) shows reconstructed result without boundary term, and row (5) shows our result.

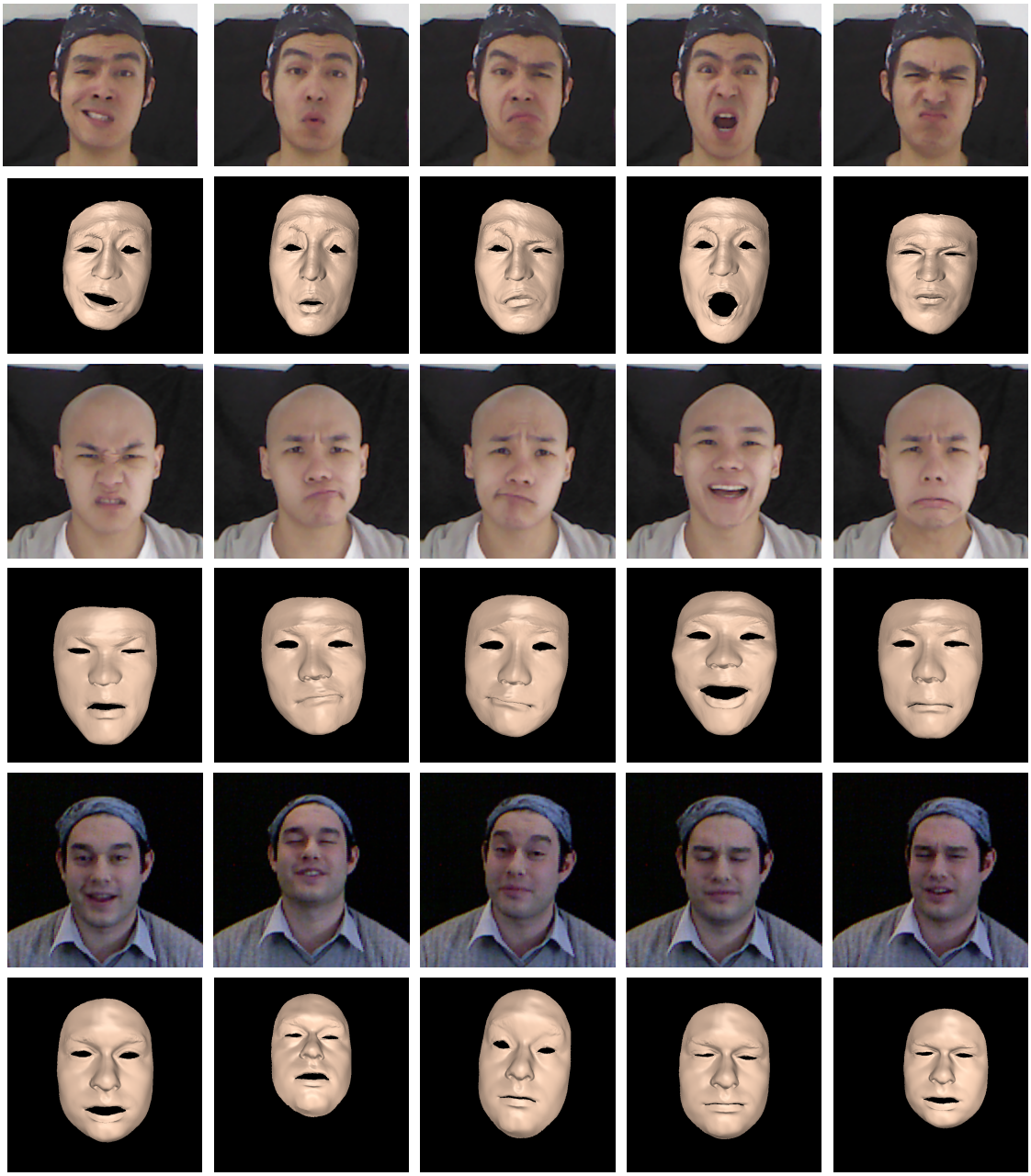


Figure 2.19: Results of our high-quality performance capture system. Rows (1)–(2), rows (3)–(4), and rows (5)–(6) show the reference images and the captured facial performances for Rock, Muscle, and Matt, respectively.

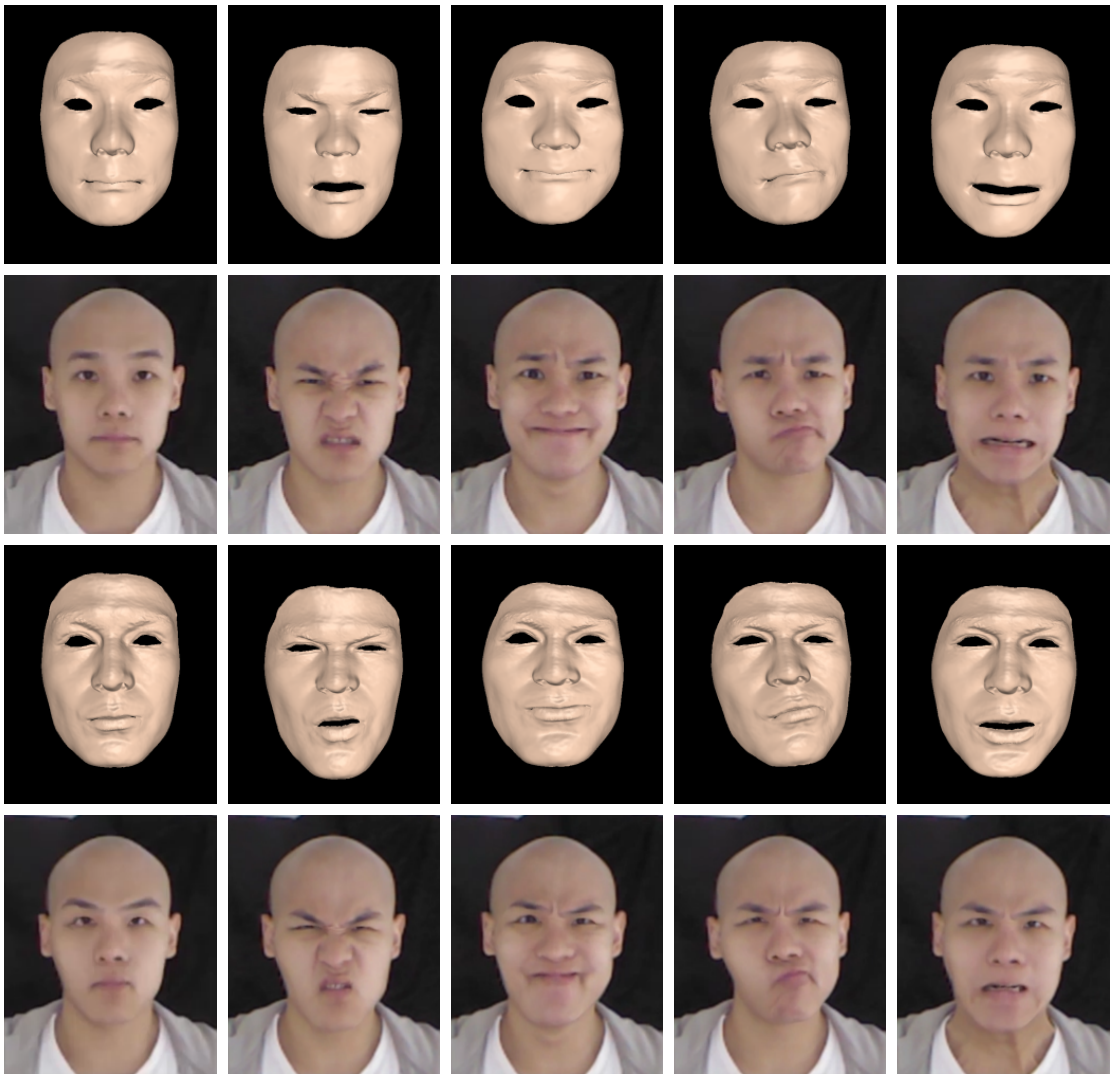


Figure 2.20: Our video editing result of editing the geometry of the first frame (demonstrated in first column) and apply deformation transformation to the rest frames. Row (1) shows the reconstructed meshes, row (2) shows the input reference images, row (3) shows the edited meshes after applying deformation transformation from the reconstructed meshes, and row (4) shows the edited images.

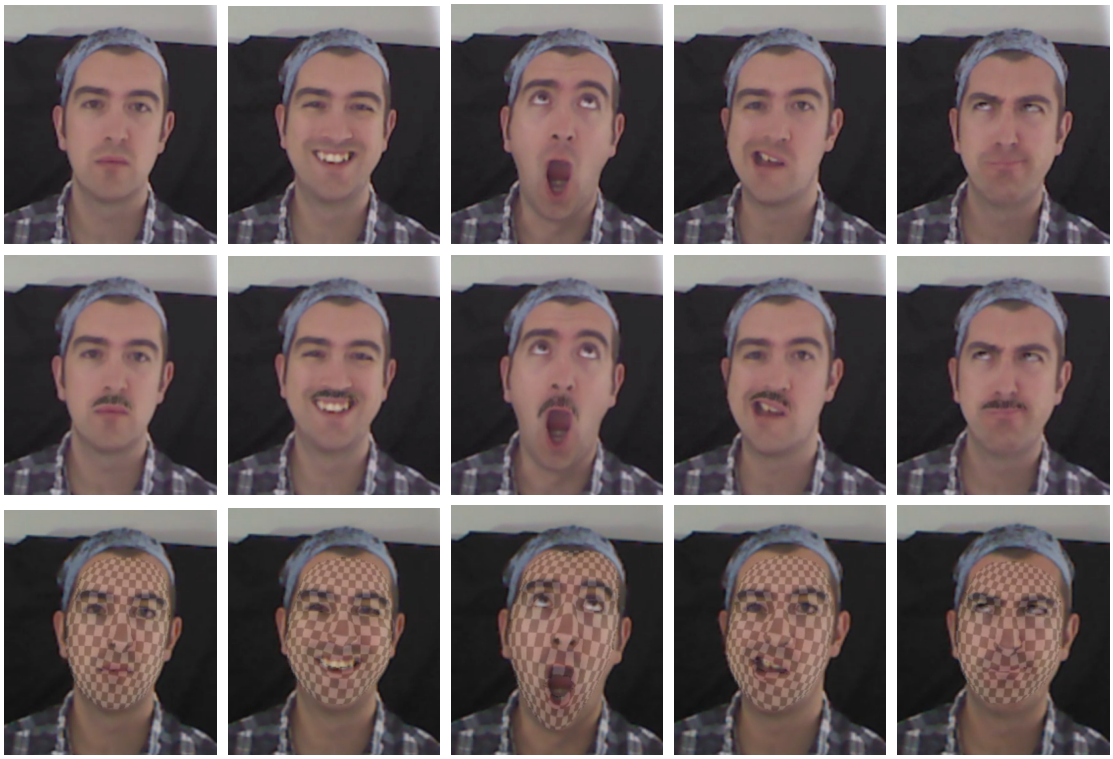


Figure 2.21: Our video editing result of adding a beard onto the texture image at the first frame (demonstrated in first column). Row (1) shows the reference images, Row (2) shows our results of the corresponding images, and row(3) shows the underlying result meshes with chessboard pattern.

3. ACQUIRING HIGH-QUALITY FACIAL PERFORMANCES USING A SINGLE KINECT CAMERA

Capturing detailed 3D facial performances remains challenging because it requires capturing spatial-temporal facial performances involving both large-scale facial deformations and fine-scale geometric details (e.g. wrinkles). We propose a markerless performance capture framework for acquiring high-fidelity facial performances with realistic dynamic wrinkles and fine-scale facial details (Figure 3.1). The key idea of our acquisition system is to leverage image and depth data obtained by a single *Kinect* camera and high-fidelity 3D face scans constructed by a high-resolution 3D scanning system. We choose *Kinect* cameras for facial performance acquisition because they are low-cost, portable, and require no markers or no controlled illumination.

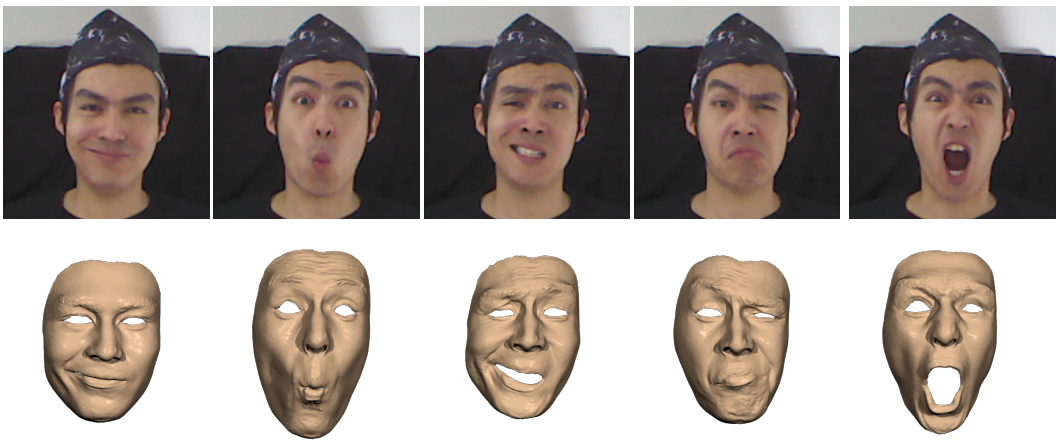


Figure 3.1: Our system captures a variety of facial performances with a single *Kinect* Camera: (top) image data; (bottom) the reconstructed facial performances.

We start the process by recording facial performances of an actor using a single

Kinect camera. Similar to the facial performance capture system presented in Chapter 2, which deforms a 3D template mesh model to match observed data at each frame individually, here we track the whole sequence of image and depth data by incorporating temporal coherence to ensure the reconstructed facial performance is smooth. We then perform automatic facial analysis on the facial tracking data and thereby obtain a minimal set of face scans required for accurate facial reconstruction. We introduce a novel registration process that utilizes image and depth data across the entire sequence to automatically build dense and consistent surface correspondences between all the face scans. Lastly, we combine depth and image data with the minimal set of registered face scans to reconstruct high-fidelity facial performances in a keyframe interpolation framework.

Our high-fidelity facial performance acquisition is made possible by a number of technical contributions:

- A novel facial tracking framework that automatically translates, rotates and deforms a template mesh model to match image and depth data obtained from a single *Kinect* camera.

- An automatic facial analysis technique that determines a minimal set of face scans required for accurate facial performance reconstruction. This not only improves the accuracy of 3D facial reconstruction but also significantly reduces the time and effort required for 3D face scanning.

- A new registration process that automatically builds dense, consistent surface correspondences across all the face scans using image and depth data obtained from a *Kinect* camera. This is nontrivial because face scans often contain high resolution facial details such as pores and wrinkles, and a small misalignment between any two scans will result in unpleasant visual artifacts in the captured

facial performance.

- Finally, an efficient facial reconstruction method that uses observed image and depth data as well as a minimal set of registered face scans to accurately reconstruct facial performances in a keyframe interpolation framework.

3.1 Background

Our system acquires high-fidelity facial performance by leveraging image and depth data captured by a single kinect camera and high-resolution face scans obtained by 3D scanning technology. Therefore, we will focus our discussion on methods and systems developed for acquiring 3D facial performances.

One appealing approach to capturing 3D dynamic faces is image-based facial capture, which deforms a 3D template mesh model to sequentially match input image sequences [15, 14, 29]. Recent effort in this area has been focused on using prior models (*e.g.*, [8, 35]) to reduce the ambiguity of image-based facial deformations. Because these methods make use of generic templates or example-based priors models, the reconstructed geometry and motion do not approach the quality of person-specific captured data. More recent research has been focused on using multi-view stereo reconstruction techniques to improve the resolution and details of captured facial geometry. For example, Bradley and his colleagues [10] reconstructed initial geometry using multi-view stereo reconstruction and used it to capture 3D facial movement by tracking the geometry and texture over time. While their approach produces much higher resolution than previous passive methods, their results still lack such details as pores and wrinkles. Beeler and his colleagues [5] presented an impressive multi-view stereo reconstruction system for capturing the 3D geometry of a face in a single shot and late extended it to acquiring dynamic facial expressions using multiple synchronized cameras [6]. Their system produces fine-scale facial details but the

geometry that is recovered is qualitative and not metrically correct.

An alternative approach for 3D facial capture is to use marker-based motion capture systems [38, 17, 7], which robustly and accurately track a sparse set of facial markers and use them to deform a pre-scanned 3D facial mesh. Recent technological advances in motion capture equipment (*e.g.*, [26]) have made it possible to acquire 3D motion data with stunningly high temporal resolution (up to 2000 Hz), but due to their low spatial resolution (usually less than 200 markers) they are not capable of capturing fine facial details such as wrinkles and bulges. Bickel and his colleges [7] recently augmented the marker-based motion capture system with face paints and two synchronized video cameras for tracking medium-scale expression wrinkles. However, their approach, while powerful, is not appropriate for capturing small wrinkles (*e.g.*, nose wrinkles) and the fine-scale stretching and compression targeted in this system. Most recently, Huang and his colleagues [18] proposed a system that combines the power of marker based motion capture and 3D scanning technology for acquiring fine-scale geometric details in facial performances. Our system is also capable of capturing both large-scale and fine-scale facial deformation. However, our system is based on image and depth data obtained from a single kinect camera instead of marker-based motion capture and therefore is more portable and far less expensive and intrusive.

Structured light systems are capable of capturing 3D models of dynamic faces in real time [40, 23, 21]. One notable example is the spacetime facial capture system developed by Zhang and his colleagues [40]. They captured 3D facial geometry and texture over time and built the correspondences across all the facial geometries by deforming a generic face template to fit the acquired depth data using optical flow computed from image sequences. Ma and his colleagues [23] achieved high-resolution facial reconstructions by interleaving structured light with spherical gradient pho-

tometric stereo using the USC Light Stage. Recently, Li and his colleagues [21] captured dynamic depth maps with their realtime structured light system and fit a smooth template to the captured depth maps using embedded deformation techniques [32]. More recently, Weise and his colleagues [36] presented a realtime facial performance acquisition sysem that combines image and depth data obtained from a Kinect camera with animation priors retargeted from prerecorded facial data. Their system, while powerful for avatar animation and control, cannot capture fine-scale facial details such as wrinkles because it is based on example-based priors from generic subjects.

A number of commercial systems have been developed for 3D facial performance capture in the entertainment industry. For example, Borshukov and his colleagues [9] developed the *Universal Capture* system to recreate actors for *The Matrix Reloaded*. Their system deformed a laser-scanned 3D facial model by using optical flow fields computed from multiple image sequences. Alexander and his colleagues [2] created a photo realistic facial modeling and animation system in the *Digital Emily Project*. Among all the systems, our approach is most similar to [2]. Both systems utilized a number of preselected face scans for high-fidelity facial performance capture. Our approach, however, is different in that we perform quantitative analysis on image and depth data obtained by a single Kienct camera and use it to automatically select a minimal set of facial expressions required for 3D facial performance capture, thereby minimizing the effort and time involved in the scanning process. In addition, we develop an efficient registration process that utilizes image and depth data as well as facial tracking data to automatically build consistent dense surface correspondences across all the scans, which significantly reduces the time and effort required to align all the face scans for facial performance interpolations.

3.2 Overview

Our system acquires high-fidelity facial performances with realistic dynamic wrinkles and fine-scale facial details. The key idea of our system is to leverage image/depth data captured by a single Kinect camera and high-resolution 3D face scans for 3D facial performance acquisition. We choose Microsoft Kinect cameras for facial performance acquisition because it is low cost, portable and non-intrusive. It simultaneously captures depth maps with a resolution of 320×240 and color images with a resolution of 640×480 at 30 frames per second based on infrared projection.

We start the process by recording facial performances of an actor using a Microsoft Kinect camera. We track facial expressions across the entire sequence by deforming a 3D template mesh model to match observed color and depth data. This allows us to obtain both rigid transformations and large-scale facial deformations at each frame. We then perform automatic facial analysis on the facial tracking data and scan a minimal set of keyframe face shapes required for accurate facial reconstruction. We build consistent dense surface correspondences between the keyframe face scans using observed data at intermediate frames. Lastly, we combine observed data with the minimal set of the registered face scans to reconstruct high-fidelity facial performances in the keyframe interpolation framework.

We choose to formulate the facial acquisition and reconstruction process in a keyframe interpolation framework because keyframe interpolation is one of the most popular and successful animation techniques [27]. Mathematically, we model high-fidelity facial performances $\mathbf{z}_t, t = 1, ..., T$ as a weighted interpolation of high-resolution keyframe face meshes $\mathbf{b}_k, k = 1, ..., K$:

$$\mathbf{s}_t = (1 - w_t)\mathbf{b}_k + w_t\mathbf{b}_{k+1}, \quad t_k \leq t < t_{k+1}, \;\; \mathbf{w}_t \geq \mathbf{0}, \tag{3.1}$$

where the scalar w_t represents the weight for interpolating two corresponding key frames located at frame t_k and t_{k+1}. One major benefit of this representation is to decouple spatial details $\mathbf{b}_k, k = 1, ..., K$ from temporal details $\mathbf{w}_t, t = 1, ..., T$. This allows us to obtain high-resolution static facial geometry $\mathbf{b}_k, k = 1, ..., K$ using 3D scanning systems and model high-quality temporal details $w(t), t = 1, ..., T$ using color and depth data obtained from Kinect cameras.

One of the potential problems in using keyframe interpolations for facial modeling and acquisition is that interpolation weights might not be sufficient to capture all the facial details at intermediate frames, thereby requiring a large number of key frames for interpolations. To address this challenge, we apply both rigid transformations and nonrigid deformation to interpolated facial performances to model the residual differences between keyframe interpolations and ground truth facial data. Our final representation for 3D facial performances is therefore formulated as follows:

$$\mathbf{z}_t = ((1 - w_t)\mathbf{b}_k + w_t\mathbf{b}_{k+1}) \oplus \rho_t \oplus \mathbf{g}_t, \quad t_k \leq t < t_{k+1}, \ \mathbf{w}_t \geq \mathbf{0}, \tag{3.2}$$

where the vectors ρ_t and $\mathbf{g}_t$ represent rigid transformations and nonrigid deformation at frame t.

The key challenge for our process is how to accurately reconstruct interpolation weights w_t, rigid transformations ρ_t and nonrigid deformation $\mathbf{g}_t$ from image and depth data O_t obtained by the Kinect camera. In the following, we highlight the issues that are critical for the success of this endeavor and summarize our approach for addressing them.

The first challenge is to scan a minimal set of keyframe face scans $\mathbf{b}_k, k = 1, ..., K$ required for facial performance reconstruction. Minimizing the number of face scans (K) is important because it reduces the time and effort spent on the scanning pro-

cess. This problem, however, is challenging because ground truth facial performance data $\mathbf{z}_t, t = 1, ..., T$ are not available. To address this challenge, we propose an efficient nonrigid registration algorithm to automatically track dynamic facial expressions across the entire sequence. We then introduce an automatic facial analysis process that selects a minimal set of keyframe shapes required for accurate keyframe reconstruction. We scan the 3D geometry of the selected facial shapes by asking the actor to perform the same expressions as shown in the selected key frames. We also register face scans to image and depth data at key frames, which is required by facial performance interpolations.

Keyframe interpolations require building dense, consistent surface correspondences across all the face scans $\mathbf{b}_i, i = 1, ..., K$. This task is challenging because keyframe face scans often display extreme facial expressions and geometric details which appear in one scan might disappear in another one. In addition, face scans often contain high-resolution facial details such as pores and wrinkles. Even a small misalignment will result in unpleasant visual artifacts in the reconstructed facial performance. We propose a novel registration algorithm for aligning all the face scans. Our idea is to utilize facial tracking results at intermediate frames to build dense and consistent correspondences between keyframe shapes.

Our last challenge is how to combine image and depth data with registered keyframe face scans to reconstruct the high-fidelity facial performances across the entire sequence. We formulate the problem in an optimization framework by maximizing the consistency between the reconstructed facial performances $\mathbf{z}_t, t = 1, ..., T$ and observed data O_t. We present an efficient optimization process to estimate interpolation weights, rigid transformations, and nonrigid deformation from observed data. We describe these components in more detail in the following sections.

3.3 Keyframe Extraction and Scanning

This section describes the process of constructing a minimal set of high-resolution face scans for facial performance interpolations. Given a sequence of color and depth images obtained from a Kinect camera, we first track 3D facial expressions by deforming a template face model to match observed data at each frame (Section 3.3.1). We then select a minimal set of key shapes based on facial tracking results and use them to obtain high-resolution face scans (Section 3.3.2). Lastly, we identify all the key frames from the entire sequence and obtain high-resolution facial meshes by registering high-resolution face scans to observed color and depth images at each of key frames (Section 3.3.3).

3.3.1 Template-based facial Tracking

Our system automatically tracks 3D dynamic facial expression by deforming a template mesh model, $\mathbf{s}_0$, to match observed color and depth data at each frame, $O_t = [I_t, D_t], t = 1, ..., T$. We obtain the template mesh model $\mathbf{s}_0$ by scanning facial geometry of the subject under a neutral expression. The system sequentially estimates both rigid transformations (ρ) and nonrigid deformation ($\mathbf{g}$) at each frame by registering deformed template meshes to observed data.

We model nonrigid deformation $\mathbf{g}$ using embedded deformation representation developed by Sumner and his colleagues [32]. Embedded deformation builds a space deformation represented by a collection of affine transformations organized in a graph structure. One affine transformation is associated with each node and induces a deformation on the nearby space. The influence of nearby nodes is blended by the embedded deformation algorithm in order to deform the vertices or the graph nodes themselves. We choose embedded deformation because it allows us to model the deformation in a reduced subspace, thereby significantly reducing the ambiguity for

facial tracking.

In embedded deformation, the affine transformation for individual node is defined by a 3-by-3 matrix $\mathbf{A}_i$ and a 3-by-1 translation vector $\mathbf{t}_i$. In this way, the collection of all per-node affine transformations, denoted as $\mathbf{g} = \{\mathbf{A}_i, \mathbf{t}_i\}_{i=1,\ldots,M}$, expresses a non-rigid deformation of the template mesh model in a reduced deformation space. In our experiment, graph nodes are chosen by uniformly sampling vertices of the template mesh model in the frontal facial region. We have found 300 graph nodes are often sufficient to model large-scale deformation of facial expressions.

We formulate the facial tracking process in a model-based registration framework and adopt the "analysis-by-synthesis" strategy to sequentially register the template mesh model to observed data. Let $\mathbf{q} = [\rho, \mathbf{g}]$ denote the state of our tracking process. The 3D representation of facial performances at frame t is thus defined as $\mathbf{s}_t = \mathbf{s}_0 \oplus \mathbf{q}_t$. We have

$$\min_{\mathbf{q}_t} \alpha_{rot} E_{rot} + \alpha_{reg} E_{reg} + \mathrm{M}(O_t, \mathbf{s}_0 \oplus \mathbf{q}_t), \tag{3.3}$$

where the first term E_{rot} ensures that local graph nodes deform as rigidly as possible. The second term E_{reg} serves as a regularizer for the deformation by indicating that the affine transformations of adjacent graph nodes should agree with one another. For details of E_{rot} and E_{reg}, please refer to [32]. The last term is the *data fitting* term that measures how well the reconstructed facial performance matches the observed data. In our experiment, we define the data fitting term $\mathrm{M}(O, \mathbf{s}_0)$ as a weighted combination of image term, depth data and facial feature term:

$$\alpha_{img} E_{img} + \alpha_{depth} E_{depth} + \alpha_{aam} E_{aam}, \tag{3.4}$$

where E_{img}, E_{depth}, and E_{aam} ensure the transformed template mesh is consistent

with observed data, including both image and depth data, as well as important facial features automatically extracted by Active Appearance Models (AAM) techniques [13, 24]. In our experiment, we set α_{rot}, α_{reg}, α_{img}, α_{depth}, α_{aam} to 1.0, 0.5, 0.001, 0.1 and 3.0.

The image term, E_{img}, measures how well the reconstructed facial performances match observed image data. Let $\mathbf{p} = [x; y; z]$ be the 3D coordinates of a point on the reconstructed face mesh and $\mathbf{x} = [u; v]$ be the 2D coordinates of the corresponding pixel on the image plane. We have

$$E_{img} = \sum \| I_{render}(\mathbf{x}(\mathbf{p}; \mathbf{q})) - I_{observe} \|^2, \tag{3.5}$$

where $\mathbf{x}(\mathbf{p}; \mathbf{q})$ maps a 3D point $\mathbf{p}$ on the reconstructed facial mesh to a 2D pixel $\mathbf{x}$ in the "rendered" image space. The reconstructed face mesh is specified by the tracking state $\mathbf{q}$, which includes both rigid transformations ρ and non-rigid deformation $\mathbf{q}$. Since our system runs in a sequential mode, the colors of rendered pixels can be retrieved from registered images in a previous registered frame.

The depth term, E_{depth}, ensures that the reconstructed face mesh is consistent with observed depth data D. We render the depth data using the reconstructed face mesh and evaluate the difference between the rendered and observed depth data. The cost function is defined as follows:

$$E_{depth} = \sum \| D_{render}(\mathbf{x}(\mathbf{p}; \mathbf{q}), \mathbf{q}) - D_{observe} \|^2, \tag{3.6}$$

where $\mathbf{x}(\mathbf{p}; \mathbf{q}) = (u, v)^T$ is a column vector containing the pixel coordinates of "rendered" depth images and is computed in the same way as the image term evaluation. However, unlike the image term, pixel values D_{render} in the depth term are not fully

dependent on pixel coordinates $\mathbf{x}(\mathbf{p};\mathbf{q})$ because they also directly vary with transformations $\mathbf{q}$. This is due to reparameterization of 3D depth data using 2D depth images.

The feature term, E_{aam}, measures how well facial features in a rendered image match facial features in the observed image. This term minimizes the sum of squared distances between the rendered and observed features:

$$E_{aam} = \sum_i \|\mathbf{x}(\mathbf{q};\mathbf{p}_i) - \mathbf{f}_i\|^2, \tag{3.7}$$

where the vector $\mathbf{f}_i$ are 2D coordinates of observed facial features, which are automatically detected by Active Appearance Models (AAM) techniques [24]. The vector $\mathbf{p}_i$ are 3D coordinates of facial features on the template mesh model. We annotate facial features on the template mesh model in advance. This allows us to automatically locate facial features in the rendered images.

To ensure the reconstructed facial performance is smooth, we add an extra term, E_{smooth}, to penalize the sudden changes of the tracking states between two consecutive frames:

$$E_{smooth} = \|\mathbf{q}_t - \mathbf{q}_{t-1}\|^2, \tag{3.8}$$

where $\mathbf{q}_t$ and $\mathbf{q}_{t-1}$ represent tracking states in the current frame and previous frame, respectively.

Reconstruction of facial meshes at each frame requires minimizing a sum of squared nonlinear function values. Our solution is to extend the Lucas-Kanade algorithm [4] to solve the non-linear least squares problem. Lucas-Kanade algorithm, which is a Gauss Newton gradient descent non-linear optimization algorithm, assumes that a current estimate of $\mathbf{q}$ is known and then iteratively solves for increments

to the parameters $\delta\mathbf{q}$ using linear system solvers. In our implementation, we initialize the current pose using the previous reconstructed pose and iteratively update $\mathbf{q}$ until the change of the state $\mathbf{q}$ is smaller than a specified threshold. The algorithm usually converges within fifteen iterations as we initialize the solution using the previous tracking poses.

In our experiments, we have observed that a direct estimation of rigid transformations and embedded deformation is prone to local minima and often produces poor results. We thus decouple rigid transformations from nonrigid deformation and solve them in two sequential steps. In the first step, we drop off E_{rot} and E_{reg} from the objective function and estimate the rigid transformation ρ using iterative linear solvers. In the second step, we keep the estimated rigid transformation constant and estimate the nonrigid transformation $\mathbf{g}$ iteratively based on the complete objective function defined in Equation (3.4). We denote the estimated rigid transformation and nonrigid deformation as $\tilde{\rho}$ and $\tilde{\mathbf{g}}$, respectively.

3.3.2 Key Shape Selection and Scanning

We now discuss how to use facial tracking data, $\tilde{\mathbf{q}}_t = [\tilde{\rho}_t, \tilde{\mathbf{q}}_t], t = 1, ..., T$, to select a minimum set of face scans $\mathbf{b}_k, k = 1, ...K$ for keyframe interpolations. This task is nontrivial because it requires determining not only the minimal number of key shapes (K) but also the corresponding key shapes ($\mathbf{b}_1, ..., \mathbf{b}_K$) for 3D scanning.

We formulate key shape selection as a data clustering problem and select key shapes based on the centers of each cluster. The intuition here is that by clustering facial performances at all the frames and choosing a key shape in each cluster, coverage of the entire range of motion is ensured. In our implementation, we choose K-medoids techniques for data clustering because we want to ensure that cluster centers are selected based on original frames instead of centroids of the frames. K-

medoids is also more robust to noise and outliers as compared to standard K-means clustering techniques because it minimizes a sum of general pairwise dissimilarities instead of a sum of squared Euclidean distances.

We define the distance between two frames, *e.g.* frame i and frame j, as follows:

$$\mathrm{H}(i,j) = \alpha_1 \mathrm{H}_{def}(i,j) + \alpha_2 \mathrm{H}_{aam}(i,j) + \alpha_3 \mathrm{H}_{img}(i,j), \tag{3.9}$$

where the first term $\mathrm{H}_{def}(i,j)$ measures the similarity of nonrigid deformation at frame i and j. The nonrigid deformation at each frame is specified by embedded deformation $\tilde{\mathbf{g}}$ obtained from facial tracking process. Specifically, the function H_{def} evaluates the average vertex distance between the two deformed meshes, $\mathbf{s}_0 \oplus \tilde{\mathbf{g}}_i$ and $\mathbf{s}_0 \oplus \tilde{\mathbf{g}}_j$. The second term $\mathrm{H}_{aam}(i,j)$ considers the average distance of corresponding facial features on the deform meshes located at frame i and j. Similar to tracking process, AAM features are used to evaluate this term. The third term $\mathrm{H}_{img}(i,j)$ evaluates the average differences of the colors for corresponding vertices located on the deformed meshes as frame i and j, where vertex colors are obtained by texturing mapping the registered image data onto the deformed meshes. In our experiment, the weights α_1, α_2 and α_3 are set to 0.1, 0.001 and 3 respectively. Note that we remove the effect of rigid transformations $\tilde{\rho}$ when evaluating all three terms.

One remaining issue is to determine a minimal number of key shapes (K) for 3D scanning because clustering processes such as K-medoids often assume the number of clusters is known in advance. We address this challenge by obtaining initial clusters via K-medoids and recursively merging or splitting clusters until clustering errors fall below a threshold. Two clusters are merged if their distance is smaller than the threshold. A cluster is split when the clustering error is higher than the threshold. This process continues recursively until no further splits or merges are possible. In

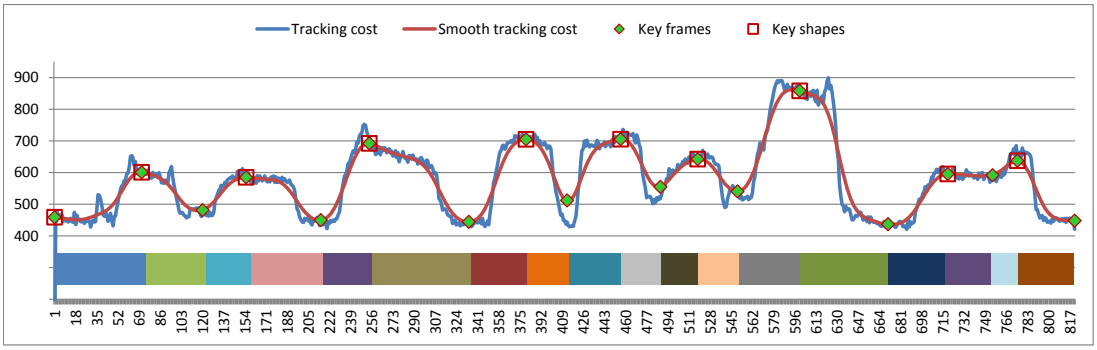


Figure 3.2: Key shape extraction and key frame identification: (top) the blue and red curve show the original and smoothed tracking errors across the entire sequence, respectively. Note that the same key shapes might appear at different times and be associated with different a number of key frames in the sequence. (bottom) the whole sequence is then divided into multiple segments using all the identified key frames. Facial performance reconstruction is achieved by keyframe interpolations and refinement in each segment.

our implementation, initial clusters are obtained by applying K-medoids to a selected set of input frames instead of all the input frames. The initial cluster number and the threshold are experimentally set to 5 and 20. We limit the initial clusters to extreme facial expressions because key frames/shapes are often corresponding to extreme poses of the animation. The system automatically detects all extreme facial expressions based on the tracking errors obtained from facial tracking process. To achieve this, we smooth the entire tracking error curve and extract all local minima or maxima of the curve[1]. Figure 3.2 shows the original and smoothed tracking error curve as well as the locations of all the key shapes on the timeline via clustering process. Figure 3.3 and Figure 3.4 show the grouped keyframes, while Figure 3.5 shows the final key shapes.

Given a minimal set of key shapes located at frames, $t_1, ..., t_K$, we can look up the

[1]The first and last frame of the input sequence are always included in the extreme facial expressions set.

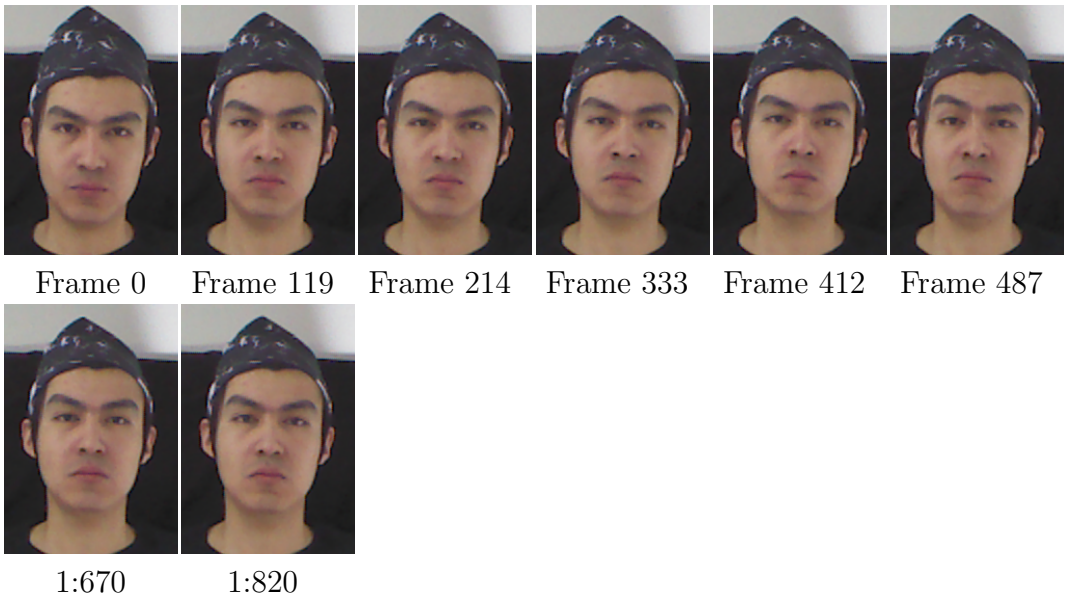


Figure 3.3: Key frames that are grouped to the same cluster of frame 0.

corresponding images $\mathbf{I}_{t_1}, ..., \mathbf{I}_{t_K}$ and use them as references to scan high-resolution facial meshes $\mathbf{b}_1, ..., \mathbf{b}_K$. We use a Minolta VIVID 910 laser scanner to record high-resolution static facial geometry of an actor (see Figure 3.6). During each scan, VIVID 910 acquires a face mesh with 100k to 200k vertices in about 2.5 seconds and achieves an accuracy of 0.008 mm on the x-y plane and 0.1 mm along the Z-axis. The scanned meshes $\mathbf{b}_k, k = 1, ..., K$ are high-resolution and display subtle spatial details such as pores and wrinkles.

3.3.3 Keyframe Identification and Registration

To reconstruct facial performances using keyframe interpolations, we need to identify the start and end key frames for every input frame. For nonrepetitive facial performances, the scanned faces $\mathbf{b}_1, ..., \mathbf{b}_K$ define all the key frames $t_1, ..., t_K$ required for interpolations. In practice, facial performances often contain repetitive facial expressions and poses. As a result, the same key shapes might appear at different

Frame 754 Frame 774

Figure 3.4: Key frames that are grouped to the same cluster of frame 754.

Frame 0 Frame 70 Frame 154 Frame 253 Frame 379

Frame 455 Frame 517 Frame 599 Frame 718 Frame 774

Figure 3.5: Final key shapes.

(a) (b) (c)

Figure 3.6: Key shape scanning: (a) a reference facial expression selected by key shape selection process; (b) scanning static facial geometry of an actor use a Minolta VIVID 910 laser scanner; (c) the scanned high-resolution facial mesh.

times and be associated with different frames in the sequence. This section focuses on how to identify all the key frames and how to register the scanned key shapes to them.

Our idea is to utilize the clustering results to identify all the frames which are associated with key frames and include them into the list of key frames for facial interpolations. Specifically, we segment the whole sequence into multiple subsequences, each of which consists of a window of frames assigned to the same cluster. For each subsequence, we extract the frame corresponding to the smallest clustering error and include it into key frames. Once we obtain all the key frames, we can use them to divide the whole sequence into multiple segments (see Figure 3.2). For each segment, we interpolate facial expressions at intermediate frames using key shapes associated with the start and end frames of the segment.

Keyframe interpolation also requires registering face scans with observed data at key frames. This registration process involves not only rigid transformations between the two capturing systems but also non-rigid deformation caused by possible differences between the "reference" expressions and the "performed" expressions. We

solve the registration problem in a similar way as template-based facial registration described in Section 3.3.1, except that we initialize the rigid transformations by aligning AAM features on face scans with corresponding features at keyframe images.

3.4 Face Scans Registration

After registering face scans with image and depth data at every key frame, we obtain a set of face scans required for keyframe interpolations. However, keyframe interpolations require dense, consistent surface correspondences across all the scans. This section describes a novel two-step registration algorithm that achieves this goal.

3.4.1 Mesh Registration and Resampling

Our goal herein is to build dense, consistent correspondences across all the face scans $\mathbf{b}_k, k = 1, ..., K$. To achieve this goal, we select one of the face scans $\mathbf{s}_0$ as a template mesh and deform the template mesh to precisely matches the face scans $\mathbf{b}_k$. The deformed template meshes, denoted as $\mathbf{d}_k, k = 1, ..., K$, preserve all the fine details in the face scans $\mathbf{b}_k$. However, unlike the target meshes $\mathbf{b}_k, k = 1, ..., K$, the deformed meshes $\mathbf{d}_k, k = 1, ..., K$ have the same topology as the template mesh $\mathbf{s}_0$ and therefore are amenable for keyframe interpolations. We call this process as mesh resampling.

Mathematically, we obtain the resampled meshes $\mathbf{d}_k$ by minimizing the following objective function:

$$\arg\min_{\mathbf{d}_k} \; w_1 dist^2(\mathbf{d}_k, \mathbf{b}_k) + w_2 \|L(\mathbf{v}_d) - L(\mathbf{v}_{\mathbf{s}_0})\|^2, \tag{3.10}$$

where the first term measures how well vertices on the deformed template mesh $\mathbf{d}_k$ are mapped to the corresponding vertices on the target mesh $\mathbf{b}_k$. The second term is the Laplacian term, which preserves the fine details of the original template mesh

$\mathbf{s}_0$ and is mainly used to regularize the solution space. The operator L is the Laplace operator of the mesh model [1, 30]. The weights w_1 and w_2 control the importance of each term, respectively. Note that we factor out the rigid transformation of each scan before we perform the large-scale mesh registration process. This process, however, requires dense correspondences between the template mesh $\mathbf{s}_0$ and face scans $\mathbf{b}_k$ because the deformed template $\mathbf{d}_k$ has the same topology as the template mesh $\mathbf{s}_0$.

So how can we build dense correspondences between the template mesh and the face scans? Our solution is to register both the template mesh and face scans with observed images at key frames. Note that we have already obtained dense correspondences between the template mesh and observed images at each frame via the template-based facial tracking process described in Section 3.3.1. And dense correspondences between the face scan and the keyframe images can be obtained by keyframe registration step described in Section 3.3.3. Specifically, for each vertex on the template mesh $\mathbf{s}_0$, we project it onto the 2D image space to find a corresponding pixel at a key frame $\mathbf{t}_k, k = 1, ..., K$. We then ray cast the pixel into 3D space to find its corresponding point on the face scans $\mathbf{b}_k, k = 1, ..., K$. For vertices invisible to the camera, we find the corresponding points between the two by searching the closest points between the face scan $\mathbf{b}_k$ and the deformed template $\mathbf{s}_0 \oplus \mathbf{g}_{t_k}$ obtained from the template based facial tracking process.

To solve the optimization defined in Equation (3.10), we initialize the deformed mesh $\mathbf{d}_k$ by deforming the template mesh $\mathbf{s}_0$ to match corresponding 3D points on the target mesh $\mathbf{b}_k$ via Laplacian deformation techniques (see Figure 3.7(b)). After that, we iteratively find the closest points between the deformed mesh $\mathbf{d}_k$ and the target mesh $\mathbf{b}_k$ and use them to deform the template mesh with least-squares techniques. We experimentally set the weights of w_1 and w_2 to 1 and 30, respectively. The optimization typically converges in three iterations because of

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

Figure 3.7: Mesh registration deforms the template mesh to fit every face scan: (a) the template mesh $\mathbf{s}_0$; (b) the initial deformed template mesh $\mathbf{d}_k$; (c) the final deformed mesh $\mathbf{d}_k$; (d) the target mesh $\mathbf{b}_k$. Note that the final deformed mesh $\mathbf{d}_k$ preserves all the fine details in the target mesh $\mathbf{b}_k$ while still having the same topology as the template mesh $\mathbf{s}_0$.

very good initializations. During the iterations, we gradually decrease the weight for the second term (w_2) from 30, to 20, to 10, to 1 in order to ensure that the final deformed mesh can precisely match the target mesh without triangle flipping artifacts. Figure 3.7(c) and (d) show a side-by-side comparison between the final deformed mesh and the target mesh.

3.4.2 Registration Refinement

Mesh registration and resampling ensures all the face scans are topologically consistent. Meanwhile, we have built dense surface correspondences between all the face scans. But the quality of mesh registration is highly dependent on the accuracy of the template-based facial tracking process. A single template mesh, however, often cannot be deformed to match fine-scale facial expressions such as wrinkles. This is because geometric details such as wrinkles are expression dependent–the geometric details that appear in one face scan might disappear in another one. As a result, mesh registration and resampling often fails to register fine-scale geometric details on the

Figure 3.8: Testing on synthetic data: (a) ground truth facial performances; (b) our final facial reconstruction results; (c) the facial tracking results obtained from the template-based facial tracking described in Section 3.3.1.

face scans, thereby producing unpleasant visual artifacts in keyframe interpolations. We propose to improve facial tracking results by utilizing all the keyframe face scans and then use the improved tracking results to refine mesh registration.

Based on all the key frames obtained from Section 3.3.3, we first divide the whole sequence into multiple segments and then track facial expressions in each segment using the start key frame of the segment. More specifically, for each segment, we track facial expressions across the entire segment by deforming the start and end key frames to match observed data at each frame. In addition, we deform the end key frame to match observed data at each frame in reverse order.

We now discuss how to refine mesh registration results using forward and backward tracking results at each segment. We formulate the process as the following per-vertex optimization problem:

$$\min_{\mathbf{d}'_1,\ldots,\mathbf{d}'_K} \sum_{k=1}^{K} \sum_{m=1}^{N_k} G(U_{k_m}, V_{k_m}; \mathbf{d}'_k, \mathbf{d}'_{k_m}), \qquad \mathbf{d}'_k \in Surf(\mathbf{d}_k), \tag{3.11}$$

where $\mathbf{d}'_k$ represents the refined mesh for the k-th face scan $\mathbf{d}_k$. Two face scans are neighbors if they are the start or end key frames of the same segment. $\mathbf{d}'_{k_m}$ is the m-th neighbor of the k-th face scan $\mathbf{d}'_k$ and N_k is the total number of neighbors for the k-th face scan $\mathbf{d}_k$. A face scan could have more than two neighbors because facial performances might contain repetitive expressions. Note that the goal here is to refine the correspondences across all the face scans rather than change the underlying geometry and topology of the face scans. As a result, we keep the topology of the original face scans $\mathbf{d}_k$ and constrain the new vertices $\mathbf{d}'_k$ to a point on a surface of the old face scan $\mathbf{d}_k$: $\mathbf{d}'_k \in Surf(\mathbf{d}_k)$.

Intuitively, the objective function G measures the alignment inconsistency between face scans and their neighbors. We use both forward and backward tracking results, denoted as U_{k_m} and V_{k_m}, to evaluate the discrepancy between the k-th face scan $\mathbf{d}_k$ and its m-th neighbor $\mathbf{d}_{k_m}$. Assume $\mathbf{d}_k$ and $\mathbf{d}_{k_m}$ are the start and end key frames of one segment, respectively. For a vertex $\mathbf{p}$ located on the surface of the start key frame $\mathbf{d}_k$, we can utilize the forward tracking results to find its corresponding pixels on the 2D image space across the entire segment. Similarly, for the corresponding vertex $\mathbf{p}'$ on the surface of the end key frame $\mathbf{d}_k$, we can use the backward tracking results to obtain its corresponding pixels on the 2D image space. The inconsistency between the two face scans is therefore measured by the distance of corresponding pixels across the entire segment.

Given forward and backward tracking results obtained from keyframe based tracking, we solve the optimization described in Equation (3.4.2) iteratively with the over-relaxing algorithm. We initialize each refined mesh with the face scans obtained from large-scale mesh registration: $\mathbf{d}'_k = \mathbf{d}_k, k = 1, ..., K$. In each iteration, we sequentially update all the face meshes $\mathbf{d}'_k, k = 1, ..., K$ one by one. For each mesh $\mathbf{d}'_k$, we update vertex positions of $\mathbf{d}'_k$ by fixing all the other meshes and minimizing the inconsistency between the current mesh $\mathbf{d}'_k$ and its neighboring meshes:

$$\min_{\mathbf{d}'_i} \sum_{m=1}^{N_k} G(U_{k_m}, V_{k_m}; \mathbf{d}'_k, \mathbf{d}'_{k_m}), \qquad \mathbf{d}'_k \in Surf(\mathbf{d}_k). \tag{3.12}$$

Due to the topology consistency between the different face scans $\mathbf{d}'_k$, adjusting vertex positions on one mesh changes the correspondences across all the meshes. The algorithm converges quickly. One remaining issue is how to update the mesh $\mathbf{d}'_k$ by minimizing the inconsistency between the current mesh and its neighboring meshes based on forward and backward tracking results.

We update the vertex positions of the current mesh with forward and backward tracking results. For simplicity, we focus our discussion on updating the mesh alignment based on one single segment, including frames $1, ..., L$. For each vertex $\mathbf{v}$ on the current face mesh $\mathbf{d}'_k$ as well as the corresponding vertex $\mathbf{v}'$ on the neighboring mesh $\mathbf{d}'_{k_m}$, we project them onto the 2D image space and obtain the corresponding pixel coordinates $\mathbf{x}_1, ..., \mathbf{x}_L$ and $\mathbf{x}'_1, ..., \mathbf{x}'_L$ and obtain an offset vector $\delta\mathbf{p} = \mathbf{x}' - \mathbf{x}$ on the image space. We project the offset vector $\delta\mathbf{p}$ from the image space back to the mesh surface to obtain the 3D offset vector $\delta\mathbf{v}$. During the projection process, we ensure the updated vertices are located on the original mesh $\mathbf{d}_k$ (*i.e.* $\mathbf{d}'_k \in Surf(\mathbf{d}_k)$). In our experiment, we adopt a weighted combination of 3D offset vectors in order to sum the contributions of offset vectors computed from every frame. We also consider

the visibility of each vertex in computing offset vector. When a vertex or its corresponding vertex is not visible at a particular frame, its corresponding offset vector from that frame is not computed.

We now can update the mesh by moving its vertices $\mathbf{v}$ to the new positions $\mathbf{v} + \delta\mathbf{v}$. To avoid triangle flips, we update the vertex positions by constraining the mesh update step with Laplacian deformation. We again solve a least-squares Laplacian deformation problem. Thus, we generate the final mesh by solving the following quadratic optimization problem:

$$argmin_{\mathbf{v}'}\|\mathbf{v}' - \mathbf{v} - \delta\mathbf{v}\|^2 + \alpha\|L\mathbf{v}' - L\mathbf{v}\|^2, \tag{3.13}$$

where $\mathbf{v}$ and $\mathbf{v}'$ are vertex positions on the current mesh and the refined mesh, respectively. L is the cotangent Laplacian matrix. We experimentally set the weight α to 1.0.

3.5 Facial Performance Reconstruction

We now discuss how to combine image and depth data with registered keyframe shapes to reconstruct facial performances across the entire sequence. Here we focus our discussion on facial performance reconstruction in one segment because we apply the same reconstruction process to each segment. Given the start and end key frames ($\mathbf{d}'_s$ and $\mathbf{d}'_e$) as well as observed data ($O_t, t = 1, ..., L$), our goal herein is to reconstruct facial performances ($\mathbf{m}_t, t = 1, ..., L$) across the entire segment.

The key of our facial performance reconstruction process is to construct dynamic template meshes for facial registration via keyframe interpolations: $\mathbf{s}_t = (1 - w_t)\mathbf{d}'_s + w_t\mathbf{d}'_e$, t=1,...,L. Linear interpolations between the start and end key frames allow us to preserve fine details captured in face scans. Similarly, we apply rigid transformations to the dynamic templates to match observed image and depth data. The transformed

dynamic template at an intermediate frame t is described as follows:

$$\mathbf{z}_t = ((1 - w_t)\mathbf{d}'_s + w_t\mathbf{d}'_e) \oplus \rho_t, \tag{3.14}$$

where $\mathbf{z}_t$ represents the translated and rotated dynamic template at frame t and ρ_t models rigid transformation of the dynamic templates. In our experiment, rigid transformations are estimated by the template-based tracking process described in Section 3.3.1. The next challenge is how to estimate interpolation weights $\mathbf{w}_t$ from observed image and depth data O_t.

We formulate this as an optimization problem by minimizing the inconsistency between the transformed dynamic templates and observed data:

$$\begin{gathered} \tilde{w}_t = \arg\min_{\{w_t\}_{t=1,\ldots,L}} \textstyle\sum_t \mathrm{M}(O_t, \mathbf{z}_t) + \alpha_1 \|w_t - w_{t-1}\|^2, \\ \text{s.t. } w_1 = 1, w_L = 0, \end{gathered} \tag{3.15}$$

where the first term is the *data* term that measures the inconsistency between the facial performances $\mathbf{z}_t$ and observed data O_t. The function D is evaluated in the same way as the objective function used for facial tracking (see Equation (3.4)). The second term is the *smoothness* terms that penalizes sudden changes of the dynamic template meshes between two consecutive frames. We initialize weights by linear interpolations of the start and end frames. We optimize weights via iterative linear solvers, in a similar way as the template-based tracking process described in Section 3.3.1.

After we optimize the weights for each frame, we obtain the dynamic template meshes required for deformation registration: $\mathbf{s}(t) = (1 - \tilde{w}_t)\mathbf{d}'_s + \tilde{w}_t\mathbf{d}'_e$. We can then deform the dynamic template meshes to match observed data at each frame. This requires solve the same optimization problem defined in Equation (3.4), except

Table 3.1: Statistics of our data set.

Subject	# of frames	# of keyframes	# of scans	Mesh resolution
Darren	958	21	12	80K
Muscle	719	97	21	80K
Rock I	820	19	11	80K
Rock II	910	23	14	80K

that we now deform the dynamic template meshes $\mathbf{s}_t$ instead of a constant template mesh $\mathbf{s}_0$ to match observed data. Again, we optimize nonrigid deformations at each frame in the same way as facial tracking described in Section 3.3.1. We denote the estimated deformation as $\tilde{g}_t, t = 1, ..., L$.

Finally, we obtain 3D facial performances of each segment using the estimated interpolation weights and deformation:

$$\tilde{\mathbf{z}}_t = ((1 - \tilde{w}_t)\mathbf{d}'_s + \tilde{w}_t\mathbf{d}'_e) \oplus \tilde{\rho}_t \oplus \tilde{g}_t, \tag{3.16}$$

where rigid transformations $\tilde{\rho}_t$ are obtained from the template-based facial tracking described in Section 3.3.1. Face scans $\mathbf{d}'_s$ and $\mathbf{d}'_e$ are registered in Section 3.4. And interpolation weights $\tilde{w}_t$ are computed by dynamic template estimate process and deformation $\tilde{g}_t$ are constructed by deformation refinement step in this section.

3.6 Experimental Results

We have tested our system on acquiring 3D facial performances of three subjects. Table 3.1 lists the parameters of all the data sets, including the number of frames in each sequence, the number of keyframes used in performance reconstruction, the number of face scans required for 3D reconstruction, and the resolution of the scanned meshes.

We validate our facial data analysis method with one set of facial performance

data captured by [18], which consists of 1388 frames. We first synthesize a sequence of color and depth images based on a setting similar to a Kinect camera. The resolutions of image and depth data, therefore, are 640×480 and 320×240, respectively. We then apply our facial reconstruction process to reconstruct the facial performances across the entire sequence. The result shows that the facial performances reconstructed from low resolution depth and image data capture both large scale deformations and fine-scale facial details in the ground truth facial performances. In Figure 3.8, we also show comparisons of sample frames between the ground truth data and our final reconstruction data, as well as the results obtained from the template-based tracking process described in Section 3.3.1.

Our system can capture realistic dynamic wrinkles and fine-scale facial details. Figure 3.9 shows several sample frames from our reconstruction results. We also show a side-by-side comparison between the captured 3D facial performance and the recorded image data. Our experimental results show that the reconstructed facial performance is consistent with the recorded video data and well retains a lot of spatial facial details captured by 3D scans.

We implemented our system in C++ on a PC with an Intel 3.4GHZ Core i7-2600K CPU and 16GB memory. The template-based tracking code is implemented in CUDA and executed on a GeForce GTX Graphics card. For a typical data set that contains 1000 frames and 12 face meshes with 80K vertices, our system takes about 1.0 hours for facial data analysis, including template based tracking, key shape selection, and keyframe identification, 0.5 hours for registering face scans to the observed data at all the key frames, about 1.0 hours for face scan registrations, and about 1.0 hours for facial performance reconstruction. After the data is acquired, the high-fidelity face geometry for each frame can be rendered in real time.

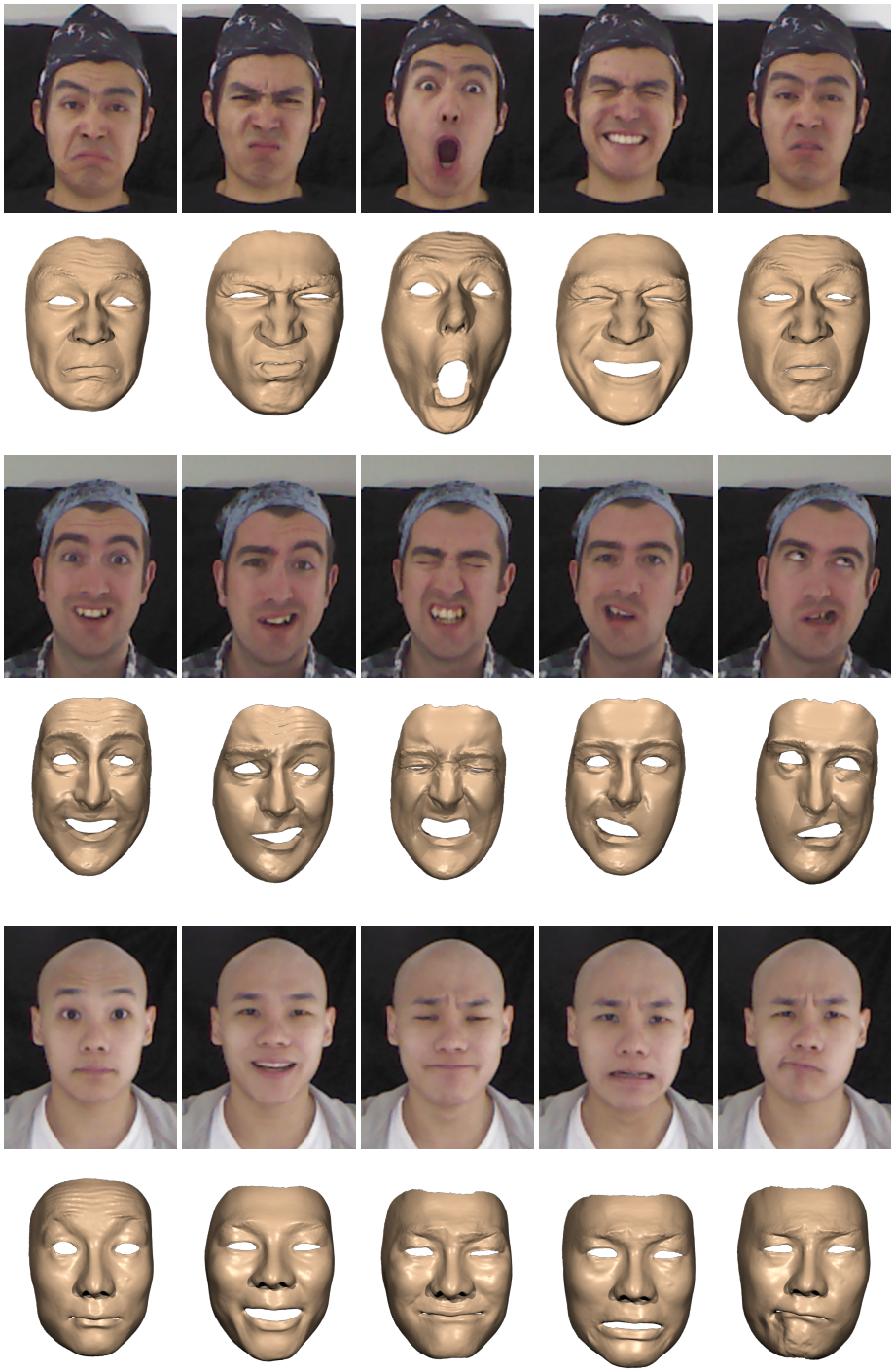


Figure 3.9: Sample frames from our reconstruction results. Rows (1)–(2) show the captured facial performances for Rock. Rows (3)–(4) show the captured facial performance for Darren. Rows (5)–(6) show the captured facial performances for Muscle. For each result, the first row shows the reference images, while the second row shows the reconstruction results.

3.7 Conclusion and Discussion

We present an end-to-end system for acquiring high-fidelity 3D facial performances using a single Kinect camera. The proposed system combines the power of automatic facial tracking and 3D scanning. The quantitative analysis of image and depth data allows us to obtain a minimal set of face scans required for spatial-temporal facial performance reconstruction, thereby minimizing the time and effort for 3D scanning. Our results show that the system can capture high-fidelity 3D facial performances using low-resolution image and depth data obtained from a single Kinect camera.

Static high-resolution face scans and dynamic facial tracking data are complementary to each other as they capture different aspects of the facial performances. In addition, they also benefit from each other. On the one hand, we can utilize facial tracking data to automatically build dense and consistent correspondences between the face scans. On the other hand, the use of multiple face scans for facial tracking can significantly improve the accuracy of the facial tracking process.

The final quality of the reconstructed facial performances highly depends on both spatial resolution of the face scans and quality of observed image and depth data. The current system uses a Minolta VIVID 910 laser scanner to record high-resolution static facial geometry of an actor. We believe the quality of the final results can be further improved with a more accurate and higher resolution 3D scanning system such as XYZ RGB systems [39] or Light Stage [2]. In addition, we expect the quality of the tracking results, as well as the final reconstructed results, can be further improved with a higher resolution depth/color camera.

The quality of the reconstructed facial performances also depends on the accuracy of the face scans registration process because even a small misalignment will result in

unpleasant visual artifacts in the captured facial performance. Our experiments show that the current mesh registration algorithm, which minimizes the misalignments of image and depth data across all the face scans, is effective for retaining fine-scale geometric features obtained from high-resolution face scans. In the future, we will continue to improve the accuracy of our mesh registration process. One possibility is to extract geometric features from static face scans and integrate them into the current registration framework.

While this work focuses on capturing high-fidelity facial performances with realistic dynamic wrinkles and fine-scale facial details, in the future we are interested in modifying the captured facial data for new applications. For example, the captured facial performance data can be interactively edited to generate new facial expressions with direct manipulation interfaces or sketching interfaces [19], interpolated to match low-dimensional signals extracted from vision-based interfaces [12], or retargeted to animate a different human avatar model [22]. Direct applications of previous techniques might not work well for high-fidelity facial performance datasets because they are mainly focused on prerecorded facial data without dynamic wrinkles or fine scale facial details. One of the immediate directions for future work is, therefore, to investigate new methods for editing, retargeting, interpolating, and understanding high-fidelity facial data with 801 dynamic winkles and fine-scale facial details.

Believable facial animation also requires realistic movements of the eyes and lips synchronized with expressive speech. The current system, however, is not suitable for accurately capturing such eye and lip movements synchronized with expressive speech. In the future, we are interested in extending the current system to capture eye behavior and lip synchronization.

4. CONCLUSION AND FUTURE WORK

Capturing quality facial performance for common users by only using a single low-cost capturing device is challenging but will bring broad and potential impacts in many fields. This dissertation presents two systems focusing on different aspects for common users to choose from depending on their needs. The proposed two systems could be used to reconstruct different levels of details of facial performances by using a single Kinect camera only or combining with a 3D scanning system.

In chapter 2, we propose an end-to-end system for acquiring 3D facial performances using a single *Kinect* camera. The proposed system is general for common users for it requires no prior 3D database or statistic models, which is more expensive to retrieve comparing to 2D RGBD priors. The performance capture system combines the power of automatic facial feature detection and image-based nonrigid facial registration, which makes it capable of capturing 3D facial performance automatically, robustly, and accurately. Our results show that the system can capture a variety of 3D facial performances using low-resolution image and depth data obtained from a single *Kinect* camera. The two components actually benefit from each other. We utilize facial feature detector to provide good initialization of the face pose and locations, and moreover use the detected facial features as effective constraints to guide through the registration process. However, facial features component itself only gives us discrete and sparse information of facial movement; while the image registration component reconstruct dense and consistent correspondences of result meshes. Therefore, our reconstructed meshes are suitable of being used in many applications, such as video editing, sequential tracking, and model editing, etc. We also demonstrate that our result is comparable to the commercial Vicon motion capture

system, which uses 12 expensive cameras to track marker positions on the faces.

We further extend our performance capture system for acquiring high-fidelity 3D facial performances. The proposed system combines the power of automatic facial tracking and 3D scanning. The goal here is to incorporate high-resolution 3D facial priors into the system to reconstruct facial details with minimum extract user efforts and also keep the possibility of using a single Kinect camera to reconstruct facial performance (by reusing the face scans). Static high-resolution face scans and dynamic facial tracking data are complementary to each other as they capture different aspects of the facial performances. In addition, they also benefit from each other. We utilize facial tracking data to automatically build dense and consistent correspondences between the face scans. On the other hand, the use of multiple face scans for facial tracking can significantly improve the accuracy of the facial tracking process. The result meshes will fit better to the observed data comparing to that by using a single deformable template model as proposed in Chapter 2. The quantitative analysis of the observed data, as well as the tracking mesh sequence, allow us to obtain a minimal set of face scans required for spatial-temporal facial performance reconstruction, thereby minimizing the time and effort for 3D scanning. The tracking process also provides good reference for registering face scans. Our results show that the system can capture high-fidelity 3D facial performances using low-resolution image and depth data obtained from a single Kinect camera.

The final quality of the reconstructed facial performances highly depends on both the quality of observed data, as well as the spatial resolution of the face scans. We expect the quality of the tracking results, as well as the final reconstructed results, can be further improved with a higher resolution depth/color camera. Moreover, the current system uses a Minolta VIVID 910 laser scanner to record high-resolution static facial geometry of an actor, and we also use the face scan of the subject

under neutral expression as the template. This ensures the reconstructed results encodes personal static facial details, such as pores or scars. To fully get rid of using a 3D scanning system in the performance capture system, one possible solution is constructing an initial template directly from the point cloud that captured from a Kinect camera [36]. A 3D template mesh could be reconstructed directly from a single RGBD camera as well. By recording a sequence of RGBD images by asking the subject to rotate the head in front of a kinect, we could automatically deform a morphable model to fit the recorded RGBD images using an objective function similar to our nonrigid registration step. Even though a template model can be reconstructed directly by using a Kinect camera, we believe the quality of the final results can be further improved with a more accurate and higher resolution 3D scanning system.

The quality of the reconstructed high-detailed facial performances depends on the accuracy of the face scans registration process. A small misalignment will result in visual artifacts in the captured facial performance. Our experiments show that the current mesh registration algorithm, which minimizes the misalignments of image and depth data across all the face scans, is effective for retaining fine-scale geometric features obtained from high-resolution face scans. In the future, we will continue to improve the accuracy of our mesh registration process. One possibility is to extract geometric features from static face scans and integrate them into the current registration framework. Another possible extension of this work is to design an iterative learning and evaluation scheme, which evaluates the best quality of the reconstruction result from the existing scan faces and gives suggestions of next few possible scan faces and estimate the quality improvement to iteratively guide the user to add face scans.

We are also interested in modifying the captured facial data for new applications. For example, the captured facial performance data can be interactively edited

to generate new facial expressions with direct manipulation interfaces or sketching interfaces [19], interpolated to match low-dimensional signals extracted from vision-based interfaces [12], or retargeted to animate a different human avatar model [22]. Direct applications of previous techniques might not work well for high-fidelity facial performance datasets because they are mainly focused on prerecorded facial data without dynamic wrinkles or fine scale facial details. One of the immediate directions for future work is, therefore, to investigate new methods for editing, retargeting, interpolating, and understanding high-fidelity facial data with dynamic winkles and fine-scale facial details.

Our system does not capture the movements of eyes, inner mouth, and hair, though they also play important role in realistic facial performance. The current system, however, is not appropriate to capture accurate movements of those parts. In the future, we are interest in extending the current system to capture eye behavior and do lip movement synchronization with expressive speech.

REFERENCES

[1] Marc Alexa. Differential coordinates for local mesh morphing and deformation. *The Visual Computer*, 19(2):105–114, 2003.

[2] Oleg Alexander, Mike Rogers, William Lambeth, Matt Chiang, and Paul Debevec. The digital emily project: photoreal facial modeling and animation. In *ACM SIGGRAPH 2009 Courses*, page 12. ACM, 2009.

[3] Yali Amit and Donald Geman. Shape quantization and recognition with randomized trees. *Neural computation*, 9(7):1545–1588, 1997.

[4] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[5] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. *ACM Transactions on Graphics (TOG)*, 29(4):40, 2010.

[6] Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W Sumner, and Markus Gross. High-quality passive facial performance capture using anchor frames. In *ACM Transactions on Graphics (TOG)*, volume 30, page 75. ACM, 2011.

[7] Bernd Bickel, Mario Botsch, Roland Angst, Wojciech Matusik, Miguel Otaduy, Hanspeter Pfister, and Markus Gross. Multi-scale capture of facial geometry and motion. In *ACM Transactions on Graphics (TOG)*, volume 26, page 33. ACM, 2007.

[8] Volker Blanz, Curzio Basso, Tomaso Poggio, and Thomas Vetter. Reanimating faces in images and video. In *Computer graphics forum*, volume 22, pages 641–

650. Wiley Online Library, 2003.

[9] George Borshukov, Dan Piponi, Oystein Larsen, J. P. Lewis, and Christina Tempelaar-Lietz. Universal capture: image-based facial animation for "the matrix reloaded". In *ACM SIGGRAPH 2003 Sketches & Applications*, page 1, 2003.

[10] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High resolution passive facial performance capture. *ACM Transactions on Graphics (TOG)*, 29(4):41, 2010.

[11] Jin-xiang Chai, Jing Xiao, and Jessica Hodgins. Vision-based control of 3d facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 193–206. Eurographics Association, 2003.

[12] Jinxiang Chai and Jessica K Hodgins. Constraint-based motion optimization using a statistical dynamic model. *ACM Transactions on Graphics (TOG)*, 26(3):8, 2007.

[13] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(6):681–685, 2001.

[14] Douglas Decarlo and Dimitris Metaxas. Optical flow constraints on deformable models with applications to face tracking. *International Journal of Computer Vision*, 38(2):99–127, 2000.

[15] Irfan Essa, Sumit Basu, Trevor Darrell, and Alex Pentland. Modeling, tracking and interactive animation of faces and heads//using input from video. In *Computer Animation'96. Proceedings*, pages 68–79. IEEE, 1996.

[16] faceshift — face animation software. http://www.faceshift.com/, 2011.

[17] Brian Guenter, Cindy Grimm, Daniel Wood, Henrique Malvar, and Fredric Pighin. Making faces. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 55–66. ACM, 1998.

[18] Haoda Huang, Jinxiang Chai, Xin Tong, and Hsiang-Tao Wu. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. In *ACM Transactions on Graphics (TOG)*, volume 30, page 74. ACM, 2011.

[19] Manfred Lau, Jinxiang Chai, Ying-Qing Xu, and Heung-Yeung Shum. Face poser: Interactive modeling of 3d facial expressions using facial priors. *ACM Transactions on Graphics (TOG)*, 29(1):3, 2009.

[20] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1465–1479, 2006.

[21] Hao Li, Bart Adams, Leonidas J Guibas, and Mark Pauly. Robust single-view geometry and motion reconstruction. *ACM Transactions on Graphics (TOG)*, 28(5):175, 2009.

[22] Hao Li, Thibaut Weise, and Mark Pauly. Example-based facial rigging. *ACM Transactions on Graphics (TOG)*, 29(4):32, 2010.

[23] Wan-Chun Ma, Andrew Jones, Jen-Yuan Chiang, Tim Hawkins, Sune Frederiksen, Pieter Peers, Marko Vukovic, Ming Ouhyoung, and Paul Debevec. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Transactions on Graphics (TOG)*, 27(5):121, 2008.

[24] Iain Matthews and Simon Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135–164, 2004.

[25] Microsoft Kinect Face Tracking SDK. http://msdn.microsoft.com/, 2013.

[26] Motion Capture Systems from Vicon. http://www.vicon.com, 2011.

[27] Frederic I Parke, Keith Waters, and Thomas R Alley. *Computer facial animation*, volume 55. AK Peters Wellesley, 1996.

[28] F Pighin, J Hecker, D Lischinski, R Szeliski, and D Salesin. Synthesizing realistic facial expressions from photographs. In *Proceedings of ACM SIGGRAPH*, pages 75–84. 1998.

[29] F Pighin, R Szeliski, and D Salesin. Resynthesizing facial animation through 3d model-based tracking. In *International Conference on Computer Vision*, pages 143–150. 1999.

[30] O Sorkine, D Cohen-Or, Y Lipman, M Alexa, C Rössl, and H.-P. Seidel. Laplacian surface editing. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 175–184, 2004.

[31] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, August 2004.

[32] Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. *ACM Trans. Graph.*, 26(3):80:1–80:7, 2007.

[33] Robert W Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. Mesh-based inverse kinematics. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 488–495. ACM, 2005.

[34] Levi Valgaerts, Chenglei Wu, Andrés Bruhn, Hans-Peter Seidel, and Christian Theobalt. Lightweight binocular facial performance capture under uncontrolled lighting. *ACM Transactions on Graphics (TOG)*, 31(6):187, 2012.

[35] D Vlasic, M Brand, H Pfister, and J Popović. Face transfer with multilinear models. *ACM Transactions on Graphics*, 24(3):426–433, 2005.

[36] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime performance-based facial animation. *ACM Trans. Graph*, 30(4):77, 2011.

[37] Thibaut Weise, Hao Li, Luc Van Gool, and Mark Pauly. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 7–16. ACM, 2009.

[38] L Williams. Performance driven facial animation. In *Proceedings of ACM SIGGRAPH 1990*. 1990. 24(4):235-242.

[39] XYZ RGB 3D Scanning Systems. http://www.xyzrgb.com/, 2011.

[40] L Zhang, N Snavely, B Curless, and S Seitz. Spacetime faces: high resolution capture for modeling and animation. *ACM Transactions on Graphics*, 23(3):548–558, 2004.