# 博士論文 (要約）

## Adaptive Improvement of Raw-Scanned 3D Models by using Domain Specific Knowledge

## （ドメイン知識を利用した ３次元スキャンモデルの適応的改善）

by

Seung-Tak Noh

盧 承鐸

A Doctor Thesis

博士論文

Submitted to

the Graduate School of the University of Tokyo

on December 6th, 2019

in Partial Fulfillment of the Requirements

for the Degree of Doctor of Information Science and Technology

in Computer Science

Thesis Supervisor: Takeo Igarashi　五十嵐 健夫

Professor of Computer Science

# ABSTRACT

Three-dimensional scanning has increased in popularity to generate 3D models in computer graphics. This technology was originally used by professionals to digitize expensive real-world objects using highly expensive devices, such as a laser scanner. However, recent advances in 3D scanning technology enable a casual user to scan models in our daily lives using a commodity camera. Nonetheless, despite its potential, the scanning method is not widely used in various contexts due to low quality and controllability.

There are three representative problems in a raw-scanned 3D model, from the perspective of quality and controllability. First, geometry artifacts appear in a scanned 3D model such as environmental ground, internal and external noise. In addition, originally separated but adjacent parts become fused. This fused limb parts also hinder from flexible pose changing. Since there is no semantic information in 3D scanning, these artifacts are difficult to be removed automatically. Second, sharp features on the surface become overly smoothed. From the incompleteness of observation and limitation of reconstruction algorithm, a raw-scanned 3D model usually has lower spatial resolution than the source images. Third, additional geometry features such as micro fur strands are also missing. There are several data representations such as a parametric model but conventional 3D scanning methods only cover 3D mesh representation as the final goal. For these reasons, it is quite limited to use a raw-scanned 3D model to other applications such as 3D printing, computer animation, and virtual reality.

In this dissertation, we introduce methods for improving raw-scanned 3D models by leveraging domain knowledge on target applications. The first approach is the skeleton-based shape refinement method for 3D character animation, which fixes the topological issues in the raw-scanned 3D model by leveraging the user-specified skeleton information as the domain knowledge for shape enhancement. Combined with automatic rigging method, it also provides flexibility of pose changing. The second approach is the transfer-based detail enhancement method for the 3D printed human face replica, which is based on the geometric ridges and valleys of human face components. We detect and parameterize faces from a full-body scanned model, and then transfer the geometric features created by experts. The third approach is utilizing the expert's knowledge on reproducing the parametric fur workflow. We find the conceptual similarity between the expert workflow and perceptual feature-based texture synthesis methods, so we establish an optimization framework that utilizes features from the deep convolutional network.

We show the feasibility and effectiveness of each approach, as well as its limitations. We believe our approaches in this dissertation will serve as a foundation for further raw-scanned 3D model improvement.

# 論文要旨

コンピュータグラフィックス分野における３次元モデルの生成法にとして，３次元スキャン技術を用いることが普及されつつある．従来，この技術は専門家がレーザースキャナのような高額な機器を用いて実世界の貴重なオブジェクトをデータ化する際に用いられたものである．それが，近年の３次元スキャニング技術の進歩により，一般ユーザが日常品を普及型のカメラによってスキャンすることも可能な時代になっている．しかし，その可能性にも関わらず，スキャニング技術はその結果得られるモデルのクォリティや制御性が低いことが原因となり，広く活用されていないのが現状である．

スキャンモデルのクォリティや制御性が低さには３つの代表的な問題点が挙げられる．１つ目の問題点として，スキャンデータ上に幾何的アーティファクトが生じることがある．オブジェクトに環境の地面が混ざってしまったり，ノイズによる内外部のアーティファクト等が発生しまったりする．更に，キャラクタモデルのスキャンにおいて，本来別々の手足のパーツが癒着されてしまい，スキャンモデルのポーズ変更の妨げとなっている．これは，３次元スキャン技術に意味論的な情報が反映されてないため生じる問題であり，自動的に削除することが難しい．２つ目の問題点として，シャープな幾何形状が鈍ってしまう問題が挙げられる．これは，スキャンにおいて観測データの不備や再構築アルゴリズムの限界等によるものであり，結果的に幾何形状の解像度は入力画像より劣ってしまう．３つ目の問題点として，毛並みなどのような，表面の詳細な幾何形状が再現されない問題がある．このような形状を再現するには，パラメトリックモデルを利用することが想定されるものの，既存のスキャン技術では3次元メッシュ表現以外の再現は想定されていない．このような問題によって，スキャンで得られた生の３次元モデルは３次元プリント，アニメーション，バーチャルリアリティなどの応用分野であまり広く使われていない．

本論文では，目的とする応用先でのドメイン知識を利用し後処理することでスキャンで得られた生の３次元モデルを改善する，３つのアプローチを提案する．１つ目のアプローチは，３次元キャラクタアニメーションとして応用を想定したものであり，ユーザの指定したスケルトン情報を利用し生スキャンに混ざっているアーティファクトを除外する手法である．また，提案手法を自動リギング手法と連携させることにより，柔軟なポーズ変更を支援することが可能になる．２つ目のアプローチは，３次元プリント等でヒトの複製品を作る際に用いられることを想定したものであり，ヒトの顔パーツにおける凹凸情報を転写する手法である．この手法では，ヒトのスキャンモデルから顔部分を探し出し顔のパーツをパラメータ化する．この処理を専門家によってリタッチされた３次元モデルにかけリタッチ情報を抽出し，異なる生スキャンモデルに適用することでそのリタッチ情報を転写する．３つ目のアプローチは，パラメトリック表現として毛並みを再現する際に用いられることを想定したものである．この手法は，予備実験で発見した，専門家の毛並みの再現作業における考慮事項と，知覚的特徴量によるテクスチャ合成手法とのコンセプトの類似性に基づいたものであり，畳み込みニューラルネットワークから得られる特徴を用いた最適化問題としてのフレームワークで定式化されたものである．

本論文では，提案する各手法の妥当性，有効性並びに各手法の限界をも示す． 提案された各々の手法は，今後，生の３次元スキャンモデルを改善する技術の基盤としての役割を果たすことが期待される．

# Acknowledgements

First and foremost, I would like to present my deepest gratitude to Professor Takeo Igarashi for his continuous support of my Ph.D. study. He provided me an excellent environment and valuable advice on my research. I already owed my master course, but I wonder when I have a chance to repay my debt.

Next, I would like to express gratitude to Professor Nobuyuki Umetani for his helpful comments before defense. Without those comments, I may suffer a considerable difficulty on the summarization and presentation of my work.

I also would like to express my sincere thanks to Dr. Tsukasa Fukusato, who gave me numerous, useful and interesting information. Without his presence, I could not keep my motivation for my research.

Also, I would like to appreciate my Ph.D. dissertation committee members, Prof. Shinpei Kato, Prof. Reiji Suda, Prof. Yoshihide Yoshimoto, Prof. Imari Sato, and Prof. Yutaka Ohtake, who rigourously reviewed my dissertation.

I would like to thank laboratory members who I met within about four years as laboratory staff and Ph.D. student. Especially, I would like to thank Haoran Xie, Chia-Ming Chang, Hironori Yoshida, Hirofumi Seo, Christian Arzate Cruz, Kazutaka Nakashima, who were my office mates for some time. I enjoyed discussions with these smart guys. I also appreciate Hirotaka Suetake, Shunnki Chou, and Kai Kubota. Even though we did not co-work literally, but I believe that we were indirectly and mutually supported each other, by staying and working at the laboratory all night owing to one's own deadline.

Moreover, I also would like to thank my collaborators; Koki Toda and Zheyuan Cai, who worked together for a while. I also would like to thank Kabuku members including Tobias Mohr, Masahiko Adachi, and Kenichi Takahashi who worked two projects together in the remote.

Besides, I thank visitors from abroad, Shaojun Hu, Amal Dev P., Matthew Lakier, Raniero Lara-Garduno, I-Chao Shen, Katja Wolff, Valentin Nigolian, and Riyasat Rateeb. They gave me some intellectual stimulation.

During my Ph.D. student time, I have received financial support from various sources. I was supported by the Graduate School of Information Science and Technology and Foundation for Supporting Internation Students Scholarship, the University of Tokyo. I was also supported by a fellowship from the Japan Society for the Promotion of Science.

Finally, I am deeply grateful to my parents Chi-Soo Noh and Young-Hee Park and my sister Yun-Jea Noh for their constant encouragement and support.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Recently, three-dimensional digitized data (hereafter, 3D data) becomes important gradually due to needs in society. 3D data has been originally used in the engineering field such as computer-aided design and manufacture (CAD/CAM) and cultural heritage documentation. In addition to these conventional needs, there have been increasing the number of applications: 3D animation [110], 3D video games, virtual reality, and 3D printing. Even though all these concepts are not new in the research field, the advances of computation resources and widespread of off-the-shelf devices make possible to democratize these concepts in the general public. According to such needs, the way to create 3D data becomes more diverse.

The ways to create a 3D model were largely divided into two approaches. One is generating 3D models using a 3D tool which is usually designated to create specific constraints, such as polygonal meshing, 3D sculpting (e.g., ZBrush [4]), and sketch-based modeling tool (e.g., Teddy [84]).

Another way is by using 3D scanning technology. This technology has been originally developed for preserving and analyzing a real-world object and scene. Due to this purpose, 3D scanning technologies were originally used by professionals to digitize expensive real-world objects as 3D data using highly expensive devices, such as a laser scanner (e.g., Digital Michael Angelo project [113]). However, recent advances in 3D scanning technology enable casual users to scan a model using standard RGB camera (e.g., PMVS [61] and MVE [56]) or a commodity RGB-D camera (e.g., KinectFusion [88, 127]). Now that 3D scanning is no longer a special way used by professionals, and we can capture objects in daily life. Nonetheless, despite its potential, the scanning method is not widely used in various contexts.

Why is it not used widely? There are various reasons, but the brief and the clear answer would be the deficiencies in the resulting 3D models. The first and representative deficiency is the visual artifact on the scanned 3D model, which is almost inevitable in the scanned 3D model. In the conventional approach, 3D modeling experts need to handle these kind of problem in a manual way. Some artifacts are relatively easy to remove, but most of them need a certain amount of effort. The second deficiency is the lack of detail. This is similar to the conventional accuracy criterion, which evaluates the reproducibility of the real-world object. However, here we think about the adequate detail which is necessary in each domain. The last deficiency is *controllability*. Conventional 3D scanning methods suppose that the scanning scene is static, and only focus on reconstructing the monolithic 3D data. There were relatively small attention on the modification and adjustment of resulting scanned 3D model. Although several recent work, such as [48, 87, 69], has been focused on reconstructing the dynamic object, they are not so widely used due to its instability. Because of these deficiencies, a raw-scanned 3D model usually does not meet the expected level for the application side. To fix this issue, they are usually dependent on manual editing, but it takes a lot of time and human effort by professionals. In this dissertation, we discuss the improvement of the raw-scanned 3D model by domain-specific knowledge.

## 1.1 Taxonomy of 3D Scanning

Before explaining our approaches for 3D scanned model improvement, we roughly review the 3D scanning technology and its context. Taxonomy of 3D scanning is quite challenging, because of its variety in techniques and objects. The most common categorization is dividing by sensing device [17] (Figure 1.1). In this point of view, 3D scanning technology is largely divided into two types. One is contact-based scanning that the device (or medium) touches and measures the

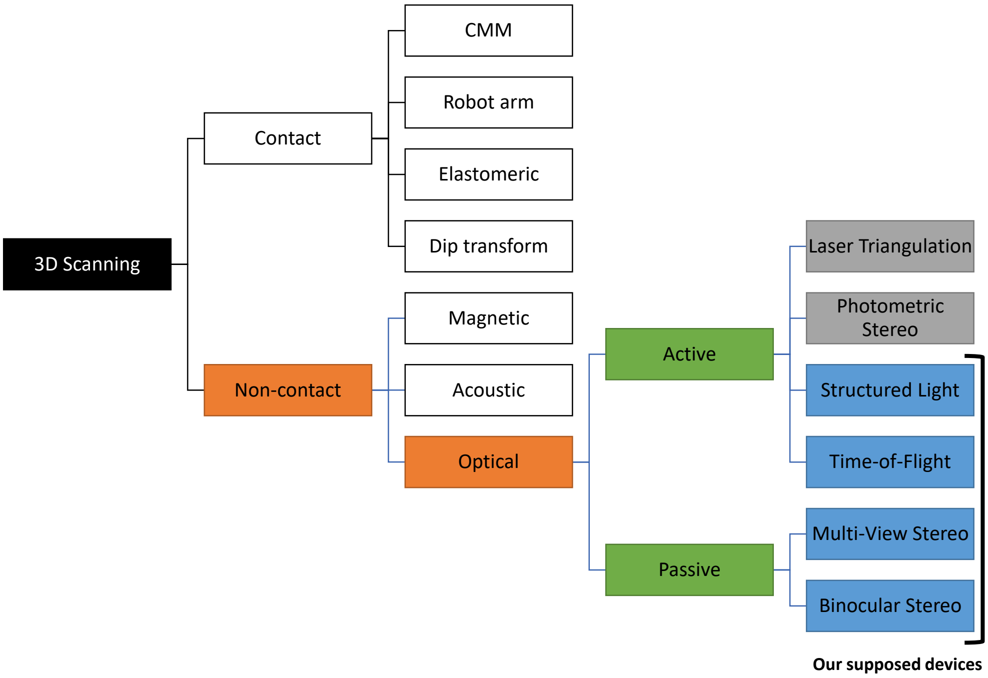


Figure 1.1: Taxonomy of 3D acquisition techniques.

object. The typical example is coordinate measuring machines, robot-arms, and elastomeric sensing [52, 96]. Recently, the dip transform method [5], which is based on Archimedes equality, is proposed. This type of sensing technology is reliable but inevitably accompanies with the cumbersome and tedious capturing process. The major drawback of this technology is that it cannot be utilized to fragile or priceless objects, such as cultural heritage. Besides, the device in these technologies is relatively expensive. For these reasons, this type of 3D scanning is not widely used. The other is non-contact-based scanning, which is divided into types of waves in sensing. Electromagnetic waves, such as computerized tomography (CT) and magnetic resonance imager (MRI), and acoustic waves (e.g., ultrasonic resonance) are usually used for volumetric object acquisition. In this dissertation, we do not cover these types of 3D scanning methods.

The optical-based approach is the most common approach to 3D surface scanning in non-contact-based methods. It is based on optical devices, such as cameras and projectors. The major issue in this type of technique is how to acquire 3D information from 2D images. Optical-based sensing is also divided into two types: active sensing and passive sensing. Active sensing denotes that the camera acquires depth information from a calibrated projector in the same system. Triangulation by 3D laser scanner [147], photometric stereo [184], time-of-flight [104] and structured light [151] sensing are based on this principle. In the case of passive sensing, however, the system finds the pixel relationship among multiple images. Binocular stereo [155] and multi-view stereo (MVS) [158] are classified into this principle. In this dissertation, we mainly suppose that the 3D scanned model was optically acquired, but we do not focus on sensing technology itself. We rather focus on the improvement of the 3D model acquired by 3D scanning. However, we do not consider photometric stereo nor laser scanner, because they are difficult to access by novice users.

In this dissertation, we leverage domain-specific knowledge on 3D data to improve reconstructed 3D models. 3D scanning pipeline consists of several data conversions (Figure 1.2). However, 3D data conversion in the *wild* is *leaky* [90]; each conversion in the scanning process has its limitations, so the final result is poorer than expected. In the case of optical-based 3D scanning, the situation is also similar. The 3D data from the sensor already loose geometric features due to limited image resolutions, occlusions, infeasibility, and noise. This 3D data is also limited to partial view, so the integration and reconstruction are needed; however, this process is necessary to handle redundancy or inconsistency in the data. The easiest way to handle this issue is just smoothing or disposing of inconsistent data; consequently, some of the 3D data is missing in the end. To fix these problems, we need to consider beyond the pipeline; only improving a single conversion in the pipeline does not fix the problem.

Meanwhile, this vulnerability of the scanning pipeline comes from its number of 3D data conventions. If we leverage the domain knowledge, we have a chance to obtain a better reconstruction result than a raw scanned 3D data. This approach naturally assumes a certain type of object, so each method needs to be designed for specific applications. We only show three examples, but the basic idea is applicable to other domains. We believe that our approaches are practical and useful because they do not disturb the original pipeline.

In this dissertation, we only focus on the scanning of a single 3D object in a scene. In otherwords, we do not attempt to reconstruct large-scale 3D scene in our research. Although common 3D scanning technology can be utilized both target domain and 3D scene also has a large amount of applications, it is quite difficult to specify the domain knowledge, because a 3D scene can be seen as a combination of multiple objects. Therefore, we do not cover such 3D scene reconstruction and only focus on a single object in this dissertation.

## 1.2 Background

In this section, we examine representative 3D scanning pipelines (Figure 1.2) and consider the possible *leaks* [90] in these conversions. Although more than hundreds of 3D scanning methods were presented until now, common and popular methods are limited in number; depth fusion [127] and MVS method [61] method. The major issue in these methods is the difficulty in differentiation between the noise and the surface property. Each method settles this problem based on its assumption.

Depth fusion method [88, 127] (Figure 1.2, top) is one of the common approaches for less than a decade. It utilizes a consumer depth sensor [196] as a sensing device. Since the noise level in this type of consumer-level depth sensing device is quite large, it utilizes the voxel-based integration method, called *volumetric range image processing* (VRIP) [42]. This method converts and projects

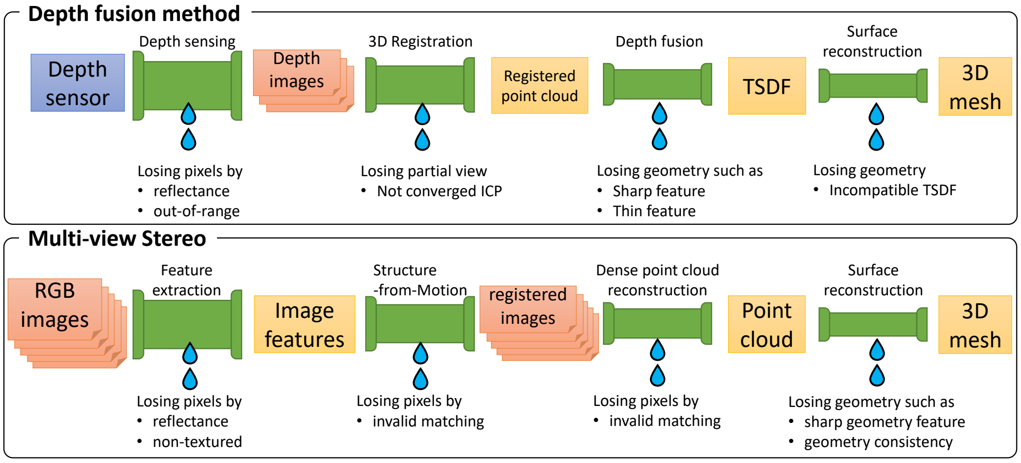


Figure 1.2: Representative pipelines of optical 3D scanning methods.

the input depth image into regular voxel grid as *truncated* signed distance field (TSDF), so it effectively cancels noise in a depth image and reconstructed a smoothed surface 3D model in spite of noisy sensing. However, it inherently has several problems with this design choice, also. Because voxel resolution has restricted as a Nyquist frequency of 3D reconstruction, the reconstructed 3D model has some geometry issues. Thin geometry features, such as sheet-like shapes and small spaces between parts, cannot be reconstructed. These underlying voxels cancel the sharp feature; the reconstructed shape becomes always overly smoothed in general. Besides, *truncated* signed distance also causes some internal artifacts. This approximation keeps the integration process simple, which is helpful for GPU implementation, but it skips to remove the internal artifacts that are created in the early integration stage in a consequence. As a result, the reconstructed result frequently has fused parts and internal artifacts.

The MVS method [60] (Figure 1.2, bottom) is another representative 3D scanning approach nowadays. It tries to find the correspondence pixels among different input images. Naïve pixel matching takes a tremendous computation cost, so they usually utilize keypoint-based matching, such as SIFT [122], SURF [12], and so on. Once found those correspond pixels, it roughly reconstructs the 3D scene by structure-from-motion technique [186]. After that, it finally computes dense reconstruction, such as Patch-based Multi-View Stereo (PMVS) [61] and Multi-View Environment (MVE) [56], to obtain the dense point cloud. Conventional surface reconstruction methods, such as Poisson surface reconstruction [100, 101], can be optionally used to reconstruct 3D mesh. The entire process in MVS method, however, assumes the propitious situation in each step. Most RGB keypoint extraction methods [122, 12] usually suppose the Lambertian surface and cannot handle the surface reflectance (e.g., BxDF). On the contrary, the non-textured surface also suffers from matching problems, neither. For these reasons, corresponded points were usually missing in the point cloud reconstruction. This matching failure influences reconstructed 3D mesh quality, also. Another problem is the imbalanced detail in the reconstructed 3D model. Some parts of the 3D model need more detailed geometry. For instance, facial parts need detail, although it is only part of the entire human body. If the original input images do not cover those areas, the reconstructed 3D model has a lack of detail. Poisson-based surface reconstruction method [100, 101] can handle this kind of non-uniform point cloud distribution. However, it tends to reconstruct 3D mesh in those sparse point area by simply interpolating the shape. For these reasons, MVS methods also suffer from an overly smoothed result, like the depth fusion method.

In summary, recent 3D scanning technologies seem to perform without any issues, but there are several problems as we examined. The problems can be summarized as follows. First, not only the noise but also the partial 3D data (depth image or point cloud) cannot be estimated due to the surface properties. Second, incompleteness in matching or registration algorithm causes an error in 3D reconstruction. Third, the reconstruction methods cannot fully recover the geometry features from acquired primitive 3D data. Conventional approaches focused on solving these problems by improving the accuracy of each step. Recently, there are few attempts to improve the overall scanning process [156, 188], but they are mainly focused on improving the input data for 3D scanning method. In this dissertation, we take a different approach. Neither proposing new scanning process nor improving each step, we rather improve the final reconstructed 3D model based on the domain-specific knowledge. We believe that our approach is practical and useful, because we do not disturb the conventional scanning pipeline and what we need is the reconstructed 3D model in the end.

## 1.3 Our Approach

We introduce the concept of post-process on the raw scanned 3D model by leveraging the domain knowledge (Figure 1.3). Conventional 3D scanning methods only focused on generating the raw 3D model. However, raw scanned 3D models usually do not have enough quality and controllability, so they were difficult to be utilized in other application domains. To overcome these issues, we design the post-process systems that leverage the domain knowledge. Although some previous work, such as [197, 198], were based on the post-process to enhance detail, they usually relied on cues from raw color images in the scanning process. Rather, we are dependent on higher level knowledge, such as the user-specified skeleton, editing by experts, and procedural modeling with perceptual features.

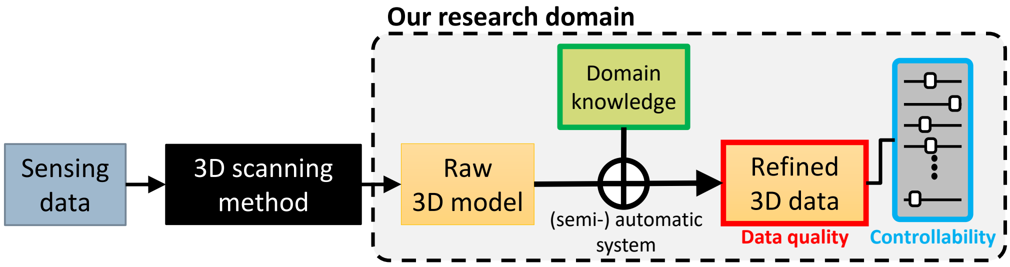


Figure 1.3: Conceptual diagram of our research.

## 1.4 Contribution

In this dissertation, we introduce methods for improving raw-scanned 3D data by leveraging domain knowledge on target application domains. We show three application domains, their domain knowledge, and our proposed methods. Our contributions of this work are summarized as below:

**The concept of 3D data improvement by domain knowledge** We propose new concept for improving raw-scanned 3D data based on the domain knowledge, which is used in a specific application domain. Existing work mainly focused on improving the process module, which is the part of the entire pipeline. In contrast, we mainly focus on the refinement of final reconstructed 3D model by leveraging the domain knowledge. This approach naturally supposes a specific application domain, but we do not have to interfere with existing 3D scanning pipeline.

**Three independent methods for respective application domains** In this dissertation, we introduce three practical methods as examples of our thesis. Each method is not only an example, but also useful standalone system at the same time. Those systems are as follows:

1. We propose a skeleton-based method for fixing several geometry artifacts to animate raw-scanned plush toy models. The key domain knowledge is that the skeleton information can be used to fix the topological issue in the raw-scanned 3D character. We also remove the merged floorplane, internal noise, and topological fused parts by leveraging domain knowledge on the scanning environment and object shape.
2. We propose a transfer-based method for enhancing the geometry detail of the raw-scanned human face in 3D replica. The key domain knowledge for this application is that the editing of the raw-scanned model by experts was done according to the texture information. Based on our finding, we design the system that automatically gathers the retouched geometry features from exemplar 3D model, and transfer those features to raw-scanned target 3D model.
3. We propose a deep feature-based optimization method for estimating the parameters in a procedural fur model. The key knowledge is the experts' behavior that they do not reproduce the reference image pixel-by-pixel manner but reproduce the overall style in a target model. By leveraging this knowledge, we formulate an optimization method to minimize the style feature between the rendered result of parametric fur and reference image.

## 1.5 Organization

This dissertation is organized as follows. We first review the literature and related work of the 3D scanning method in Chapter 2. We examine two representative 3D scanning pipelines and specify the problematic part from them. We also classify each 3D scanning method and its application domains and review the recent advances of them. Next, we introduce three examples of our approach on 3D scanning application domains from Chapter 3 to 5. In each chapter, we first introduce a specific application and previous work on each domain. We then explain the domain knowledge to improve raw-scanned model, and formulate the method that leverages the domain knowledge. Chapter 3 describes our skeleton-based shape refinement method for animating plush toys. Chapter 4 describes our transfer-based method for 3D printed face replica. In Chapter 5, we describe the single-view 3D scanning method that estimates parameters in the procedural model. Finally, we conclude this dissertation with discussions on the limitation of our approaches and further research directions in Chapter 6.

# Chapter 2

# Related Work

In this section, we review the recent advances in 3D scanning methods. As previously described, there are tons of methods already proposed. Therefore, we restrict the scope to the three types of the 3D scanning method: depth fusion method, MVS, and single-view 3D scanning method.

## 2.1 Extensions of Depth Fusion Method

Depth fusion method denotes a seminal work KinectFusion [88, 127] and its variations. This method assumes a specific fixed and static area as a 3D scanning scene, and integrates raw depth image stream from depth camera (e.g., Kinect [196]) to the regular 3D volumetric grid (Figure 2.1). In each frame, it first converts raw depth image to low-level partial 3D representations (Figure 2.1a). These converted 3D data are reconstructed as a volumetric 3D representation (Figure 2.1c). The integration process is done by VRIP [42] that is more robust for depth sensing noise than zippering method [176] that completely relies on the 3D partial views of each frame. After the first frame, it tracks appropriate 6-DoF camera pose by raycasting on reconstructed volume (Figure 2.1d) and iterative closest point method [21] (Figure 2.1b).

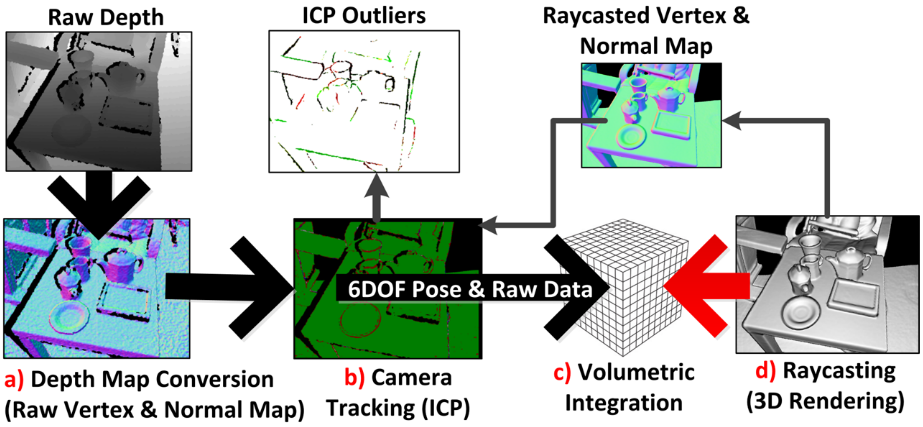

Raw Depth
ICP Outliers
Raycasted Vertex & Normal Map
6DOF Pose & Raw Data
a) Depth Map Conversion (Raw Vertex & Normal Map)
b) Camera Tracking (ICP)
c) Volumetric Integration
d) Raycasting (3D Rendering)

Figure 2.1: Systematic diagram of depth fusion method (by Izadi et al. [88]).

After the publication of KinectFusion, there have been several attempts to extend two major limitations, fixed spatial size and static scenery. The first direction is to ease the spatial limitation in this method. In KinectFusion, it reserves the reconstructed 3D model as TSDF in the regular 3D voxel grid. However, it only utilizes the basic 3D regular grid to preserve TSDF, it needs a huge amount of memory. To overcome these issues, there have been several attempts by using the advanced data structure. Zeng et al. [193] utilized the octree-based data structure to represent the scene-scale. Chen et al. [36] set the active volume to shallow the grid point to be updated for speed-up. Additionally, speed-up on the access time and data compression were achieved by voxel hashing [128]. All methods above were based on the grid structure. Another approach named Kintinuous [182] that converts the outside of active volume to 3D mesh. By this approach, it is not necessary to keep a large size of grid structure for the entire scene. This approach, however, needs a lot of data conversion between 3D mesh and grid structure, so it still needs a huge amount of memory and computation resources. Due to increasing of scene scale from these extensions, conventional motion planning issues, such as loop closure problem, become more important [44, 183]. However, these issues are not so important in the context of 3D object scanning because the 3D volume is relatively small. In this dissertation, we rather focus on fixing the geometry issues in a smaller object than the others by user-specified input.

Another direction attempts to reconstruct a temporally dynamic object from the RGB-D sequence. KinectFusion supposes only static 3D scene, so it tracks and integrates the run-time partial view to reconstructed TSDF volume. To handle the deformation, Newcombe et al. [126] set a warp field to deform the TSDF volume, while Dou et al. [48] utilized the embedded deformation graph model [169], instead of it. Additionally, Innmann et al. [87] utilized SIFT [122] keypoints to track the scene based on the RGB image, not depth image. Recently, Guo et al. [69] estimate not only the dynamic geometry but also lighting and albedo of the captured 3D scene. However, these approaches still relied on VRIP [42] that has geometry issues in the integration. In chapter 3, we descibe the method for fixing the geometry issues from this grid problem.

Aside from these two major extensions, researchers have been explored in different directions. Several works explored different data structures, instead of the TSDF grid; a common approach is to reconstruct dense point cloud [51]. Keller et al. [102] proposed an extension to reconstruct surfel-based representation [139] of the indoor scene. These works, however, usually aim to reconstruct the environment for robot navigation, so they are not suitable for the graphics application. Also, the difficulty in the visibility checking bothers from using them in a graphics application. Recently, Schöps et al. [157] extended surfel-based method to reconstruct mesh together, but the resulting mesh still has a deficiency. 3DLite system [81] took a different approach; it reconstructs the indoor scene as overly simplified planar mesh with a high-quality texture.

Meanwhile, the detail improvement of 3D reconstructed data is an issue since the commodity depth camera released [196]. To improve the 3D quality, there have been several attempts to improve the quality of depthmap by shape-from-shading technique [192, 35, 138, 70]. From the perspective of scanning pipeline (Figure 1.2), these approaches refine the input of scanning method. Although it may gain the quality of single-view sensing data, there is no guarantee that the reconstructed 3D data will be improved due to other leaks. For the purpose of improving the final 3D model, post-processing after the reconstruction (Figure 1.3) is preferable.

Several methods have been proposed to enhance the reconstruction quality by using the color information in offline optimization because a raw-scanned 3D mesh from depth fusion does not have high resolution. Zhou and Koltun [197] utilized color information to reconstruct high-quality vertex colored mesh. Zollhöfer et al. [198] refined raw-scanned mesh by using the shape-from-shading technique. Our work in chapter 4 shares the same motivation, but we enhance the detailed geometry from the transfer-based method. From the perspective of the methodology, Kwon et al. [82] also proposed the transfer-based method to improve 3D data. However, their method improves the sensing 3D depthmap from a commodity depth sensor, while our method improves the reconstructed 3D model. Moreover, their method needs lots of high-quality 3D data and training session, our method only need a limited number of retouched data without training.

There were relatively few attempts on replacing the sensing devices. For instance, the depth fusion method with a monocular RGB camera is also proposed by Pradeep et al. [142]. However, this setting is not widely used yet, because of its instability in depth sensing. We believe that this kind of approach is not so meaningful anymore, because a depth camera is more appropriate device for this type of method, and it has become a commodity hardware (e.g., Project Tango [125]).

## 2.2 Multi-View 3D Scanning Method for Computer Graphics

Multi-view stereo is another representative 3D scanning method (Figure 2.2). This method takes multiple color images of 3D scene as input (Figure 2.2, top left). The method first aligns appropriate camera position of each image by structure-from-motion (Figure 2.2, top right). Once all images are registered in a 3D scene, the method then reconstructs the representation of 3D data (Figure 2.2, bottom right). To obtain the final 3D mesh, modern MVS approaches set a goal of the point cloud representation as a temporary 3D representation. Some of approaches (e.g., [61]) aim to reconstruct global point cloud directly, while others (e.g., [56]) first reconstruct depthmap for each image then project them

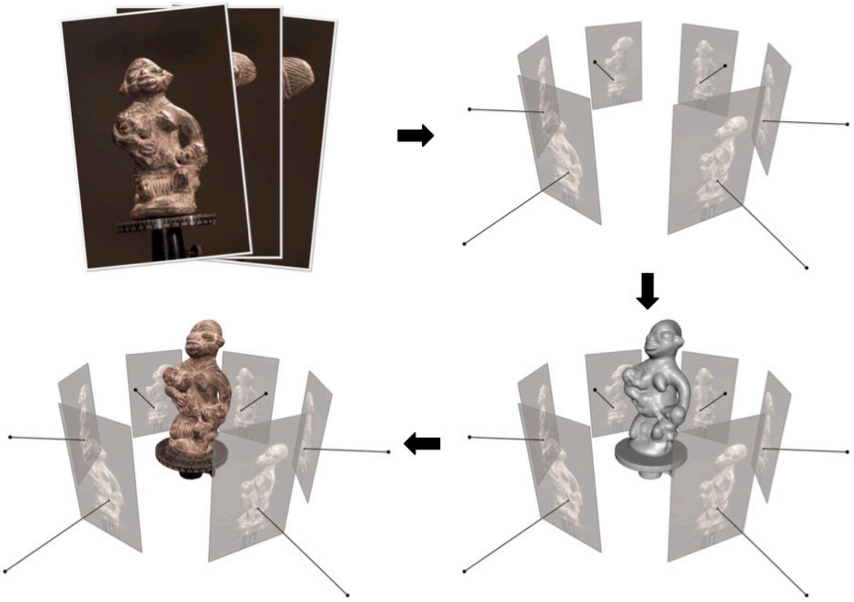


Figure 2.2: Diagram of multi-view stereo method (by Furukawa and Hernández [60])

to create global point cloud. Once raw 3D point cloud is obtained, the system moves to reconstruct the final 3D mesh. Poisson surface reconstruction [100, 101] is commonly used technology for this task. Texture information is also optionally reconstructed in the end (Figure 2.2, bottom left).

Compared to depth fusion, there were not so much theoretical advances in the raw model reconstruction by MVS approach. Rather, most methods mainly focused on and tailored to their application domain. One of the common MVS application is 3D scanning of the outdoor environment [6]. Photo Tourism [165] is the system to explore a city-scale community photograph database based on structure-from-motion. Since it permits the only exploration of photographs based on the sparse point cloud reconstruction, so dense reconstruction was a spontaneous extension of this work. Furukawa et al. [58] applied the MVS technique to reconstruct the dense point cloud of a building-scale scene. The major technology issue in these methods is scalability due to the number of photographs. Because of this, they usually relied on parallel and out-of-core processing.

In the context of multi-view stereo, many researchers had been working on dense 3D point cloud reconstruction. However, images may have a different scale, so the quality of reconstructed 3D mesh may suffer from visual artifacts. To fix this problem, Fuhrmann and Goesele [54] proposed a hierarchical SDF technique to integrate these differently scaled depth maps. This idea was view-dependent, so the same authors extended their idea to surface samples [55]. In this dissertation, we also attempt to improve the reconstructed 3D mesh for this issue, but we rather utilized an exemplar 3D mesh to fix the issue.

The work listed above mainly aimed to be applied to the outdoor environment. Compared to the outdoor scene, there was a relatively small amount of MVS work for an indoor scene [59]. Since there are not enough RGB image features that can be utilized in the architectural scene, some strong assumptions, such as Manhattan assumption [57] and Cuboid assumption [189], are usually needed to reconstruct the 3D scene. A Recent trend of indoor scene modeling is the extraction of more high-level information from the reconstructed 3D scene [86].

The human 3D model is another representative application domain. The quality of the human face model is very important in the movie industry. To achieve the production quality 3D model, MVS is the best choice at present [13]; researchers achieved a high-quality human face model from MVS [14] and texture-mapped model for 3D animation [16, 26]. More recently, researchers have been focusing on each part of the human face: eyes [19], eyelids [20], wrinkles [28], teeth [187] and so on. These works, however, need a specialized setting for face capture. Our work in chapter 4, we utilize the transfer-based method to enhance the detail on a raw-scanned 3D model.

Unlike the human face model, the human hair model reconstruction is a challenging task. The main issue in this approach is the difficulty in the matching strands, due to its reflectance property. Paris et al. [137] integrated 2D orientation fields [136, 180] to reconstruct the 3D orientation field and created hair strands according to the 3D orientation field on the scalp domain. After that, there had been several attempts to capture; mustache and beard [15], some unique hairstyles [79, 123], using simulation [77], and so on. More recently, Zhang et al. [194] reduced the number of views as four by the data-driven approach. In the above work, the reconstructed hair is represented as a bunch of explicit curves. From the user perspective, it is quite difficult to utilize these reconstructed hair models. Because of this, we attempt to reconstruct not the explicit but the parametric representation in chapter 5.

The MVS is also used for 4-dimensional human performance capture [173]. Collet et al. [39] make possible the free-viewpoint video in real-time. Dou et al. [47] set a similar system to that of [39], but they reduced the number of required cameras. By using this technology, Orts-Escolano et al. [134] realized head-mounted display based telepresence in real-time. Pons-Moll et al. [141] extracted the deformation of clothing on the body. Nevertheless, this type of MVS needs a studio-level configuration and highly tuned implementation, so its democratization is difficult. For democratization, single-camera with self-rotating [114, 115, 168, 161] is preferable.

Similar to the depth fusion method, there have been several work on the detail refinement of MVS reconstructed mesh. In general, we need the domain knowledge to refine the raw-scanned 3D mesh. Wu et al. [185] utilized shape-from-shading technique to enhance the mesh detail. Recently, Langguth and colleagues [108] reformulated MVS architecture with considering on shape-from-shading. These approaches are mainly based on the photometric stereo [184], which has ill-posedness between illumination and material reflectance. For proper refinement, we need to know the environment illumination or material reflectance in advance. Several work [149, 154] leverages semantic labels for raw-scanned urban scene refinement. By this approach, it prevents from unpleasing 3D reconstructed result, such as mesh hole in each building or building fused with roadside trees. However, these semantic labeling is usually focused on the large-scale urban scene, and it might be difficult to apply this method to other domain directly.

## 2.3 Single-View 3D Scanning Method for Computer Graphics

Single-View scanning denotes a method that relies on a single image with certain priors. Since there is no 3D information in a single image, we need additional information to reconstruct the 3D model. A common approach is sketching and annotation on the photograph [133] (Figure 2.3, top). If we suppose a certain type of primitive (e.g., generalized cylinder [68]), the tedious user interaction can be reduced; one of the representative work is 3-Sweep system [37] (Figure 2.3, bottom). In this system, the initial two lines define the 2D profile and dragging determines the main axis. This approach is not limited to the static image; for example, sketch on the video to extract the 3D animation sequence of animal behavior [145]. Although the primitive-based approach is a fascinating concept, it is not widely used due to the difficulty on defining the versatile primitives.

The mainstream of the single-view scanning is the data-driven approach. This type of approach is usually combined with the parametric modeling that can be obtained by the interpolation of well-organized 3D models. Human 3D face model is its typical example. The earlier concept can be found as a 3D morphable model [22] (Figure 2.4, top). Cao et al. [29] regress the 3D shape of human

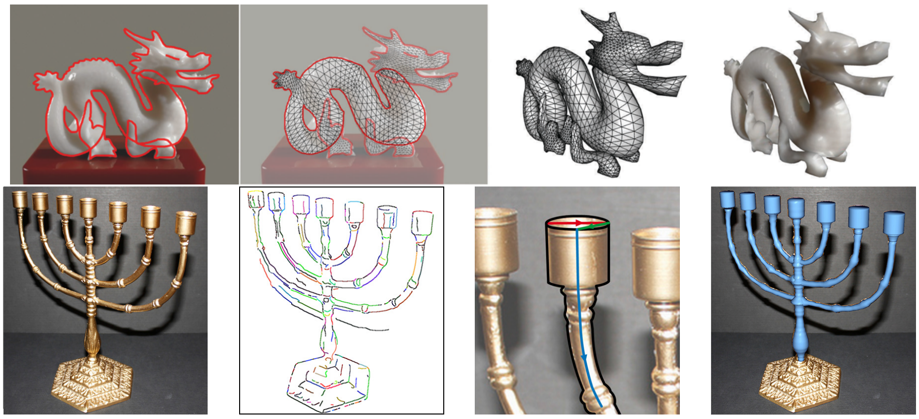


Figure 2.3: Examples of sketch-based single view scanning (by Olsen et al. [133] and Chen et al. [37]).

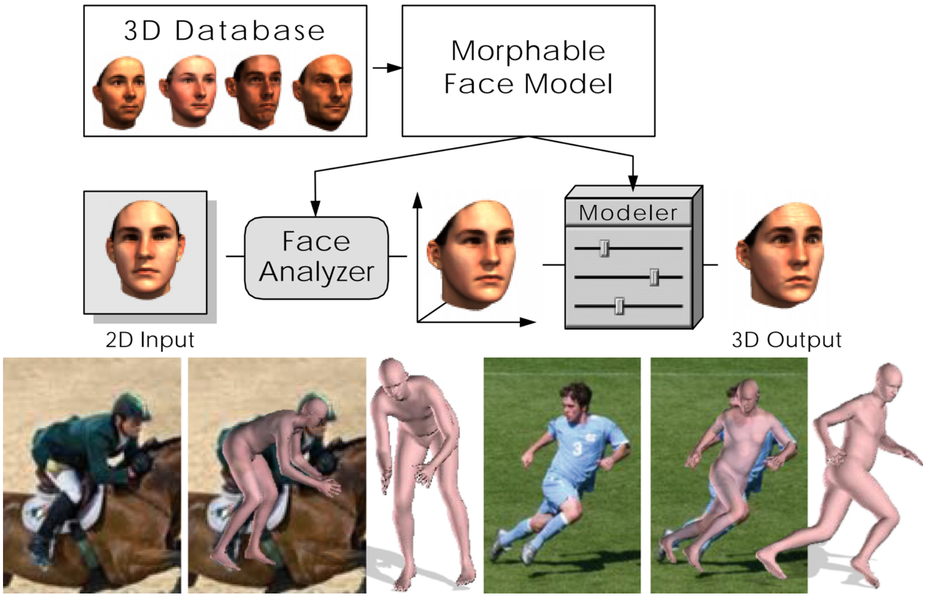


Figure 2.4: Examples of single-view scanning (by Blanz and Vetter [22] and Bogo et al. [24])

3D face models [27] based on the 2D image facial features. Garrido et al. [63] added dense expressions by using a blend shape model. Like MVS, some of work focused on the facial detail, such as lips [65]. More recently, researchers tried to reconstruct 3D rig for animation [64, 83] and image-based textured model [30].

The same concept can be found in the single-view human body scanning. Inspired by 3D morphable face [22], Allen et al. [7] took a similar approach to human body shape. Based on this seminal work, parametric human body model has been extended to cover a wide range of scanned data; SCAPE [10], BlendSCAPE [75], and SMPL model [119]. Combining such a body model and 2D body joint estimation method [140], Bogo and colleagues proposed a single-view scanning method of human body [24] (Figure 2.4, bottom). However, this type of approach is still difficult to be generalized; it needs a lot of well-organized 3D scanned data, such as CAESAR [148] and FAUST [23] dataset. For this reason, this type of approach can be found only for the human face and naked body model.

In the case of the human hair model, both sketch-based approach and data-driven approach are used. The first single-view hair modeling method was based on the user scribbles and image processing to separate hair [33]. After a few years, Chai et al. [31] proposed the method to support a more detailed depth map by the shape-from-shading method. Hu et al. [78] proposed a data-driven approach that can support extrapolating the part of out of frame. More recently, Chai et al. [32] proposed the AutoHair system that does not need user interaction.

Recently, single-view scanning approach has been extended to non-human object. This trend is mainly caused by the advances in deep convolutional networks [105] and the large-scale 3D database (e.g., ShapeNet [34]). Not so long ago, researchers utilized voxel-based 3D representation for this task (e.g., [190]). Although the regular grid is straightforward structure to apply deep convolutional operations in 3D, this representation has several technical issues. The most critical issue is the scalability; we need cubic space to represent the detailed 3D data. Consequently, the resolution is quite limited (i.e., $32^3$). Although Dai et al. [43] extended the output grid resolution to $128^3$ by using synthesis step, the resulting shape was not so pleasing, neither.

Recent progress in this field is going in 3D mesh domain, after Kato and colleagues [99]. They proposed *Neural Renderer* that can handle a 3D mesh as an output of neural networks. Pixel2Mesh system [178] extended Neural renderer to introduce template mesh deformation process and define mesh-related losses, such as mesh normal and Laplacian. Pixel2Mesh++ [181] built upon [178], and added multi-view deformation process to generate desirable mesh shape. Pan et al. [135] added topology modification process to support the shape with complex topology.

Although the work aforementioned were based on the learning of large-scale 3D database. Instead of using 3D database, Kanazawa and colleagues [98] introduced a novel framework to learn the shape variations from annotated information on image collections, such as ImageNet [152]. Their result is impressive, but the mesh quality is still not enough for a casual usage. We believe that 3D scanning method is still needed, when we handle an object which is not in the category of 3D database.

## 2.4 Summary

In this chapter, we reviewed several domain specific scanning methods. Although the various approaches have been proposed in last two decades, we still have a difficulty to obtain a high quality 3D model from 3D scanning methods. The majority of existing work mainly focused on reconstruction of human 3D model, which can be utilized in content production. However, we found that a relatively small amount of work in non-human 3D shapes, such as animals and toys. Since there are not so many 3D data on these type of model, it is also difficult to obtain the parametric model for them. Meanwhile, parametric 3D model is not a silver bullet for the human 3D model; it may not cover some 3D data out of the parametric space. Furthermore, some applications do not need the parametric model. To support these situations, we need a different approach to refine the geometry in the raw-scanned 3D model. Besides, 3D reconstruction does not always mean that the result should be corresponded with pixel-by-pixel manner. In the case of texture-like unstructured pattern, we also need to reconstruct similar-but-different result as output.

In this dissertation, we propose three systems that cover the problems above. To support not-human 3D model, we introduce the skeleton-based shape refinement. Because there is no parametric model to fit this type of model, we rely on the user-specified skeleton information to refine the raw-scanned 3D shape. For human 3D model, we introduce the transfer-based face geometry refinement method. Because our target application does not need the parametric model, we refine the raw-scanned 3D mesh without fitting the parametric model. In the case of using the parametric representation, we attempt to fit the perceptual feature extracted from deep convolution layers, instead of directly using the geometry.

# Chapter 3

# Skeleton-based Shape Refinement in Raw-Scanned 3D Character Model

In this chapter, we describe our skeleton-based method for fixing several geometry issues in a raw-scanned 3D model (Figure 3.1). Our target 3D model is the

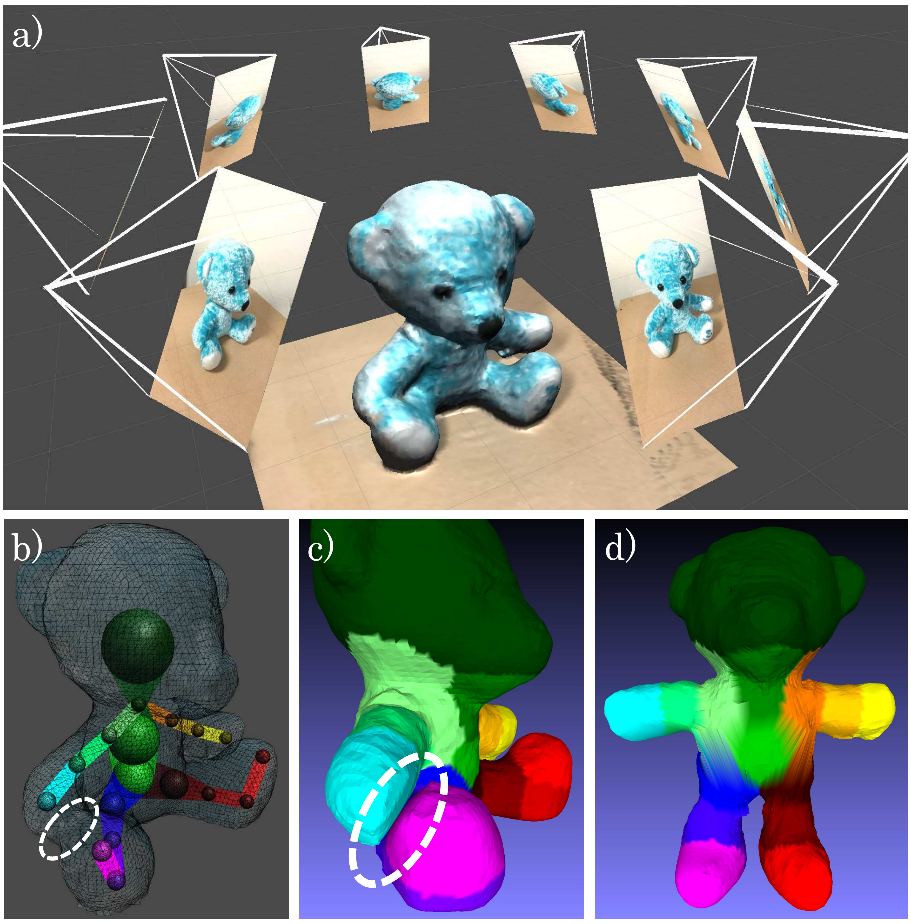


Figure 3.1: Overview of our proposed system for plush toy 3D scanning. (a) Raw-scanned 3D volume with registered photographs. (b) Our tool enables users to annotate 3D skeleton structures in the raw-scanned 3D volume. (c) Our skeleton-based segmentation and shape refinement method enables the geometric issues in the raw-scanned 3D volume to be cleaned. (d) The automatic skinning computation generates the animation-ready 3D skinned mesh.

plush toy model, and we aim to animate them. However, raw-scanned 3D model have several issues, such as (1) environment are merged together, (2) internal artifacts, (3) topological fused limb parts, and (4) shape artifacts after detachment. We leverage several domain specific knowledge to solve these issues. First, we detach object from raw-scanned 3D volume by leveraging the prior on the floor. Next, we separate fused limb parts in a raw-scanned volume by leveraging the skeleton information provided by the user. We then refine the processed shape by leveraging the fact that plush toys have rounded shapes. The method in this chapter is mainly based on the work presented in Graphics Interface 2019 [132].

## 3.1 Scope of Application

Recent advances in 3D scanning technology enable casual users to scan a model using a commodity RGB-D camera [88, 127]. Given the wide availability of 3D scanning, it is now possible to capture objects in daily life, in order to create 3D character models. However, raw-scanned 3D models present several issues for their use in animation.

We suppose the casual scanning scenario for character animation; there is an object on the plane, such as a desk or table, and no additional object in the scanning area. Even though this scenario is quite propitious situation, the raw scanned 3D model still required an intensive editing. These include the necessity to remove the ground plane, holes generated by the invisible areas created during scanning sessions, and fusion of nearby parts of the model. After that, it is also necessary to manipulate skeleton and assign appropriate bone weights, which require tedious manual operations.

Here we think differently on the process above. If we leverage the skeleton information for animation and other domain knowledge, we can simplify the steps above. We remove the ground plane from raw scanned 3D scene based on the predefined plane equation. In addition, we fill the hole in 3D model which is created by the planar removement. User-specified skeleton is not only working as a rigging skeleton but also working as a shape prior that can be used for removing fused parts on the raw scanned 3D model. Overall, the manual operations by user can be considerably reduced, compared to the conventional approach by means of complex 3D software.

We propose a semi-automatic method for converting a raw-scanned 3D model to an animation-ready model with simple annotations and domain specific knowledge (Figure 3.1). The system requires a raw-scanned 3D volume and some registered photographs as input. Subsequently, the user specifies a predefined skeleton structure and annotates it onto the provided registered photographs. The system then segments the raw-scanned volume based on the skeleton and generates a cleaned 3D mesh, after which it automatically computes the bone weights of each vertex and generates a 3D rigged mesh. This approach simplifies operations associated with both geometry cleaning and rigging.

Although several methods clean geometry and generate animation-ready models, we combined these concepts into a single system to democratize scanning for 3D animation. In this method, we clean the ground and fused portions of the scanned object by using a small amount of annotation on the raw-scanned 3D volume. Although most automatic rigging methods assume clean 3D models [45, 46], we deal with raw-scanned 3D data with uncleaned visual artifacts.

## 3.2 Existing Approaches

Since the release of the commodity RGB-D camera [196] and technical advances in scanning methods [48, 69, 88, 127], 3D scanning has become a popular method for generating 3D models. However, despite its potential, the scanning method is not widely used across fields. To democratize this technology, we need to address common problems in the raw-scanned 3D objects.

### 3.2.1 Segmentation of Scanned 3D Data

One major problem with scanning technology is object segmentation. Because there is no semantic process in typical scanning methods [88, 127], the target object and surrounding scene are usually not separated. There are several ways to segment an object from the scene, but most of them completed in an automatic way. One representative approach is plane-based segmentation [76], where the main plane(s) is estimated in the view, and each object is segmented from the plane. This also has several limitations, resulting in application of deep-learning methods to segment the 3D point cloud (e.g., PointNet [143]). Compared with the automatic approach, user-assisted 3D data segmentation has not been widely investigated. SemanticPaint [177] supports user-assisted segmentation but only supports large-scale object segmentation from a room-scale 3D scene.

In this study, we assumed a small scanning volume containing a single object on a flat surface and only considered removing the flat surface from the raw-scanned volume. Although this does not represent a contribution to the field, it is a reasonable assumption, because additional objects usually disrupt the scanning process (i.e., partial scanning by unseen areas), and it consequently enforces reduced scanning resolution for each object.

### 3.2.2 Skeleton-based 3D Modeling and Refinement

Another important factor for democratization of scanning methods is shape refinement. There are several issues with a raw-scanned 3D shape, and many users rely on conventional 3D modeling tools mainly designed for modeling from scratch. A skeleton is a common structure used to segment and refine raw-scanned

3D data, with a previous survey [170] offering a comprehensive introduction to this approach.

In skeleton-based shape processing, an issue remains in how to generate skeletons. Most previous methods [80, 146] depend on automatic skeleton extraction; however, because it is difficult to achieve a completely automatic approach, user-defined parameters are usually required. The Morfit system [191] supports generalized cylinder fitting [37, 68] on point cloud data to complement imperfect 3D scanning. The concept of using the skeleton for geometric improvements to raw-scanned data is similar to ours; however, our primary focus is on separation of an existing shape.

A 3D skeleton with bone thickness can be used in the context of 3D modeling. The B-Mesh system [93] utilizes a user-specified 3D skeleton with key-balls that generate an initial mesh by sweeping and stitching balls in a skeleton and subdividing them to obtain a higher-resolution mesh. Sphere-Meshes [174] uses a similar concept to that of a sphere-based 3D skeleton but adopts simplices among spheres to create a final shape. This approach allows representation of a complex shape with a small number of primitive shapes. However, these methods focus on simplifying well-defined 3D meshes and do not address raw-scanned 3D data.

In this study, we focus on *separating fused parts* in the raw-scanned 3D volume by using a user-specified 3D skeleton originally used to rig the 3D model to separate fused parts in the raw-scanned 3D volume. To the best of our knowledge, this represents the first attempt to simultaneously address issues associated with separating fused parts and rigging the limbs in a raw-scanned 3D model.

### 3.2.3 Rigging for Mesh Animation

Mesh skinning is a common approach for 3D animation, but the cost of manual rigging prevents application by a non-skilled user. This has resulted in mesh rigging being an important research topic for decades. The most direct and representative method for this activity involves a fully automated approach, such as the Pinocchio system [11]. Recently, Dionne and de Lasa [45] applied voxel-based discretization to automatically and robustly rig production-quality but non-manifold mesh; however, this approach is basically limited to a clean 3D model that has clearly separated limbs and in a rest pose (i.e., a T- or A-pose).

Recent work focused on supporting manual deformation using a novel optimization technique. Jacobson and colleagues [91, 92] introduced methods allowing the deformation of a two-dimensional (2D) or 3D mesh according to several user-specified control points by energy minimization. This method allows for intuitive mesh deformation for novices, but requires an optimization-based deformation framework, which is not widely supported in off-the-shelf graphics engines. Additionally, such methods require an additional round of discretization, such as tetrahedral meshing [163] that supposes a cleaned 3D surface mesh as

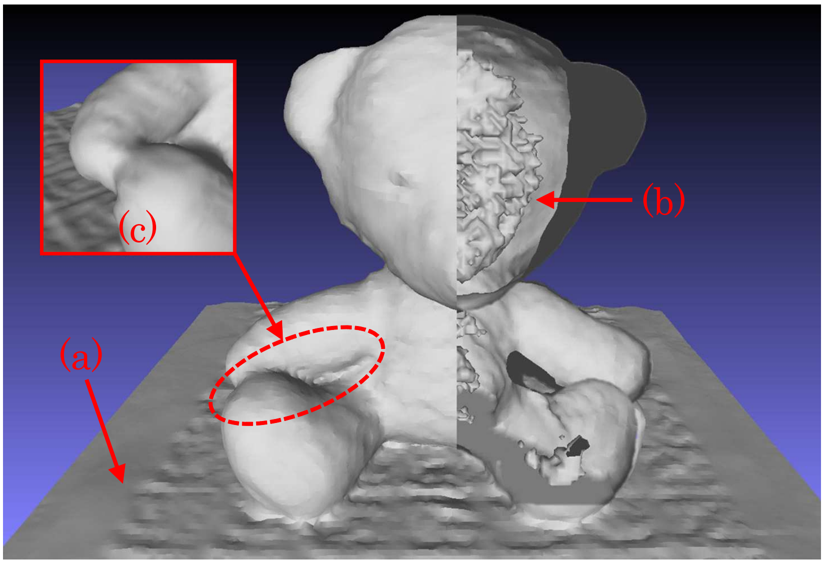


Figure 3.2: Problematic shapes in a raw-scanned 3D model. (a) Ground plane. (b) Internal artifact. (c) Fused parts.

an input. For these reasons, we continued to use bone-based skinning for mesh deformation and attempted to solve shape issues in original raw-scanned 3D voxels. To reduce such effort in mesh skinning, an alternative method generates a rigged mesh from scratch. Borosan et al. [25] proposed the RigMesh system, which enables users to create the skinned mesh using a generalized cylinder as a primitive [37, 68]. Recently, Jin et al. [94] proposed AniMesh, which enables users to animate 3D models from the RigMesh system [25] according to human motion. However, these approaches share the same shape limitations inherited from geometric primitives. Although their work inspired the present study, our goal was to use raw-scanned 3D data.

## 3.3 Problem Formulation

### 3.3.1 Motivation

Before introducing the system, we illustrate the problematic areas in a raw-scanned 3D model captured with a commodity RGB-D camera (Figure 3.2) and address the three types of problems in this work: ground plane, internal artifacts, and fused parts.

**Ground plane** We assume that the target object is placed on a flat ground plane (Figure 3.2a). Such a ground plane is included in the raw-scanned volume, making it necessary to remove it before approaching it as a 3D object.

**Internal artifacts** Voxel-based scanning methods, such as KinectFusion [88, 127], integrate each depth frame into a 3D regular grid and construct a *truncated* signed distance field. This effectively cancels the depth noise in the measurement, although the integration usually causes artifacts inside of the 3D model when using raw TSDF values without filtering (Figure 3.2b).

**Fused parts** Different limb parts, such as an arm and leg, are fused together in the volumetric scan when they touch each other (Figure 3.2c), thereby making it necessary to separate them.

These represent common problems in raw-scanned 3D models and for which there is no general solution. Conventional 3D sculpting software, such as Zbrush [4], can fix these issues; however, such tools and operations are difficult and time-consuming for novice users. We addressed this by allowing the user to provide simple annotations.

### 3.3.2 Our Approach

In this chapter, we utilize the skeleton information as *prior* to fixing the issues in a raw-scanned 3D model. Skeleton-based 3D animation is quite common in computer graphics venue. However, for animating an object, we usually follow the workflow with an effort by experts: (1) fix the geometry issues in a raw-scanned 3D model, (2) attach 3D skeleton, and (3) change the rigging weights for natural 3D animation. Here, we find some redundancy among these processes. If we utilize the skeleton information to fix geometry, which is originally attached *after* the 3D shape, how the entire workflow can be improved? To examine this idea, we design the system and conduct several experiments.

## 3.4 User Interface

### 3.4.1 Workflow

The system comprises three parts: RGB-D based scanning, user-annotation, and geometry processing (Figure 3.1). We built our scanning platform using an Intel RealSense SR300 near-range depth camera and the KinectFusion algorithm [88,

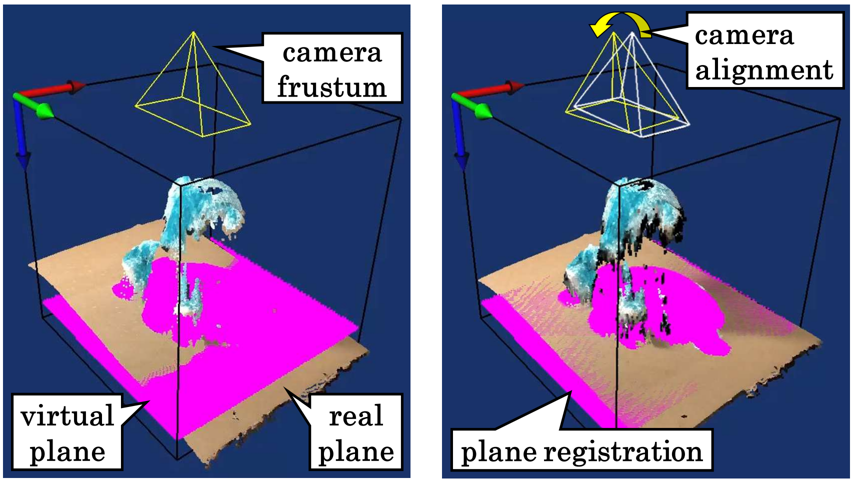


Figure 3.3: Registration of a raw-scanned 3D volume according to a predefined virtual plane (left). The user aligns the virtual plane (magenta) to the real one. The ICP algorithm aligns the camera pose to the scanning volume (right).

127] for depth integration. We assumed that the target object is placed on a flat plane that is preferably rotatable. In our preliminary study [132], we relied on fiducial markers for camera tracking [150]; however, our current prototype works without these markers in order to support casual scanning situations and uses a predefined virtual plane in the scanning volume (Figure 3.3). The user roughly orients the object at the center of view and aligns the virtual plane to the real one. The scanning method is then activated and the iterative closest point (ICP) [21] algorithm is used to fit the virtual and real planes. This simple modification removes the necessity for the fiducial markers required by the previous method [132]. Additionally, this platform captures user-specified photographs from the RGB camera.

Following acquisition of the registered multi-view photographs on the raw-scanned volume, the system removes the ground plane based on the predefined plane information as a preprocessing step. Following removal of the ground, the user annotates the skeleton structure onto the registered photographs. Because a novice user might have difficulty with 3D rotation [68], we did not include this operation in the system but rather provided registered multi-view photographs in order to allow users to perform 2D-based operations for the annotations. The number of photographs and its distribution are not limited, but we assumed less than 20 oblique views for the user interface.

### 3.4.2 Skeleton-annotation Tool

We implemented a simple skeleton-annotation tool to allow the user to provide essential information necessary to clean and rig the raw-scanned 3D volume (Figure 3.4). First, the user chooses the skeleton structure from humanoid and quadruped models (Figure 3.6) that are compatible with a common motion database [2]. In the 2D-annotation step (Figure 3.4, top), the system requests the user to specify skeletons in two views in order to obtain a 3D skeleton. After annotations in two-views are completed, the 3D position of each node is computed with epipolar geometry [73].

Once the initial 3D points are computed, the system enters the adjustment step (Figure 3.4, bottom), during which the user can check and adjust the position and size of each 3D node using simple drag-and-drop operations. More advanced technologies exist, such as inferring a 3D skeleton from multi-view 2D skeletons [106]; however, our raw-scanned 3D volume may contain fused parts, which normally make skeleton extraction difficult. We believe that editing a predefined 3D skeleton with ~20 bones will not be that tedious for a user; therefore, the system is completely reliant on a 3D skeleton specified by the user.

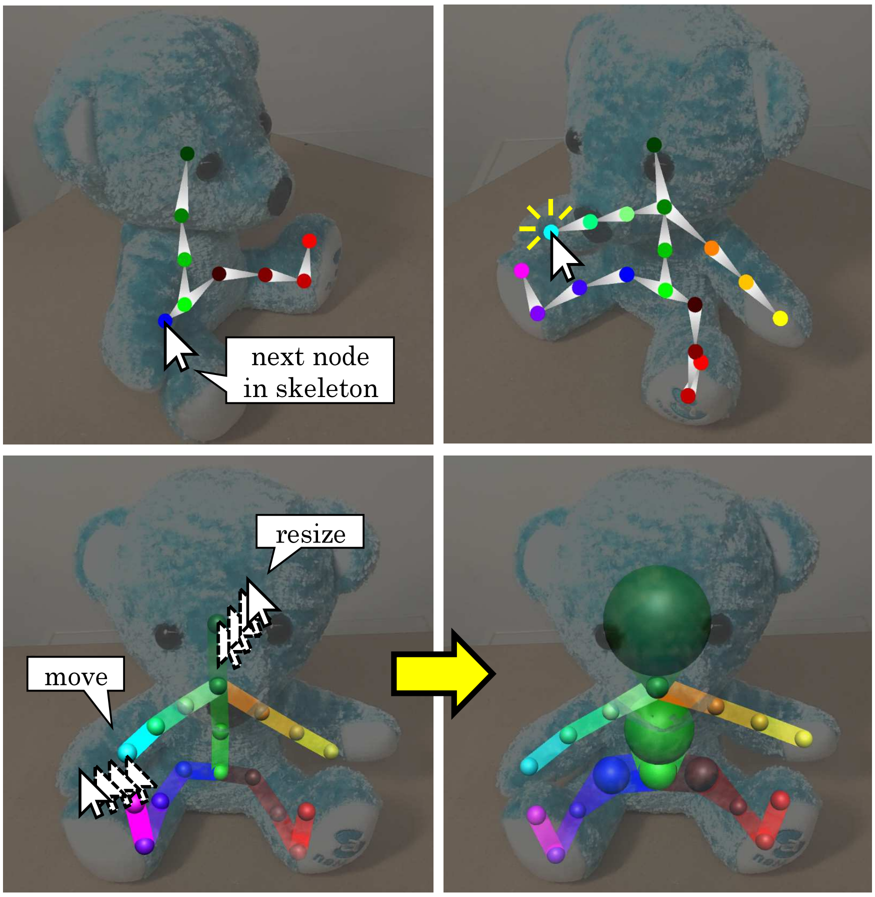
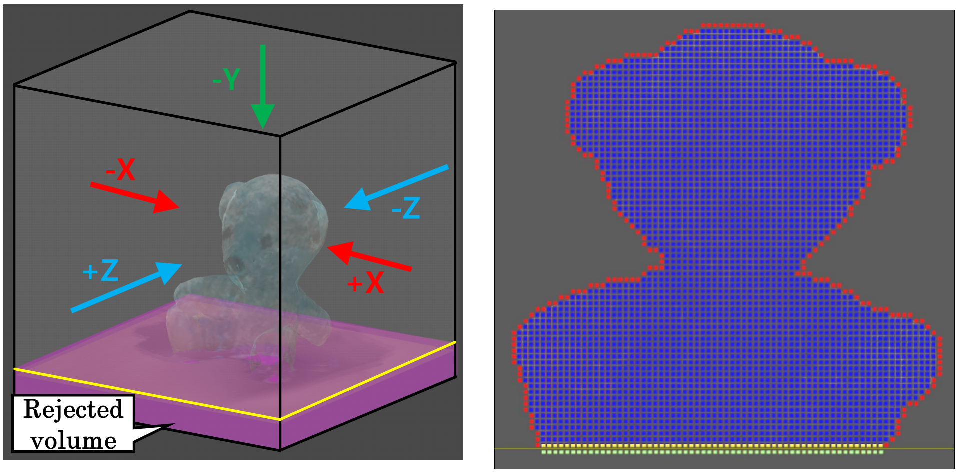


Figure 3.4: Annotation tool for the skeleton specification. The user can insert the 2D projected points from 3D node into each reference view (top). Based on the 2D points in multiple views, the system computes the initial 3D nodes (bottom), at which time the user can check and adjust the position and size of the 3D nodes.

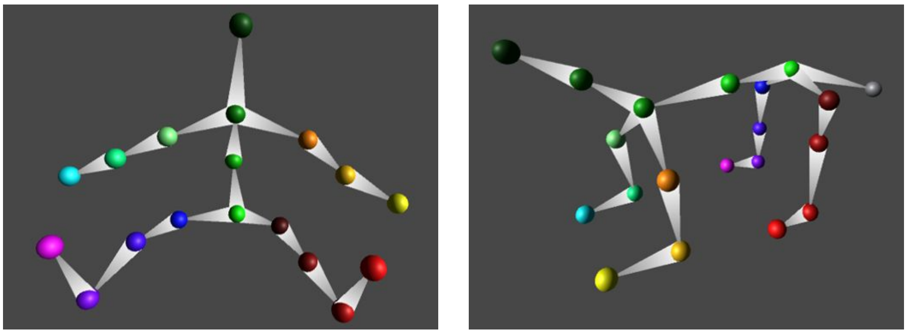
## 3.5 Algorithm

Based on the user-specified skeleton provided using the annotation tool, the system generates a cleaned mesh from the 3D volume and computes its skinning. Our geometry processing was mainly inspired by the method of Dionne and de Lasa [45, 46], which first voxelizes the 3D mesh of the character before computing the vertex-to-bone distances. Because the scanned 3D data used in our method are also contained in 3D voxel space, we adopted their main workflow; however, the artifacts present in raw-scanned 3D data prevented the direct use of this method. Therefore, we added steps to clean the those artifacts in the raw-scanned 3D volume in four parts: removal of the ground plane and internal artifacts from the volume, separation of fused parts, shape refinement, and mesh skinning.



(a) (b)

Figure 3.5: Removal of the ground plane and internal artifacts. (a) Removal of the predefined ground plane. (b) The visibility test in five directions allows the acquisition of the inside (blue) and exterior (red) voxels. Based on the predefined ground plane, the hole at the bottom is filled (green).



(a) (b)

Figure 3.6: Predefined skeleton set in our prototype. (a) Humanoid model. (b) Quadruped model.

### 3.5.1 Removal of the Ground Plane and Internal Artifacts

We first separate the ground plane from the raw-scanned 3D volume. Due to the lack of semantics, the ground plane is mixed with the object in the raw-scanned volume. Based on the predefined ground plane, we remove all of the voxels under this plane (Figure 3.5a).

We then remove internal artifacts from the raw-scanned volume by modifying the visibility checking described by Dionne and de Lasa [45, 46]. Figure 3.5 illustrates the filtering process, which checks the visibility of each voxel from five orthogonal-view directions (we did not consider the bottom-to-up direction where the plane exists). The internal and exterior voxels can be determined based on their visibilities. After determining voxel visibility (the visible voxels are exterior, and invisible voxels are interior), we remove all of the internal voxels.

We then fill the hole at the bottom of the object caused by removal of the ground plane (Figure 3.5b) and evaluate the voxel situated one layer above the rejected volume, which is specified by the ground plane equation. If it is internal, we attach the exterior under these voxels. This approach allows acquisition of a water-tight 3D volume without the internal artifacts.

### 3.5.2 Separation of Fused Parts

To leverage the user-specified skeleton in order to separate fused parts (Figure 3.2c), we first compute the distance fields from each skeleton bone to each internal voxel [45] and associate each voxel with the nearest bone. For each pair of adjacent voxels, we assess the precomputed graph geodesic in the skeleton (Figure 3.6) between the associated bones. To compute this geodesic, we consider each bone as a graph vertex and each node (bone connection) as a graph edge. If the graph geodesic is shorter than the threshold (we used 3), they remain connected; however, if the geodesic is longer, we remove these two voxels by setting their TSDF values as outside of volume. The resulting mesh maintains a water-tight configuration regardless of whether voxels are removed, because the raw-scanned volume contains truncated signed distances. We then obtain the surface mesh by applying the iso-surface extraction method [53, 121]. The bone associated with each vertex is visualized according to color to allow easy identification of the relationship between skeleton and vertex (Figure 3.7).

In our preliminary study [132], we applied the fast-marching method (FMM) [159] instead of the Dijkstra algorithm due to differences in voxel resolution. We directly employed the raw-scanned 3D volume for the voxelization; therefore, our resolution was coarser than that of Dionne and de Lasa [45]. In this situation, the distance field calculated using the Dijkstra method suffers from artifacts from Manhattan distance associated with part separation and mesh skinning (Figure 3.8). This issue was addressed by substituting the computational method with a continuous metric, although one major drawback of using the FMM is the computational cost. In this study, we replaced the FMM with the heat method [40, 41], which has linear time complexity. Additionally, this method requires some precomputational cost to build and decompose matrices, after which the runtime computation is significantly faster than the FMM. This is useful, because the system needs to repeatedly update the distance field during the adjustment step.

Compared with standard human models, our target models (plush toys) have characteristic shapes (e.g., a large head). Therefore, the naïve distance computation does not work well without considering the bone volume (Figures in 3.1 and 3.2). To address this issue, we allowed the user to specify the node size in the annotation tool. We used *a medial cone* [116] for the bone shape, which is a convex hull of two spheres potentially having different radii. Consequently, the bone shape was quite similar to that reported previously [9], although it is simpler than the shape in [9].

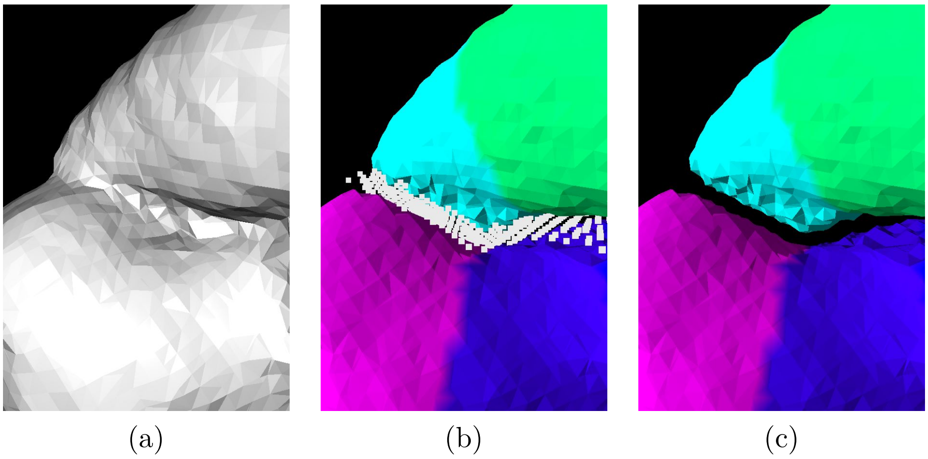


Figure 3.7: Method for separating fused parts. Colors are associated with bone. (a) Mesh from the raw-scanned volume. Near parts are improperly fused together. (b) Identification of wrong connections in the voxel domain (white dots denote voxel points). (c) Mesh after separation.

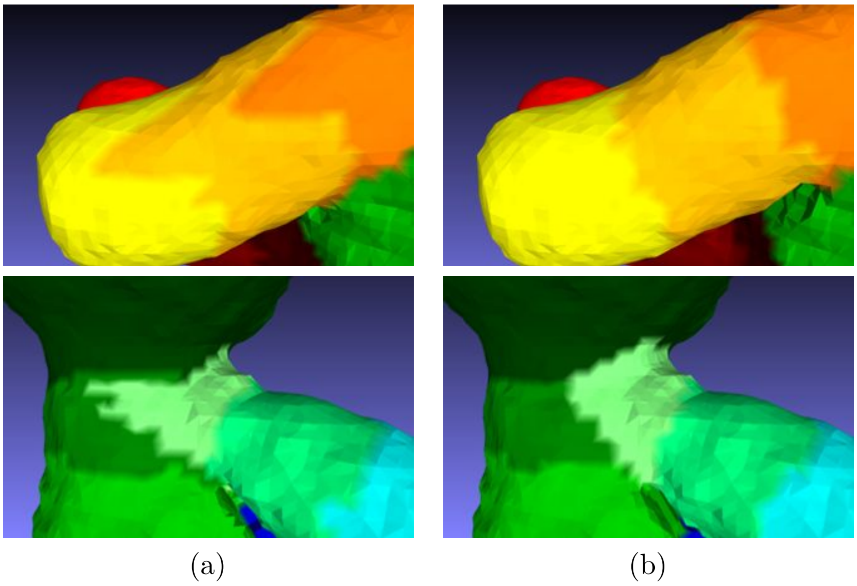


Figure 3.8: The effect of different metrics. (a) The Dijkstra-based metric causes discrete artifacts in the final shape due to the Manhattan distance. (b) The FMM or heat method generates a relatively smoother result due to a continuous metric, despite the coarse voxel resolution.

Here we describe how to determine the voxel point **p** is interior of the bone shape. Since we have two spheres with different radii in the medial cone, the main issue is basically computing the center position and radius at the cut surface (trapezoid in Figure 3.9). We compute the angle $\alpha$ by:

$$\sin\alpha = \left(\frac{\mathrm{r}_2 - \mathrm{r}_1}{|\mathbf{c}_2 - \mathbf{c}_1|}\right) \tag{3.1}$$

Note that a numerator becomes a negative value when $\mathrm{r}_1 > \mathrm{r}_2$, but it still works. By using this value, we can compute the center of a cut surface $\tilde{\mathbf{c}}_i$ and its radius

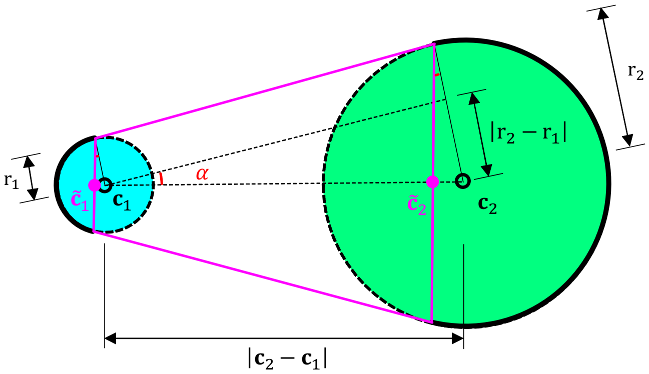


Figure 3.9: Our bone shape as a convex hull of two spheres.

$\tilde{\mathrm{r}}_i$, respectively:

$$\tilde{\mathbf{c}}_i = \mathbf{c}_i - \hat{\mathbf{v}} \cdot \mathrm{r}_i \sin\alpha \tag{3.2}$$

$$\tilde{\mathrm{r}}_i = \mathrm{r}_i \cos\alpha \tag{3.3}$$

where $\hat{\mathbf{v}}$ denotes the normalized directional vector ${}^{\mathbf{c}_2 - \mathbf{c}_1}/_{|\mathbf{c}_2 - \mathbf{c}_1|}$. Based on the relationship above, we compute Algorithm 1 to determine whether the voxel point is inside of this convex hull.

**Algorithm 1** Determine a point **p** in a bone

```
if $|\mathbf{p} - \mathbf{c}_i| < \mathrm{r}_i$ where $i = 1, 2$ then          ▷ 1) point to spheres
    return inside
else
    $\tilde{\mathbf{v}} = \tilde{\mathbf{c}}_2 - \tilde{\mathbf{c}}_1$, $\mathbf{w} = \mathbf{p} - \tilde{\mathbf{c}}_1$
    $t_1 = |\tilde{\mathbf{v}} \cdot \mathbf{w}|$, $t_2 = |\tilde{\mathbf{v}} \cdot \tilde{\mathbf{v}}|$
    if $t_1 < 0.0$ or $t_2 < t_1$ then          ▷ 2) check ortho projection
        return outside
    else
        $t = t_1/t_2$, $\tilde{\mathbf{p}} = \tilde{\mathbf{c}}_1 + t \cdot \tilde{\mathbf{v}}$
        dist $= |\mathbf{p} - \tilde{\mathbf{p}}|$
        size $= (1.0 - t) \cdot \tilde{\mathrm{r}}_1 + t \cdot \tilde{\mathrm{r}}_2$
        if dist $\leq$ size then          ▷ 3) distance to line segment
            return inside
return outside
```

After determining the voxels included in the bone, we assigned them as the initial seeds with 0-distance and initiated the heat method [40, 41] in order to compute the distance field. Although the heat method accelerated the runtime speed, computing the entire distance field required considerable time. Therefore, we only updated the distance fields of modified bones in the recomputation step. Additionally, we interactively visualized the intermediate results following each distance-field computation.

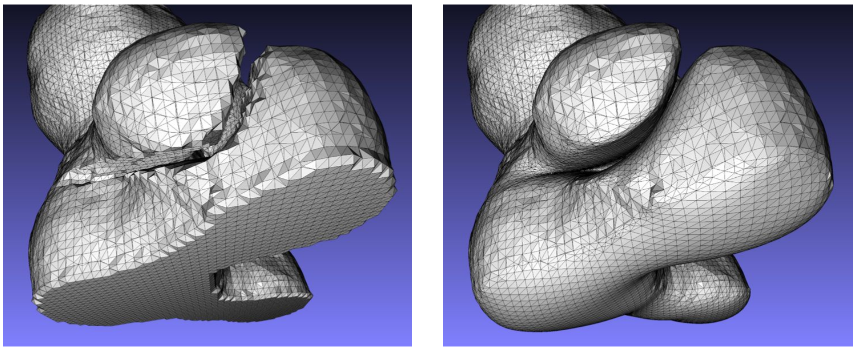


Figure 3.10: Voxel disconnection usually generates visual artifacts in the final mesh (left). Separated parts have a staircase-like shape, and ground parts become a flat surface. Laplacian-based shape refinement mitigates visual artifacts (right).

### 3.5.3 Shape Refinement

We obtained cleaned voxels with following removal of the ground plane and separation of the fused body parts. However, the extracted mesh from this volume still contained visual artifacts around the modified area (Figure 3.10). In our previous work [132], we did not refine these visual artifacts, whereas in the present study, we refined them by deforming the mesh around those regions using Laplacian-based optimization [166]. This process is quite similar to mesh hole filling, but we did not add new vertices, but rather modified the original vertices, because our mesh was already water-tight. We tested the cotangent Laplacian in our preliminary experiment based on it being the common way to compute a Laplacian in a mesh; however, this did not work well in our experiments, whereas a graph Laplacian returned satisfactory results and was subsequently used in the current implementation.

### 3.5.4 Mesh Skinning

For the final step, we computed the skinning weights for the 3D mesh by applying the automatic skinning computation used by Dionne and de Lasa [45, 46] for vertex weights. Although we previously acquired the cleaned 3D mesh and distance fields associated with the bones, those distances were originally computed *before* disconnecting the voxels (described in subsection 3.5.2). Because of this, the distance fields continued to be incorrect. A naïve approach would involve recomputing all of the distance fields; however, this would require building and solving linear systems from scratch, because the precomputed matrices were unsatisfactory due to the disconnected voxels. This would be a time-consuming process undesirable for the interactive system.

We consider the necessary part of the distance fields in the voxel area (Figure 3.11). For mesh skinning to animate the 3D model, the necessary distances are originated from nearby bones. Although distances from far bones are largely affected by voxel disconnection, they are unnecessary for skinning. As a result,

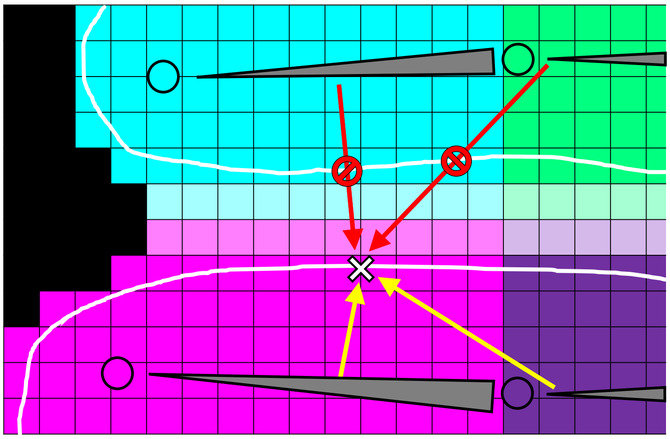


Figure 3.11: Illustration of inconsistent distance values. Although voxels in fused areas were eliminated (white-toned), incorrect distances remained (red arrows). To avoid recomputation of the distance fields, we bound only associated and adjacent bones to the vertex (yellow arrows).

it was unnecessary to recompute the distance fields, and we instead bound bones (computed in 3.5.2) with adjacent connected bones. From a practical perspective, this was reasonable, because skinning in common game engines (e.g., Unity3D [3]) limits the maximum number of bones to four for each vertex.

## 3.6 Experiment Results

### 3.6.1 Performance

We tested our method using 10 plush toys, all of which were scanned at $128^3$ voxel resolution with each axis 25 [cm] in length. Eight were human-like biped models, although the proportions differed, and the remaining two were horse-like quadruped models. Figures 3.1 and 3.2 show our skeleton annotations, part separations, and skinning results.

For measurement of method performance, we used a desktop computer with an Intel Core i7-8700 CPU and 32GB of RAM. Table 3.3 shows our experimental results. The runtime performance of the heat method was significantly better than the FMM, although the heat method involved a costly precomputation step. If we include this precomputation step into the total amount of time required by the heat method, the result would be similar to that for FMM. However, this precomputation is performed only once and prior to user interaction; therefore, it is inconsequential from the perspective of the user. In casual usage, two to five bones can be recomputed simultaneously in 1 to 2 seconds.

### 3.6.2 Animation

We compared our animation results with those of the Pinocchio system [11]; however, simply applying the raw-scanned 3D model was unable to generate a meaningful result, because the Pinocchio method does not include a mesh cleaning

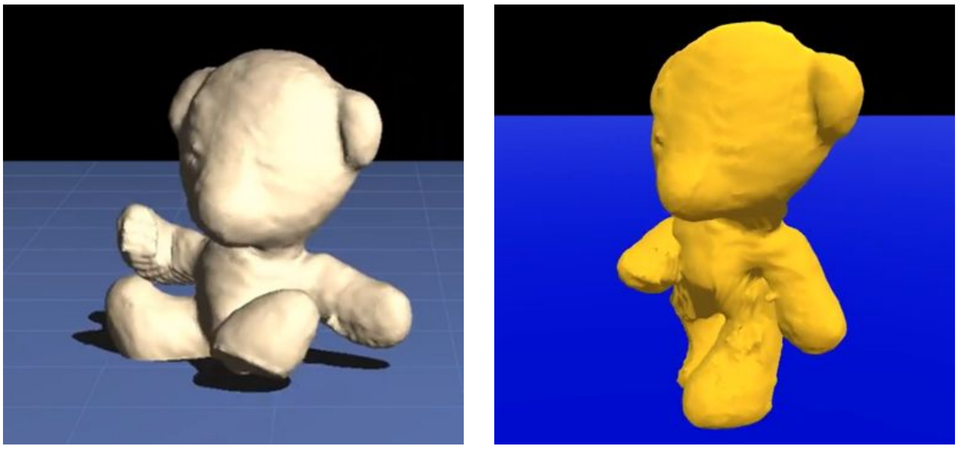


Figure 3.12: The screenshot of animation by the Pinocchio [11] system and ours. Because Pinocchio assumes an A- or T-pose at the initial status, false rigging occurred (left). In contrast, we can support the arbitrary initial pose from the user-specified skeleton (right).

process. Therefore, we used the cleaned mesh from our separation method and switched performance of mesh skinning to the Pinocchio method.

Figure 3.12 shows a screenshot of the mesh animation generated by each method. The Pinocchio method initially assumed the A- or T-pose, resulting in incorrect skinned results whereas this was not an issue for the proposed method, given its ability to support a non-standard initial pose from the user-specified skeleton. This demonstrated that the proposed method was able to cover a wider range of 3D character models than the Pinocchio method.

### 3.6.3 Pilot Study

We conducted a pilot user study in order to evaluate our initial annotation prototype using the FMM distance computation and the part-separation algorithm with the skeleton [132]. Although the upgrades implemented in the present study improved areas of the previous method submitted to a user study, we used the same details as the previous user study because the main workflow did not need to change and has many implications for further research directions.We scanned the 3D models shown in Table 3.1 in advance and captured equally-distributed side views (Figure 3.1a) for each 3D model. After scanning, we invited five users with knowledge of 3D computer graphics. One was an expert in the field, and the remaining four had intermediate knowledge of 3D computer graphics. The session took 10 to 15 minutes per model, and all participants succeeded in generating similar positioning of a skeleton to that shown in Table 3.1. However, the size of the nodes varied for each participant, resulting in variance in the partitioning results.

In post-study interviews, the participants noted that they did not have difficulty understanding the concept or the manual operation of our annotation

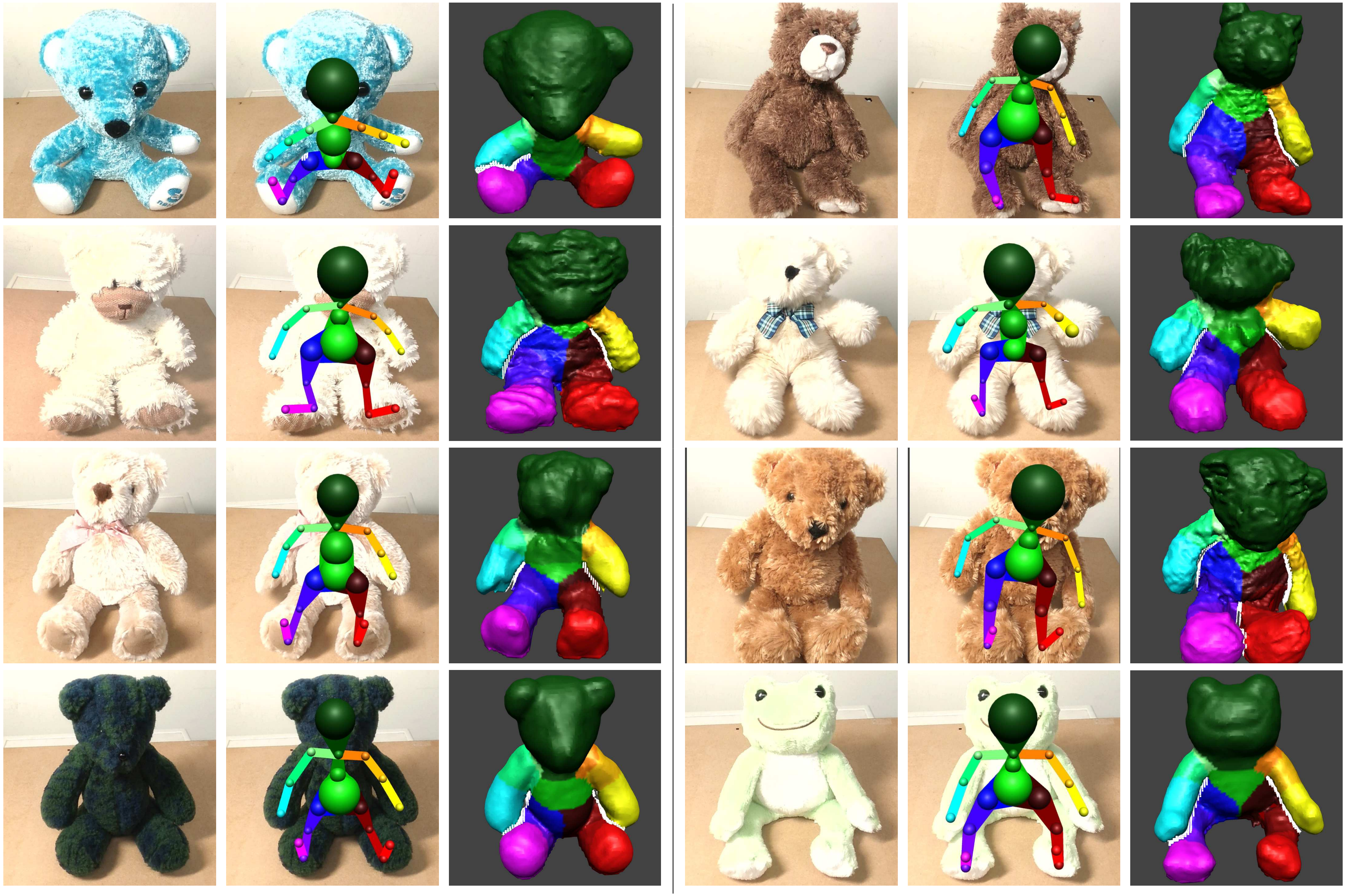


Table 3.1: The results of a humanoid shape. (left) Frontal image (middle) User-specified skeleton (right) Mesh and removed voxels.

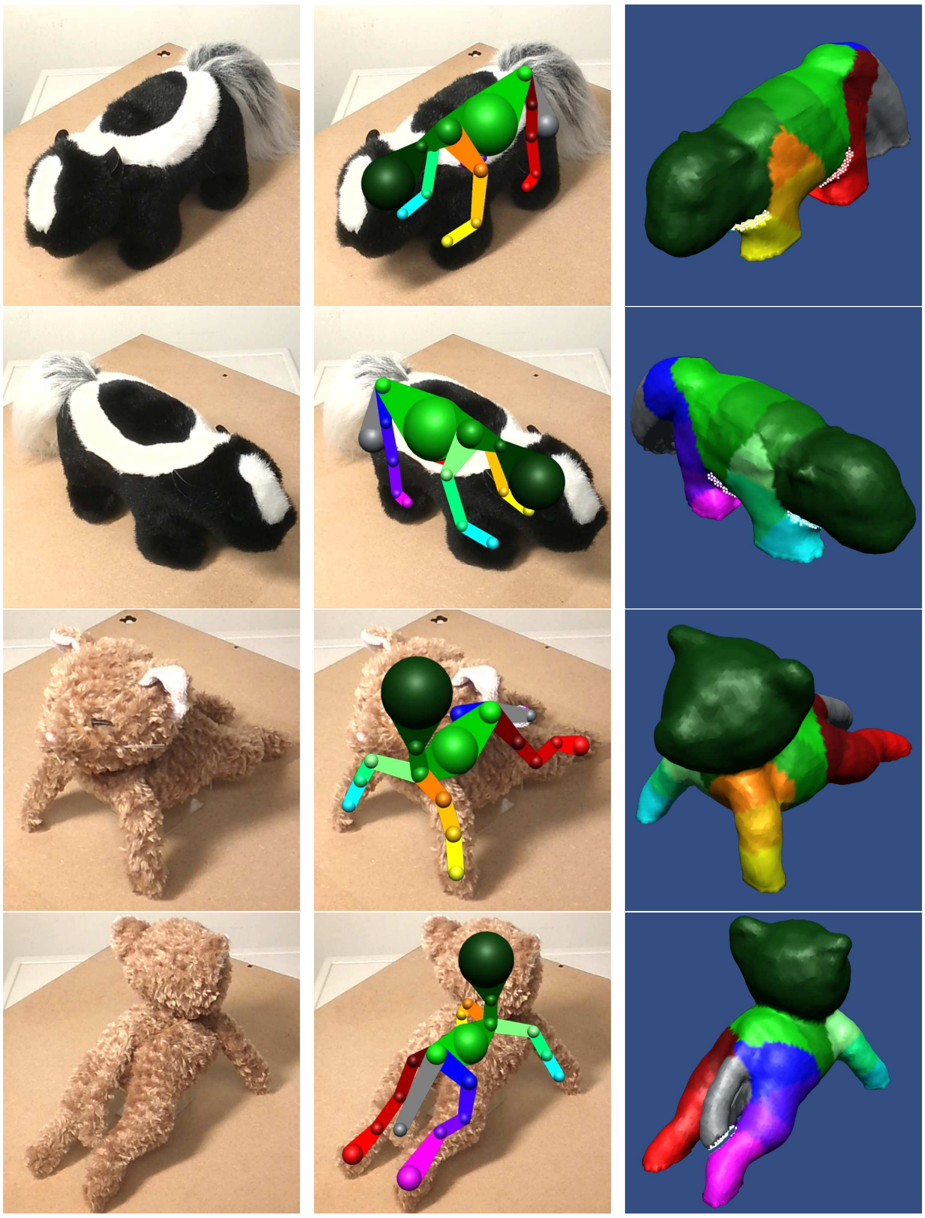


Table 3.2: The results of quadruped shape. We provide two different views for each model. Columns are consistent with Table 3.1.

system. Two participants highlighted the uniqueness of the 2D-based interface and wanted to use a 3D-based interface, and admitted to rapidly adapting to use of the system. However, all participants indicated the difficulty in anticipating the actual effect of node size on the final mesh. This required several adjustments of the node size in order to recompute the distance fields. However, they did not complain about wait times associated with the distance computations by the FMM, but implied that it would be obtrusive if they needed to work on the task for a long period. Additionally, two of the participants indicated that they wanted to know the effect of their operations on the animated 3D mesh, which is not included in the current implementation.

Table 3.3: Our experiment results. Timing is seconds.

| Model type (#bones) | Model name | #valid voxels in a raw-scan | Computing entire distances | | Voxel discon. | #verts in final mesh | Shape refinement | Mesh skinning |
|---|---|---|---|---|---|---|---|---|
| | | | FMM | heat (+precomp.) | | | | |
| Humanoid(17) | Bear_blue | 88,845 | 12.806 | 3.956 ( +5.684) | 4.075 | 20,350 | 0.979 | 2.687 |
| ” | Bear_brown | 159,945 | 22.106 | 5.877 (+14.558) | 4.518 | 34,597 | 1.863 | 7.154 |
| ” | Bear_white1 | 151,097 | 20.239 | 5.644 (+14.393) | 4.469 | 32,510 | 1.662 | 6.376 |
| ” | Bear_white2 | 156,822 | 21.173 | 5.837 (+17.388) | 4.741 | 32,639 | 2.432 | 6.361 |
| ” | Bear_beige | 125,067 | 16.996 | 4.863 ( +9.400) | 4.280 | 27,755 | 1.497 | 4.500 |
| ” | Bear_orange | 223,497 | 29.620 | 7.898 (+32.876) | 5.041 | 45,776 | 2.472 | 12.647 |
| ” | Bear_green | 101,718 | 14.313 | 4.316 ( +6.852) | 4.069 | 22,968 | 1.205 | 3.422 |
| ” | Frog | 73,520 | 10.978 | 3.570 ( +4.184) | 3.886 | 19,082 | 1.129 | 2.470 |
| Quadruped(19) | Skunk | 66,117 | 11.167 | 3.764 ( +4.139) | 3.934 | 16,475 | 0.947 | 1.776 |
| ” | Cat | 58,511 | 10.346 | 3.421 ( +3.136) | 3.865 | 15,983 | 0.752 | 1.643 |

## 3.7 Discussions

### 3.7.1 Limitations

Our method successfully cleaned and created a skinned 3D mesh from a raw-scanned 3D volume with simple manual annotations. However, there remain several issues that to be addressed.

One significant limitation is the mesh quality in the animation. Although this method refined the mesh, visual artifacts remain obvious in not only the cleaned parts but also elsewhere. This is due to various reasons, but the main reason is the low mesh resolution. Specifically, this phenomenon frequently occurs at the boundary of different bone associations (differently colored vertices in a triangle). Mesh subdivision at these areas or remeshing will likely help address these artifacts. Moreover, combining these mesh operations with the user interface will be necessary to generate a more natural animation. Additionally, either converting to a rest pose (i.e., a T- or A-pose) before remeshing or allowing users to annotate a higher level representation for remeshing and smoothing [124] will help generate good tessellation.

We utilized a graph Laplacian [166] for shape refinement (subsection 3.5.3), because we failed to optimize the linear system associated with the commonly used cotangent Laplacian. It is likely that several slivers generated by MarchingCubes [53, 121] cause a level of numerical instability in a cotangent Laplacian matrix. This fragility also implies that the mesh from a raw-scanned volume needs to be handled more carefully than a well-crafted manual mesh.

### 3.7.2 Future Work

In this study, we only evaluated the system using a small number of participants; therefore, a more intensive study is necessary to evaluate this annotation method. Additionally, we need to improve the prototype system in order to provide a smoother user experience. The main bottleneck of our previous prototype [132] was the lack of rapid user feedback. This was partially solved by adopting the heat method [40, 41] for distance computation; however, this was still insufficient to support seamless integration with a mesh-animation preview, as indicated by the participants in our pilot study. The obstacle was the costly precomputation step used by the heat method.

There are several ways to extend our method in the future. One interesting direction involves linking voxel disconnection with the components of Cholesky decomposed $\mathbf{LL}^\top$ matrices for the heat method [40, 41]. In the current skinning computation, we only bind the associated bone with adjacent bones to avoid recomputing all of the distance fields; however, two matrices in the heat method contain only voxel connectivity information, with no additional row or column used for voxel disconnections. This could be addressed by replacement of a row vector with a one-hot vector on a diagonal in the original matrices. Therefore, if we directly modify the $\mathbf{LL}^\top$ matrices to achieve this, it would be unnecessary to recreate the matrices, thereby allowing use of the heat method without the precomputation step, which would be preferable in a real-time system.

Additionally, we need to support variable skeleton structures and plan to allow users to create novel skeleton structures. In the prototype, we used a text-based skeleton definition capable of handling various skeletons, but we do not expect this to be usable by a novice user. An intuitive and easy-to-use tool, such as RigMesh [25], is needed to achieve this goal. We might need to support primitives other than cylindrical limbs in order to support objects with various skeleton structures, such as human hands and plants, in the real world. Sphere-Meshes [174, 175] might be useful to represent flat parts (i.e., palms and leaves) in these shapes.

Another interesting direction would be to support motion editing. Currently, the proposed system relies on motion provided in a database [2] for 3D animation. As demonstrated by AniMesh [94], applying human motion on the fly to a 3D model would be helpful to novice users.

The proposed method can be used to generate high-fidelity, textured 3D models. In this study, we focused only on the shape refinement and rigging of the scanned 3D mesh; however, we also captured the high-resolution photographs of 3D objects during the scanning process. Although we utilized these only in the annotation step, they can also be used as texture information for the final 3D mesh model. To support this functionality, we might need to integrate a texture mapping method into our system. Furthermore, this might require a texture-synthesis method to cover the invisible or uncaptured area in the scanning session.

## 3.8 Summary

In this chapter, we described a system allowing users to generate a rigged 3D mesh from a raw-scanned 3D volume with simple annotations and domain knowledge. We leveraged (1) floor information, (2) user-specified skeleton, and (3) major shape in plush toys as domain knowledge to animate plush toy model. In the proposed system, users are asked to annotate the skeleton structure according to registered photographs captured during the scanning step, after which the system segments the raw-scanned volume and generates a skinned 3D mesh based on the user-specified 3D skeleton. We tested our system using 10 raw-scanned 3D plush toy models and successfully generated clean, skinned 3D meshes and animations. Further research is required to improve the final shape, especially at the boundary of different bone associations.

# Chapter 4

# Transfer-based Detail Enhancement in Raw-Scanned Face Model

In this chapter, we describe our transfer-based method for improving the geometry detail in a raw-scanned 3D human face (Figure 4.1). Our target 3D model is a human 3D face in a raw-scanned 3D model, and we aim to create a 3D replica. However, a raw-scanned 3D model does not have proper geometry detail to be represented in a replica. To address this issue, 3D modeling experts usually exaggerate the geometry feature of the face. Inspired by their approach, we extract the detailed ridges and valleys from a retouched exemplar 3D model and transfer it to a target raw-scanned 3D model. The method in this chapter is mainly based on the work presented in Computer Graphics International 2017 [131].

## 4.1 Scope of Application

The recent development of 3D technologies, such as 3D scanning and 3D printing, extends conventional photography to three dimensions. Today, people scan

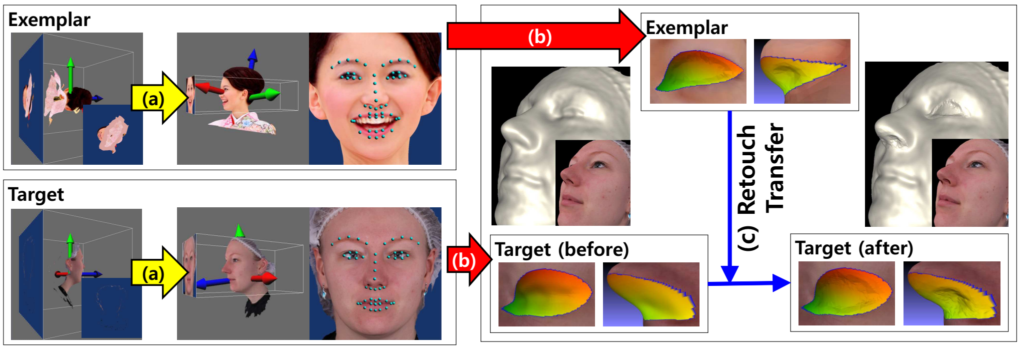


Figure 4.1: Overview of the proposed system for facial detail enhancement. Our system supports the transfer of the retouched 3D geometry from the exemplar to another raw-scanned target 3D model. (a) It automatically finds the face canonical view from arbitrary 3D models. (b) After acquiring the face canonical view, it extracts and parameterizes local patches of the facial components. (c) Once the parameterized patches are obtained, the system transfers the geometry detail with coating transfer [167].

themselves in a photogrammetry 3D scanning studio on a memorable day. Three-dimensional printing technology supports the reproduction of the captured 3D data as replicas. Commercial services are also available by using these technologies [50, 168, 172].

Even though overall information such as body pose and costume is important to remind the moment, people usually focus on the facial area to recognize the identity of the replicas. The latent technical issue in this context is the resolution of 3D scanning. When we capture a full human body, it is hard to capture details of the face at the same time. The naïve approach to solving this issue is merging the facial and body data after capturing them separately. However, this approach needs at least two different camera settings, which forces customers to undergo repetitive data captures and involves a lot of effort on merging those heterogeneous data in the end.

In the 3D printing industry, 3D modellers play an important role in solving this issue. They retouch the raw-scanned full body 3D model that has low-quality facial geometry with a 3D sculpting tool, such as Zbrush [4]. The problem is that the retouching process takes a lot of time and effort even for a professional expert. If we automate this process by leveraging the domain knowledge for retouching, we can reduce the time and effort by human experts. In our observation on the raw and retouched 3D model pair, we found that the retouching is performed by referring on the surface texture of the human face. This editing is quite different with template-based fitting approach that is commonly used in the research community, from the aspect of the high-level knowledge involved in. Based on our finding, we leverage the image-based face detection and mesh parameterization techniques to simulate the knowledge-based editing of the raw-scanned 3D model.

We present a method to automate the retouching process by transferring a retouched result by an expert to an arbitrary target face model (Figure 4.1). To do this, we extract the facial components from a raw-scanned model and an artist-retouched 3D model through exploiting 2D face landmark detection. By using this method, we transfer the geometric details of an artist-retouched model to a raw-scanned model to add details to the local parts. Each component of the face is locally parameterized by the reprojected 3D landmark points from an off-the-shelf 2D face detector and deformed by our patch-based transfer method. The entire process is fully automatic, and the only thing to provide is the exemplar model that is retouched by an artist.

## 4.2 Existing Approaches

After Beeler and his colleagues' work [14], the passive multi-view stereo-based 3D scanning method has been widely used to reconstruct 3D human face models. Recent research directions are largely divided into two ways. One direction is to generate and control the 3D face model with the video input from a monocular camera [83, 30, 64]. The other focuses on a 3D reconstruction of each facial part, such as the eye [18], lip [65] and teeth [187]. Although these state-of-the-art technologies generate a high-quality 3D model of a human face, they may need additional near-range capture after the full body scan. In the context of a photo shoot for a 3D printed souvenir, this process is too cumbersome for customers. Our main target is to generate a 3D printed replica, and we would like to achieve adequate quality for 3D facial geometry with a single-shot full-body capture.

The combination of 3D scanning and 3D printing to make a customized figurine has been a popular technology. Tena et al. [172] introduced a method that transfers a human face to a tailored 3D figurine model which is created by a 3D artist. They extracted the face template mesh from a 3D figurine model in advance and deformed the scanned human face mesh to fit it in the template. To make the scanned human face correspond to the template, their method needs to manually set landmark points. In our case, however, there is no 3D model and information before the 3D scanning. The need for a template is also diminished; therefore, we modify the raw-scanned 3D model directly. In addition, we do not need manual input since we adopt the face detection method as a part of our system. Li et al. [115] proposed an automatic pipeline that enables users to capture themselves with a commodity depth sensor. This approach may substitute the photogrammetry studio but cannot handle the detailed geometry of the human face, either. More recently, Echevarria et al. [50] presented a method for capturing human hair for 3D printing. This method is specialized for hair; therefore, the method cannot be applied to enhance facial details.

There were several attemps to stylize and edit the reconstructed mesh. Jachnik et al. [89] introduced the interactive stylization system for scanned human face. It utilized semantical segmentation on frontal face area based on educational material for students of artistic sculpture [109]. However, this work mainly focused on stylizing the *global* shape of the human face. Therefore, it is not possible to apply this method for local geometric feature enhancement, which is our main objective. In terms of replacing the part of original 3D model to another, Takayama et al. introduced Geobrush system [171]. This system, however, is not applicable in our problem because the source shape is also deformed based on the texture information of target. By parameterization from reprojected 3D landmark points, our system adaptively transfers the local geometry based on texture information.

## 4.3 Problem Formulation

### 4.3.1 Motivation

We show a typical example of retouching in Figure 4.2. We observe that these retouches are applied by referring to the surface texture; the artist identifies the

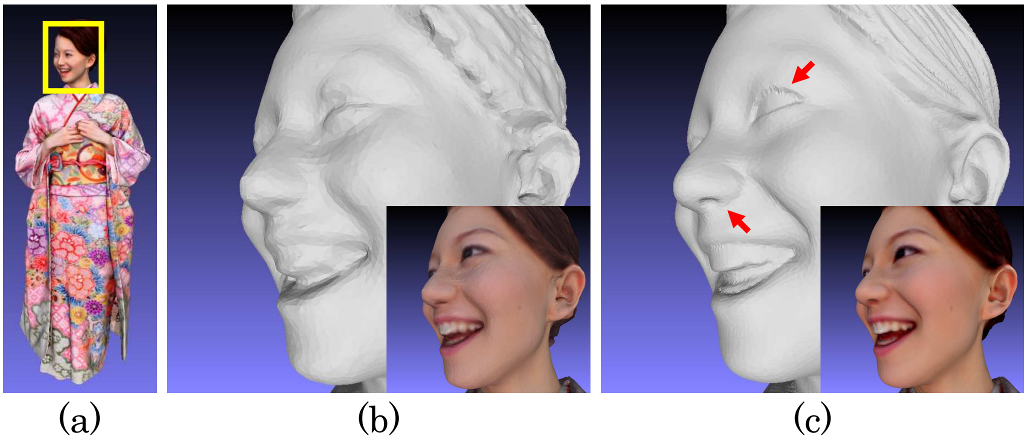


Figure 4.2: The comparison of raw-scanned 3D geometry (b) and retouched by an artist (c) from a full body scanned model (a). Note that the professional 3D modeller refines the local geometry of the raw model according to the texture information.

eyes by referring to the texture and edits the eyes. The nostrils are also created even though they were not successfully reconstructed by concavity. Based on this observation, we try to automate this retouching task by combining the face detection and mesh deformation methods.

To automate the retouching task by geometry transfer method, we need to solve the following issues: (1) find the face from the 3D model. Unlike the 2D image, the 3D model has more degree-of-freedom to generate 2D image. Directly using the 2D texture is not reliable, because it may have distortion for texture mapping on 3D model. (2) obtain the correspondences between the exemplar and target the 3D models. The face shapes in these models are different, so it is an issue of how to obtain the correspondences. (3) transfer the geometry feature from exemplar to target 3D model. Even though we get the correspondences by (2), it is not certain which geometry feature to transfer.

### 4.3.2 System Overview

An overview of the system is shown in Figure 4.1. As input data, our method needs two different textured 3D models that contain a face. One is the *Exemplar* model $\mathcal{E}$, which is retouched by an expert to emphasize the facial features. Another is the raw-scanned *Target* model $\mathcal{T}$, which we want to enhance by transferring the retouching in $\mathcal{E}$. Thanks to the automatic face alignment, we do not need to specify the facial landmark points manually. The system then extracts the patches and parameterizes them. Although the shapes of $\mathcal{E}$ and $\mathcal{T}$ are different, the system acquires dense correspondences with parameterization. By using these correspondences, the system can transfer the detailed facial component geometry from $\mathcal{E}$ to $\mathcal{T}$.

## 4.4 Algorithm

### 4.4.1 Finding the Face from the 3D Model

To automate retouching of a face model, we first identify the facial components in advance. This task seems quite easy manually, but there are two difficulties in automating it. One is the principle of 3D axes. It differs from the others based on the 3D engine used by the artist (e.g., Z-up in 3ds max and Y-up in Maya). Nonetheless, there is no guarantee that each model is always aligned to the up-direction in the 3D engine. Another aspect is the efficiency of automation. Due to its high degree-of-freedom, simple quantization of the viewpoint for 3D model will generate many futile attempts in the exploration. Due to these reasons, most previous works ([167, 172]) set landmark points manually.

In our approach, we automatically find the face canonical view and facial components efficiently. This automatic process exploits the off-the-shelf fast 2D face landmark detection method. We adopt the dlib face detector [103] in our prototype due to its availability, but any face landmark detection method can be utilized. The search is divided into two phases: rough exploration and detail refinement. Since we use the 2D landmark detector, we reproject the 2D pixels to the 3D model when we need 3D information.

In the rough exploration, the system sequentially generates an image from multiple viewpoints and applies face detection. Once the face is found in this sequence, it switches to the detail refinement phase. To guarantee the visibility of the 3D model in the search (Figure 4.3), the system first calculates the axis-aligned bounding box from the 3D model. The center of the model and the circumscribed sphere can be extracted from this bounding box. Finally, the system adjusts the view volume to inscribe the sphere above. For search efficiency, we used an icosphere to determine the rotation of the 3D model, inspired by Hinterstoisser et al. [74] that was originally used to generate a planar image

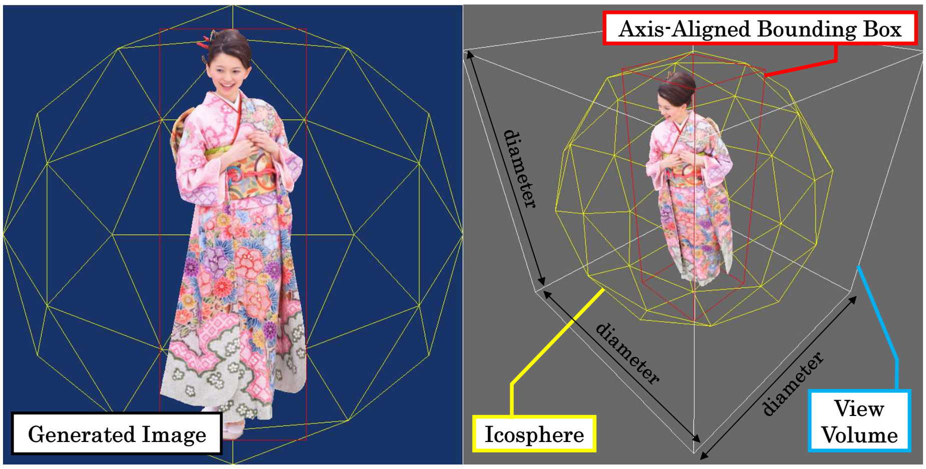


Figure 4.3: Our rendering setting for rough exploration.

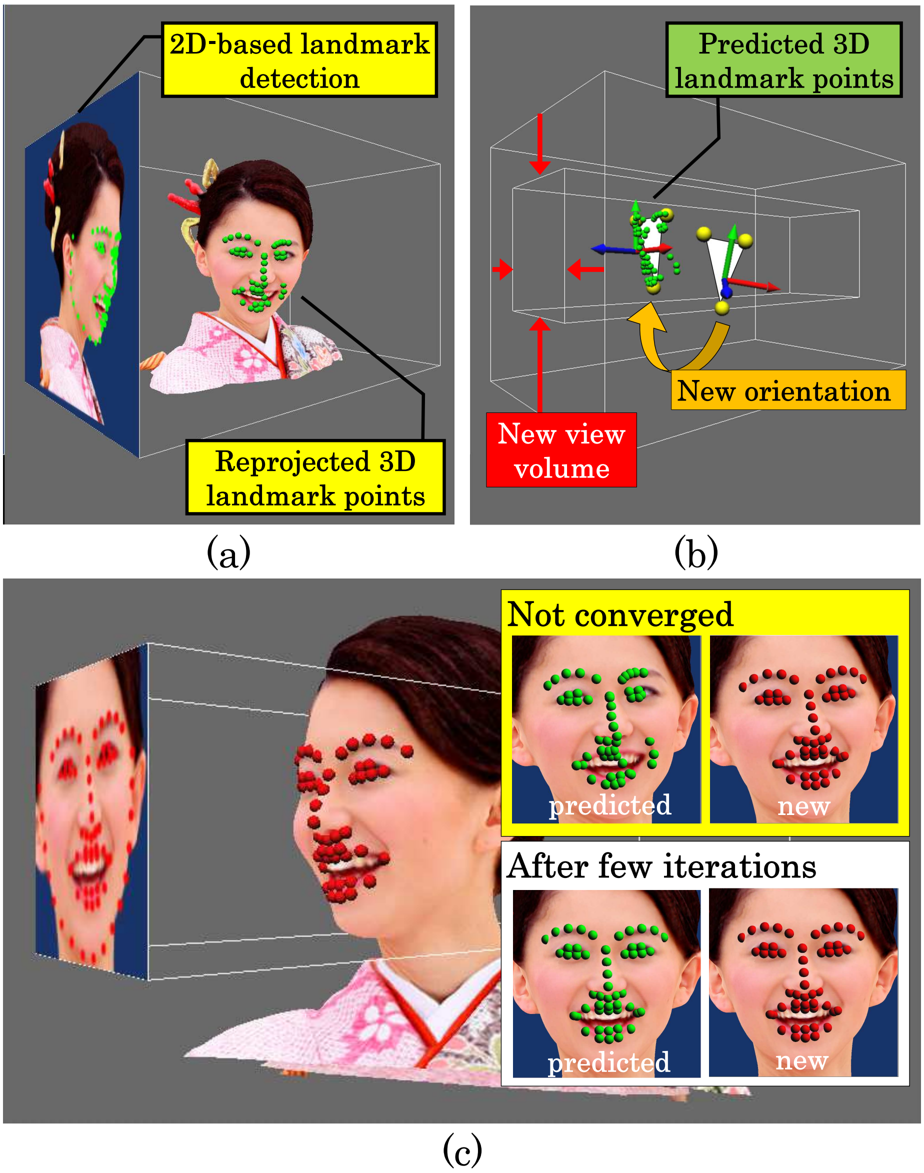


Figure 4.4: Our refinement algorithm for the face canonical view. (a) The relationship between 2D face detection and the 3D model. (b) New view setting based on appointed points. (c) Snapshot of the convergence test.

descriptor. In our system, we utilize the level 1 icosphere that has 42 vertices. To combine these vertices with 6 up directions (3 axes with both directions) and filter the 36 degenerated orientations, we sequentially check the 216 possible views in this phase.

In the detail refinement phase, the system iteratively refines the canonical view (Figure 4.4). The errors from partial occlusion and false-positive detection can be filtered in this phase. This process involves two operations: 3D model rotation and view volume adjustment. First, the system picks three appointed 3D landmark points (two from the eyebrows and one from the mouth) and rotates the 3D model so that the triangle formed by the landmarks faces the screen. The system then projects the rotated 3D landmark points to the 2D screen space (called the predicted 2D positions). Based on these predicted 2D positions in the screen space, the system also adjusts the view volume to situate those predicted points. After assigning the new 3D information for the refined view, the system renders (Figure 4.4c) and applies the face detection to get new landmark points. By comparing these predicted and new landmark points, the system determines the convergence of the canonical view.

We consider that the predicted and new 2D positions are the same when the distance between them is less than 3. The refinement step empirically converges within 2-5 steps in our preliminary test by using this criterion. In our experiment, we set the maximum iteration at 10 for safety. If it does not converge within 10 steps, then the system considers it a failure of the local refinement and goes back to the rough exploration.

### 4.4.2 Extracting the Facial Component Patches

Since we do not use a face template, we need to handle the differently tessellated models. To solve this issue smoothly, we utilize the dense grid mesh generated from the rendered image as an intermediate representation of the mesh transfer. In these grid meshes, a rendered pixel becomes a 3D vertex in a grid mesh. We determine the view volume for each facial component by using the reprojected 3D landmark points (Figure 4.5). The system gathers the 3D landmark points and specifies the screen width and height from the minimum and maximum values of the x- and y- elements. In the case of the nose, we rotate the camera at a fixed angle to capture the nostrils. In our prototype, the angle is 30 degrees. In this way, we generate a denser grid mesh than the original mesh.

After acquiring grid meshes from the raw and the retouched 3D exemplar, we specify and extract the retouched features from the information (Figure 4.6). We first parameterize the grid mesh using the 3D landmark points as *uv* constraints. For a parameterization method, we utilize Zwicker et al.'s method [200] due to its stability. Once the parameterization is obtained, we compare them and carved the unmodified vertices (Figure 4.6c). We refer to the carved grid mesh as *patch*.

### 4.4.3 Transferring the Local Geometric Features

The last step is transferring the local geometry from $\mathcal{E}$ to $\mathcal{T}$. We illustrate our transfer algorithm in Figure 4.7. To achieve this goal, we utilized the coating transfer method proposed by Sorkine et al. [167]. Since we acquire the dense correspondences with parameterization, there are two factors to be resolved in coating transfer: rotation and scale. We do not need to take care of global rotation because all the computations are operated in patches that are aligned with the rendering camera. The main issue is the scale factor.

In most cases, the global scale of the exemplar and the target are different. Using the original values of the exemplar without appropriate scaling may cause

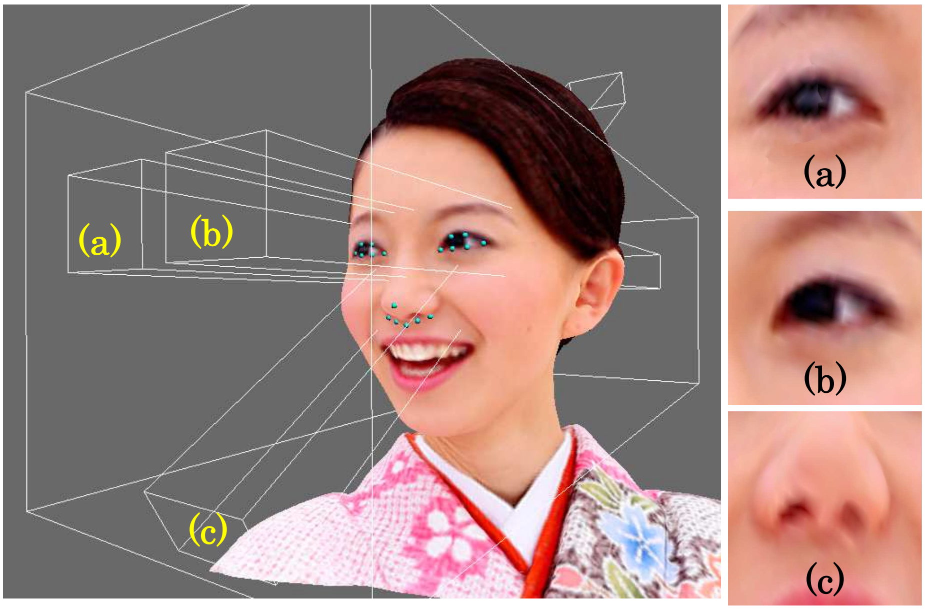


Figure 4.5: (Left) View volumes specified by 3D landmark points in the canonical view. (Right) Generated grid mesh from each view volume.

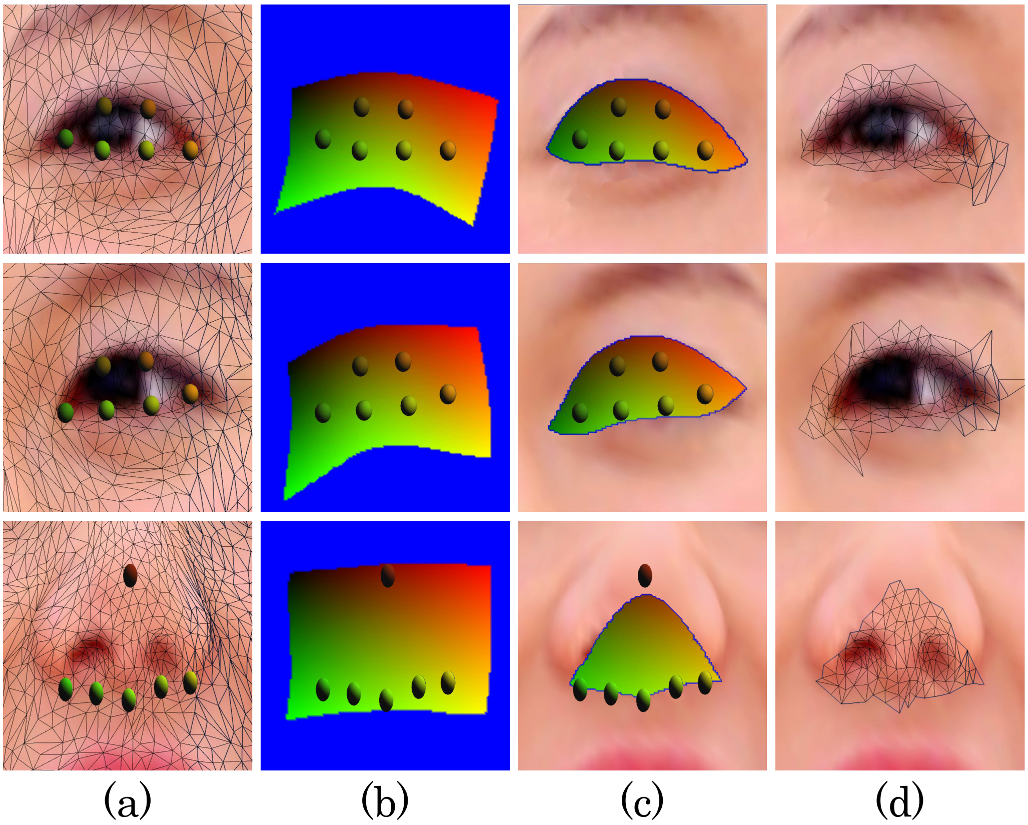


Figure 4.6: (a) Original mesh and parameterization constraints. (b) The parameterization result from the constraints. The red-green color is mapped to *uv*. (c) Patch for local geometry. (d) Submesh from the original model that is overlapped with a patch.

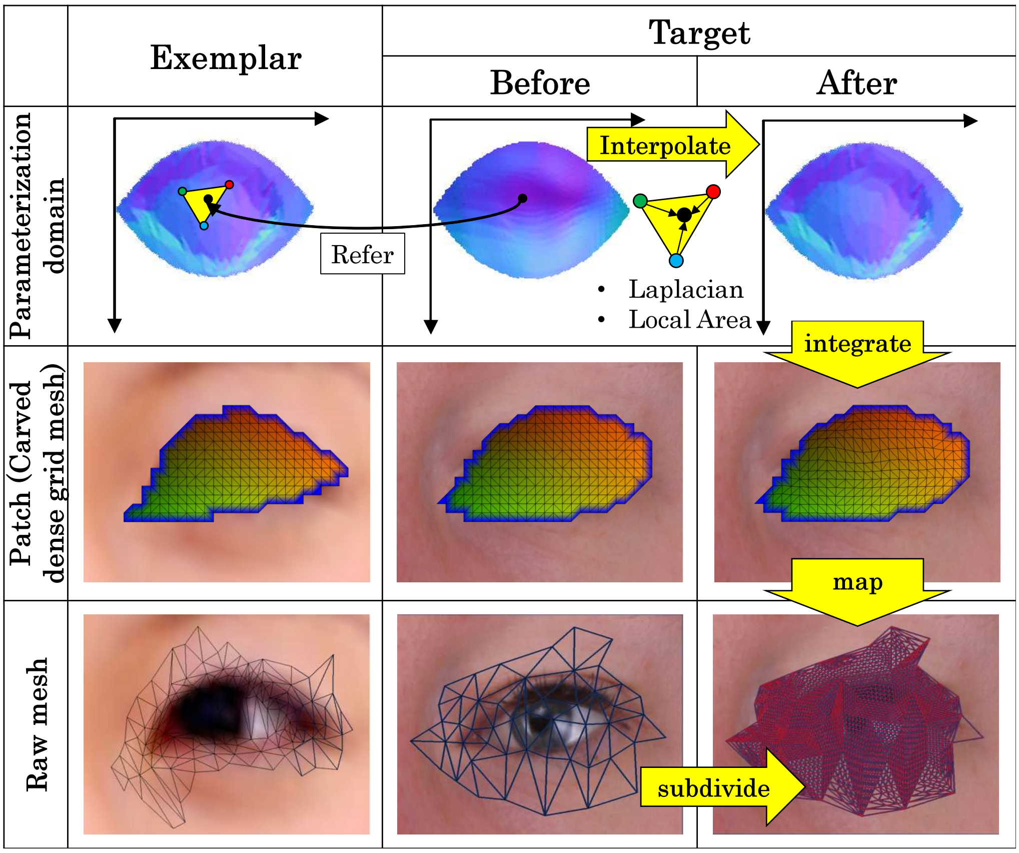


Figure 4.7: Our patch-based local coating transfer algorithm. The density of the patch is exaggerated for the clearance.

flattening or numerical explosion. To solve this issue, we also consider the scale factor in the coating transfer. Since Laplacian means the difference of the vertex position from the average position of its neighbor vertices, we can also consider its relative impact on its local area. Based on this observation, we rescale the exemplar's Laplacian value per its unit area (Equation 4.1). The operator area($\mathcal{M}$, $\mathbf{u}$) finds the triangle that contains coordinate $\mathbf{u}$ from mesh $\mathcal{M}$ and calculate the area of that triangle. This procedure corresponds to transferring *curvature* from $\mathcal{E}$ to $\mathcal{T}$.

$$\delta_{\mathcal{T}}^{(u,v)} \leftarrow \sqrt{\frac{\mathrm{area}(\mathcal{T},(u,v))}{\mathrm{area}(\mathcal{E},(u,v))}}\,\delta_{\mathcal{E}}^{(u,v)} \tag{4.1}$$

After reconstructing the target patch with coating transfer, the system maps the vertices of a patch to the raw-scanned model. The straightforward way is stitching the mesh, but it usually causes problems in the mesh topology. Instead of stitching, we map the vertices of a patch to the original model. To compensate the insufficient mesh resolution, we subdivide the original mesh to get detailed resolution. With this approach, we automate the retouch transfer for a human face 3D model.

| Image resolution | Rendering time [ms] | Detection time [ms] | Face | Bust | Figurine |
|---|---|---|---|---|---|
| $128^2$ | 9.785 | 6.789 | ✗ | ✗ | ✗ |
| $256^2$ | 10.403 | 31.814 | ✓ | ✗ | ✗ |
| $384^2$ | 11.262 | 73.197 | ✓ | ✓ | ✗ |
| $512^2$ | 13.579 | 125.253 | ✓ | ✓ | ✗ |
| $768^2$ | 22.375 | 283.693 | ✓ | ✓ | ✓ |
| $1024^2$ | 29.487 | 503.991 | ✓ | ✓ | ✓ |

Table 4.1: Average elapsed time for a single step in the face search.

| Model | Rough | | Detail | |
|---|---|---|---|---|
| name | time [ms] | #iter | time [ms] | #iter |
| F1002_N | 1981.93 | 14 | 2858.56 | 18 |
| M1014_H1 | 6317.52 | 44 | 5409.14 | 34 |
| F1018_N | 8761.37 | 61 | 8263.84 | 51 |
| M1045_N | 597.02 | 4 | 1935.81 | 12 |
| F1022_N | 9730.48 | 66 | 966.56 | 6 |
| M1047_N | 571.08 | 4 | 2289.56 | 14 |
| F1023_H1 | 6876.03 | 47 | 5240.23 | 32 |

Table 4.2: Elapsed time for face alignment. The model names are from [71]. The order corresponds to that in Figure 4.8.

## 4.5 Experiment Results

We measured the performance of the proposed system. In the experiment, we used a laptop system that consisted of a Core i7-6500U CPU, 16GB RAM and a NVidia GeForce 940M GPU for the performance evaluation. For the 3D data, we tested several models from different sources. We used the ESRC database [71] that includes about 100 individual 3D faces for the 3D face model. For the bust and figurine model, we used a small set of retouched models gathered from artists.

Our system is largely divided into two parts; the canonical view search and local geometry refinement by coating transfer. We measured their performance separately.

### 4.5.1 Performance of the Canonical View Search

We measured the performance of the canonical view search. The total elapsed time for this procedure is largely dependent on the image resolution. A smaller resolution generates a faster response per single step in the iteration, but the risk of skipping becomes larger due to the lack of image resolution. For this reason, we first measured the average time for a single step and the possibility of face finding.

Table 4.1 shows the average time for each iteration in the rough exploration phase. The rendering time was not increased significantly by GPU acceleration. On the contrary, the detection time increased linearly depending on the number of pixels. The right column shows whether our face alignment method can detect the face or not. The result varies due to the relative size of the face in the 3D model. Since our exemplar is a bust type and for safety, we determined $512^2$ for the face alignment.

Next, we measured the total elapsed time for the canonical view search. Since we have no prior in the axes convention, we randomized the order of the icosphere vertex in the rough exploration phase. Table 4.2 shows the elapsed time in the face canonical view search. It involves rough exploration and detail refinement, so we show the elapsed time and the number of iterations. We also show the acquired canonical views from 7 individuals in the left column of Figure 4.8.

### 4.5.2 Local Geometry Refinement

We show the original and refined models in Figure 4.8. To acquire these refined models, we set the grid mesh size at $128^2$. The processing time varied in each model, but it generally took 1-2 minutes to complete the entire procedure. In the comparison of the original raw-scanned and refined models, we confirmed that the geometry feature was adjusted to the texture information, although the exemplar of transfer was equal to that in Figure 4.2c.

We also confirmed our result in the 3D printed replica. Figure 4.9 shows the comparison of the 3D printed replicas. To generate this replica, we cut the 3D face model manually without editing the facial surface for a baseline (Figure 4.9 left). We then ran our method on this cut model to acquire the refined model (Figure 4.9 right). Although the proportion of refinement is very subtle, it improved the impression of 3D printed replica.

## 4.6 Discussions

In the rough exploration phase, we used the orthographic projection that generates an unusual view at a glance. It is also possible to extend our face alignment method to perspective projection, but we also need to concern about the field of view in this case. Moreover, we have no prior on the position and rotation of the face in the 3D model. This enforces the initial camera position at a distance from the center of the 3D model that makes the human face smaller in the image domain due to perspectivism. As a result, it has a bad influence on the search because face detection usually does not work when the face is too small in the image (i.e., Table 4.1). Because of this, we believe that orthographic projection is sufficient for rough exploration in our canonical view search.

It is possible to extend our work to other facial components, for example, eyebrows and the lip area. To achieve this goal, we may need a denser facial landmark set. Our current implementation is dependent on the 68 landmarks provided by the dlib face detector [103], which is defined in [153]. This landmark

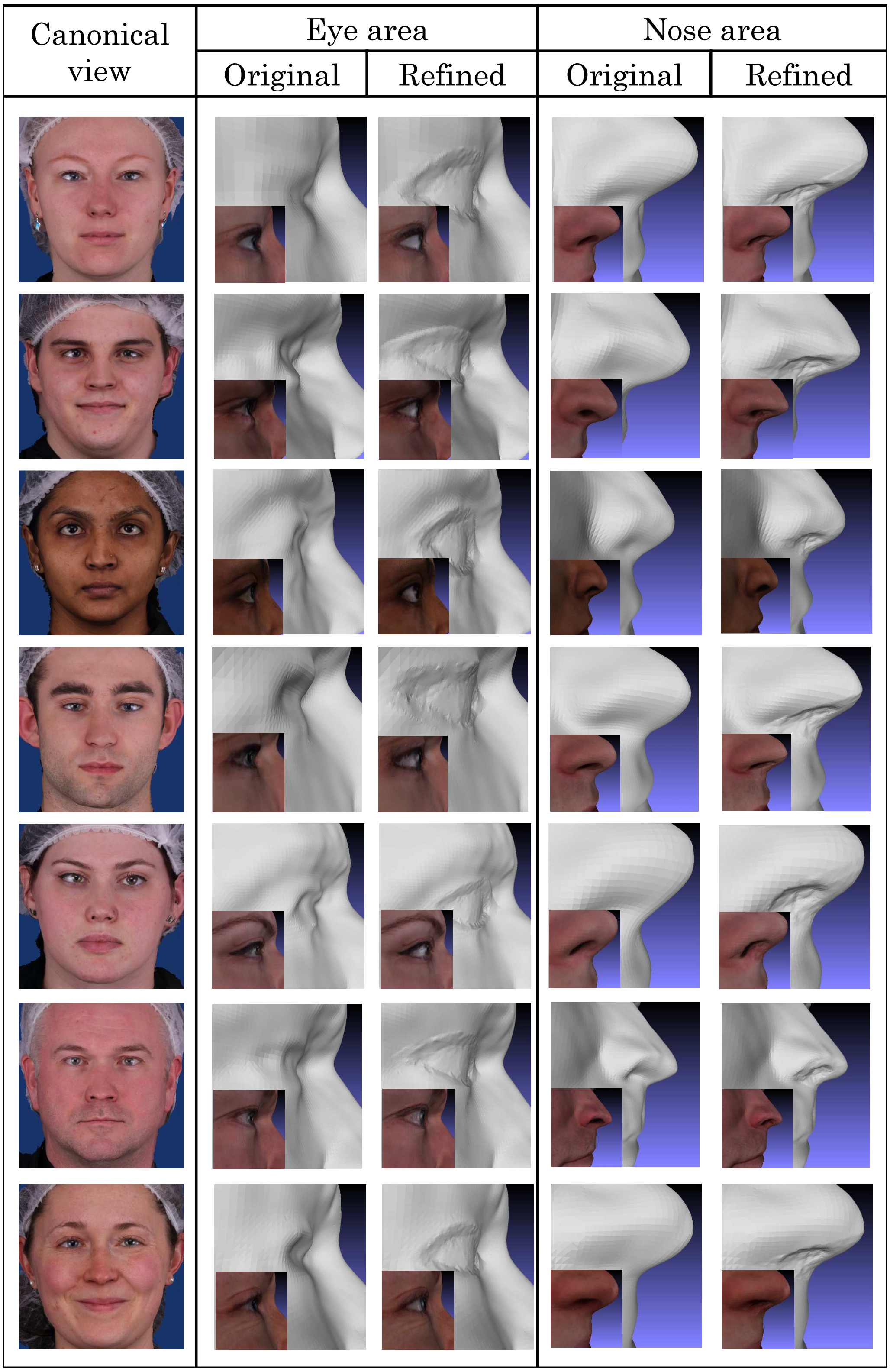


Figure 4.8: Our results on Stirling/ESRC 3D face database [71].

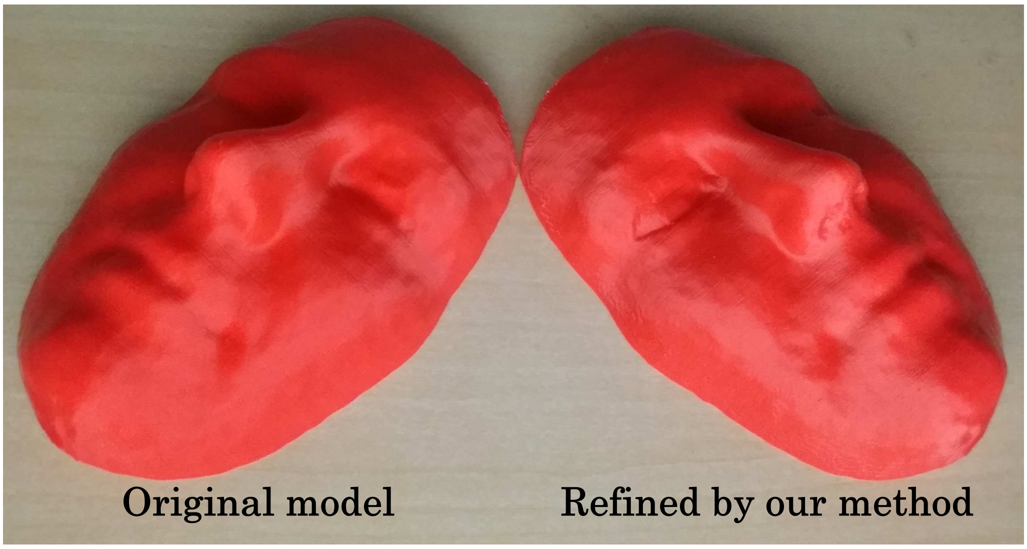


Figure 4.9: The comparison of the 3D printed replica between the raw-scanned (left) and refined (right) models.

set, however, has only the centerline information about eyebrows. It causes difficulty in parameterization because we cannot specify their boundaries from the landmark data only. If we adopt a denser facial landmark set (e.g., Helen [111]), we may achieve parameterization of these areas. It also gives a chance to improve the result of our current implementation (i.e., Figure 4.10), by using these additional landmark points in our parameterization step. To do this, we also need to explore various parameterization methods and *uv* constraints on these points, though.

### 4.6.1 Limitations

Our approach generates mesh slivers on the 3D model due to mesh subdivision (Figure 4.7, bottom right). In this chapter, we only aim to generate the improved 3D model for 3D printed replica. We intentionally ignore those artifacts to ensure the water-tightness of the result mesh for 3D printing. However, these artifacts are not so good in the 3D model in general, because it is not only visually pleasing but also needing excessive data size. Instead of subdivision and mapping approach, stitching patch (Figure 4.7, center row) with the raw mesh will be better substitution. However, it also involves with more complex mesh processing, such as mesh cutting and stitching, to ensure the water-tightness for 3D printing.

Although we automated the enhancement of a scanned 3D human face, the method has several limitations. The major limitation came from our component-wise patch transfer approach. Even if we adopt a denser landmark set [111], some parts of the human face are still difficult to be handled. For instance, transferring the edit of cheek or forehead areas are difficult, because it is difficult to find correspondence points on those areas in the human face.

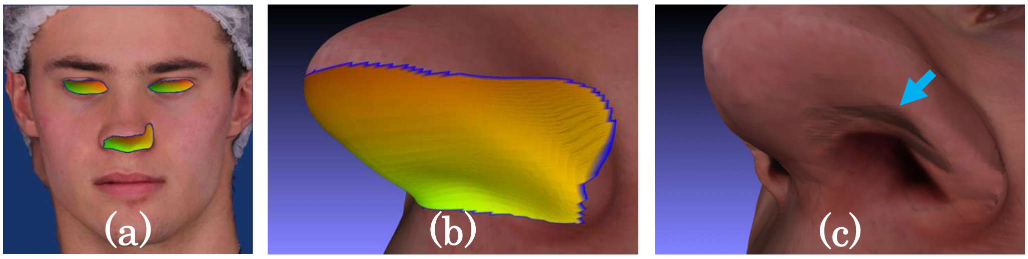


Figure 4.10: Less successful result. (a) The parameterization result in the canonical view. (b) Zoomed-in view of the nose patch. It exceeded the bottom of the nose part. (c) It also affected the final retouched geometry.

Robustness of the parameterization is also another issue in our approach. Figure 4.10 shows an example of a less successful result. Since we empirically determined the *uv* constraints for parameterization without optimizing the cross parameterization domain (e.g. [107]), the patch area would exceed the bottom part of the nose in a few cases. It would break the dense correspondence mapping from the exemplar to the target, so the transferred shape is not aligned to the texture information as well.

## 4.7 Summary

In this chapter, we described a raw-scanned 3D human face refinement system by transferring the local geometry from an artist-retouched 3D human face. We leveraged ridges and valleys geometry information of the human face as domain specific knowledge to create a 3D printed replica. To realize this concept, we implemented the system that automatically finds the canonical view and facial components by exploiting the 2D landmark detector in the 3D domain. By using texture-based parameterization, our method adjusted the local geometric feature to the target, which does not simply clone the shape of the exemplar. We also confirmed that the transfer-based method improves the visual quality of not only 3D data but also 3D printing.

５２ページから７０ページは現在特許申請及び学術誌へ投稿し審査途中であるため，一般公開できません．

# Chapter 6

# Conclusion

In this dissertation, we have first reviewed 3D scanning literature and recent 3D scanning methods under our classification. This chapter briefly summarizes our work, and discusses on the future research directions.

## 6.1 Summary of Contributions

In this dissertation, we first reviewed 3D scanning literature and classified the recent 3D scanning method into three types: (1) depth-fusion methods (2) multi-view stereo methods and (3) single-view scanning methods. Next, we also reviewed the recent advances in each method. Based on the observation of the current 3D scanning technology, we proposed the methods that are based on the domain specific knowledge.

In the case of skeleton-based method, we proposed a system that allows users to generate a rigged 3D mesh from a raw-scanned 3D volume with simple annotations. We designed the system that allows a user to annotate the skeleton structure according to registered photographs captured during the scanning step, after which the system segments the raw-scanned volume and generates a skinned 3D mesh based on the user-specified 3D skeleton. We leveraged the user-specified skeleton and several priors on the shape, such as floorplane, as domain knowledge to refine the scanned 3D shape. We tested our system using 10 raw-scanned 3D plush toy models and successfully generated clean, skinned 3D meshes and animations.

In the case of the transfer-based method, we proposed a raw-scanned 3D mesh refinement system by transferring the local geometry from an artist-retouched 3D model. We leveraged the human face structure as the domain knowledge, and image-based face detection to find facial part on 3D model. We extracted partial geometry feature of human face on the retouched model, and transfered those extracted features to the target 3D model. We tested this idea for improving a raw-scanned human face 3D model, so we design the system that automatically found the canonical view and facial components by exploiting the 2D landmark detector in the 3D domain. By using texture-based parameterization, our method adjusted the local geometric feature to the target, which does not simply clone the shape of the exemplar. We also confirmed that our method improves the visual quality of not only 3D data but also 3D printing.

In the case of deep feature-based parameter estimation method, we designed the system which gets a photograph of a real fur example as input and automatically estimates the fur parameters. We leveraged the expert's intention and workflow for reproducing fur strands in our preliminary study. We formulated this as an optimization problem so that the appearance of the rendered parametric fur are as similar as possible to the real fur. In each optimization step, we rendered the image using an off-the-shelf fur renderer and measured image similarity using the pre-trained model of a deep convolutional neural network. We evaluated our framework using rendered and real fur images and certificated that it works in most cases.

## 6.2 Possible Application Domains

In this dissertation, we only covered three application domains as examples. However, we believe that our approach is general and can be applied to other domains. We list the representative domains below.

**Buildings:** Buildings are one of the representative man-made objects. They have typical geometric structure and façade patterns. This characteristic is good for procedural modeling, so there has been several research prototypes and commercial product . However, those methods needed a manual adjustment of the parameters. Recently, Nishida and colleagues [130] attempted to reconstruct a 3D procedural model of building from a single image. They succeeded to accurately reconstruct a façade, but the feasible building structure is still limited in this method. It is likely that treating the complex structure will be the next goal in this direction.

**Animals and Insects:** Until now, most researches on 3D scanning have been focused on human capturing. Another dynamic object, such as an animal, is rarely handled as a scanning target object. This is not only because needs are limited, but also because non-human object is uncooperative to 3D scanning. Until recently, a limited number of researches handle to reconstruct animals. Recently, Zuffi and colleagues [199] introduced a method to reconstruct quadruped mammals from a single image. This method transfers learning of parametric model for human body [119] to animal shapes. This is an encouraging result, but it is still questionable that reconstructing non-mammal shapes (e.g., birds, aquatic life, and insects) can be achieved by this approach.

**Trees and plants:** Trees and plants are also an appropriate domain of our approach. There is a well-known plant development rule in botany, L-system [117], and several interactive modeling systems [85, 118] are based on this rule. Unfortunately, these tools need manual operations by human, and there is no way to create these model from a real-world image. Meanwhile, many scan methods for trees and plants does not reconstruct parametric models yet.

**Natural Phenomena:** Not only the man-made and natural objects, but also the natural phenomena will be good target domain for our approach. Recently, procedural model for natural phenomena, such as cloudscapes [179] and landscape [62], are introduced. These procedural models are based on the domain knowledge of nature, such as meteorology and geology, but they do not support image-based reconstruction yet.

These application domains have the similar technical issues; 3D procedural models already exists based on the domain knowledg, but they need a human labor to create 3D model. At this point, we can utilize the method in Chapter 5. Besides, we also believe that the methods in Chapter 3 and 4 can be auxiliarily applied to reconstruct more detailed and accurate 3D model. For example, wrongly fused branches in a raw-scanned 3D tree model can be refined by our approach in Chapter 3.

## 6.3 Future Work

From the perspective of the user, the single-view scanning method is a more promising approach than other types of scanning methods. Generally, conventional scanning methods, such as multi-view stereo and depth fusion method, are not so good for novice users because they inevitably involve a tedious scanning process to cover various views of the object. In addition, a lot of information *leaks* when the sensing data goes through 3D scanning pipelines. Compared to them, the single-view scanning method does not force the user to a heavy burden and can minimize the information lost among the pipeline. The ambiguity and uncertainty in a single image can be solved by using the recent learning-based method.

The major issue in this approach is how can we achieve those methods with a small amount of learning dataset. Furthermore, is it possible to achieve those methods independent of the type of scanning object? In other words, can we *generalize* the learning process for the single-view scanning method? This will be a longstanding research issue.

Besides, it is also promising to explore the possibility of a differentiable renderer [120] in the scanning process. Until now, the differentiable renderer is usually realized by machine learning techniques (e.g., [99]), so the set of parameters in those renderers were quite arbitrary, which is very difficult to control by the novice user. For this reason, we kept using the off-the-shelf renderer, so we did not explore this direction deeper in this dissertation. However, the graphics renderer in the next generation (e.g., Mitsuba2 [129]) embeds the differentiation of rendered image as default. We believe that this functionality gives a large opportunity to improve 3D scanning pipelines.

Another interesting research direction is an exploration of the new data structure for conventional 3D scanning methods. Until now, many 3D scanning methods have been proposed, but most of them keep using a grid-based data structure (Regular grid [42] and octree [100, 101]). Although this structure allows stable 3D processing, the reconstructed 3D model tends to be overly smoothed by grid resolution. This data structure makes several issues, such as processing speed and data size. It also makes a difficulty to handle dynamic objects, although there were not so many works indicated in this aspect. In this dissertation, we did not explore this direction at all, but we believe that many issues above can be solved by a successful data structure for 3D scanning.

# Appendix A

# Fur Parameters

We show the list of our selected fur parameters for Chapter 5 in Table A.1. We selected 25 parameters from 89 parameters in Maya Fur and converted them to normalized space. We fixed the other 64 parameters as default in our experiment.

| Attribute | Default | Min | Max |
|---|---|---|---|
| Density | 15,000 | 5,000 | 30,000 |
| Length | 1.00 | 1.00 | 5.00 |
| BaseWidth | 0.08 | 0.01 | 0.10 |
| TipWidth | 0.00 | 0.00 | 0.10 |
| BaseCurl | 0.5 | 0.5 | 1.0 |
| TipCurl | 0.5 | 0.0 | 1.0 |
| Inclination | 0.0 | 0.0 | 0.9 |
| PolarNoise | 0.0 | 0.0 | 0.5 |
| PolarNoiseFreq | 5.0 | 1.0 | 20.0 |
| Scraggle | 0.0 | 0.0 | 0.5 |
| ScraggleFreq | 5.0 | 1.0 | 10.0 |
| ScraggleCorr | 0.0 | 0.0 | 0.5 |
| Clumping | 0.0 | 0.0 | 0.5 |
| ClumpingFreq | 5.0 | 1.0 | 50.0 |
| ClumpShape | 0.0 | 1.0 | 5.0 |
| TipColorR | 0.404 | 0.0 | 1.0 |
| TipColorG | 0.275 | 0.0 | 1.0 |
| TipColorB | 0.169 | 0.0 | 1.0 |
| BaseColorR | 0.091 | 0.0 | 1.0 |
| BaseColorG | 0.057 | 0.0 | 1.0 |
| BaseColorB | 0.030 | 0.0 | 1.0 |
| SpecularColorR | 0.240 | 0.0 | 1.0 |
| SpecularColorG | 0.246 | 0.0 | 1.0 |
| SpecularColorB | 0.280 | 0.0 | 1.0 |
| SpecularSharpness | 50.0 | 0.0 | 100.0 |

Table A.1: The selected 25 parameters in our framework. Among them, 15 parameters are related to the fur geometry, and others are related to color.

# References

[1] Autodesk maya. `https://www.autodesk.com/products/maya`, 1998–2019.

[2] CMU graphics lab motion capture database. `http://mocap.cs.cmu.edu/`, accessed 2019.

[3] Unity . `https://unity.com`, accessed 2019.

[4] ZBrush. `http://pixologic.com/zbrush`, accessed 2019.

[5] Kfir Aberman, Oren Katzir, Qiang Zhou, Zegang Luo, Andrei Sharf, Chen Greif, Baoquan Chen, and Daniel Cohen-Or. Dip transform for 3d shape reconstruction. *ACM Trans. Graph.*, 36(4):79:1–79:11, July 2017.

[6] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112, October 2011.

[7] Brett Allen, Brian Curless, Brian Curless, and Zoran Popović. The space of human body shapes: Reconstruction and parameterization from range scans. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, pages 587–594, New York, NY, USA, 2003. ACM.

[8] Tobias Grønbeck Andersen, Viggo Falster, Jeppe Revall Frisvad, and Niels Jørgen Christensen. Hybrid fur rendering: combining volumetric fur with explicit hair strands. *The Visual Computer*, 32(6):739–749, Jun 2016.

[9] Baptiste Angles, Daniel Rebain, Miles Macklin, Brian Wyvill, Loic Barthe, Jp Lewis, Javier Von Der Pahlen, Shahram Izadi, Julien Valentin, Sofien Bouaziz, and Andrea Tagliasacchi. VIPER: Volume invariant position-based elastic rods. *Proc. ACM Comput. Graph. Interact. Tech.*, 2(2):19:1–19:26, July 2019.

[10] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: Shape completion and animation of people. *ACM Trans. Graph.*, 24(3):408–416, July 2005.

[11] Ilya Baran and Jovan Popović. Automatic rigging and animation of 3d characters. *ACM Trans. Graph.*, 26(3), July 2007.

[12] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[13] Thabo Beeler. Passive spatiotemporal geometry reconstruction of human faces at high fidelity. *IEEE Computer Graphics and Applications*, 35(3): 82–90, May 2015.

[14] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. *ACM Trans. Graph.*, 29(4):40:1–40:9, July 2010.

[15] Thabo Beeler, Bernd Bickel, Gioacchino Noris, Paul Beardsley, Steve Marschner, Robert W. Sumner, and Markus Gross. Coupled 3d reconstruction of sparse facial hair and skin. *ACM Trans. Graph.*, 31(4):117:1–117:10, July 2012.

[16] 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. *ACM Trans. Graph.*, 30(4):75:1–75:10, July 2011.

[17] J. . Beraldin, F. Blais, L. Cournoyer, M. Rioux, S. H. El-Hakim, R. Rodella, F. Bernier, and N. Harrison. Digital 3d imaging system for rapid response on remote sites. In *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, pages 34–43, Oct 1999.

[18] Pascal Bérard, Derek Bradley, Markus Gross, and Thabo Beeler. Lightweight eye capture using a parametric model. *ACM Trans. Graph.*, 35(4):117:1–117:12, July 2016.

[19] Pascal Bérard, Derek Bradley, Maurizio Nitti, Thabo Beeler, and Markus Gross. High-quality capture of eyes. *ACM Trans. Graph.*, 33(6):223:1–223:12, November 2014.

[20] Amit Bermano, Thabo Beeler, Yeara Kozlov, Derek Bradley, Bernd Bickel, and Markus Gross. Detailed spatio-temporal reconstruction of eyelids. *ACM Trans. Graph.*, 34(4):44:1–44:11, July 2015.

[21] Paul J. Besl and Neil D. McKay. Method for registration of 3-D shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*, volume 1611, pages 586 – 606. International Society for Optics and Photonics, SPIE, 1992.

[22] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 187–194, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[23] F. Bogo, J. Romero, M. Loper, and M. J. Black. Faust: Dataset and evaluation for 3d mesh registration. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, June 2014.

[24] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 561–578, Cham, 2016. Springer International Publishing.

[25] Péter Borosán, Ming Jin, Doug DeCarlo, Yotam Gingold, and Andrew Nealen. Rigmesh: Automatic rigging for part-based shape modeling and deformation. *ACM Trans. Graph.*, 31(6):198:1–198:9, November 2012.

[26] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High resolution passive facial performance capture. *ACM Trans. Graph.*, 29(4): 41:1–41:10, July 2010.

[27] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, March 2014.

[28] Chen Cao, Derek Bradley, Kun Zhou, and Thabo Beeler. Real-time high-fidelity facial performance capture. *ACM Trans. Graph.*, 34(4):46:1–46:9, July 2015.

[29] Chen Cao, Yanlin Weng, Stephen Lin, and Kun Zhou. 3d shape regression for real-time facial animation. *ACM Trans. Graph.*, 32(4):41:1–41:10, July 2013.

[30] Chen Cao, Hongzhi Wu, Yanlin Weng, Tianjia Shao, and Kun Zhou. Real-time facial animation with image-based dynamic avatars. *ACM Trans. Graph.*, 35(4):126:1–126:12, July 2016.

[31] Menglei Chai, Linjie Luo, Kalyan Sunkavalli, Nathan Carr, Sunil Hadap, and Kun Zhou. High-quality hair modeling from a single portrait photo. *ACM Trans. Graph.*, 34(6):204:1–204:10, October 2015.

[32] Menglei Chai, Tianjia Shao, Hongzhi Wu, Yanlin Weng, and Kun Zhou. Autohair: Fully automatic hair modeling from a single image. *ACM Trans. Graph.*, 35(4):116:1–116:12, July 2016.

[33] Menglei Chai, Lvdi Wang, Yanlin Weng, Yizhou Yu, Baining Guo, and Kun Zhou. Single-view hair modeling for portrait manipulation. *ACM Trans. Graph.*, 31(4):116:1–116:8, July 2012.

[34] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015.

[35] A. Chatterjee and V. M. Govindu. Photometric refinement of depth maps for multi-albedo objects. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 933–941, June 2015.

[36] Jiawen Chen, Dennis Bautembach, and Shahram Izadi. Scalable real-time volumetric surface reconstruction. *ACM Trans. Graph.*, 32(4):113:1–113:16, July 2013.

[37] Tao Chen, Zhe Zhu, Ariel Shamir, Shi-Min Hu, and Daniel Cohen-Or. 3-Sweep: Extracting editable objects from a single photo. *ACM Trans. Graph.*, 32(6):195:1–195:10, November 2013.

[38] François Chollet et al. Keras. `https://keras.io`, 2015.

[39] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Trans. Graph.*, 34(4):69:1–69:13, July 2015.

[40] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph.*, 32(5):152:1–152:11, October 2013.

[41] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. The heat method for distance computation. *Commun. ACM*, 60(11):90–99, October 2017.

[42] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM.

[43] A. Dai, C. R. Qi, and M. Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6545–6554, July 2017.

[44] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph.*, 36(4), May 2017.

[45] Olivier Dionne and Martin de Lasa. Geodesic voxel binding for production character meshes. In *Proceedings of the 12th ACM SIGGRAPH/ Eurographics Symposium on Computer Animation*, SCA '13, pages 173–180, New York, NY, USA, 2013. ACM.

[46] Olivier Dionne and Martin de Lasa. Geodesic binding for degenerate character geometry using sparse voxelization. *IEEE Transactions on Visualization and Computer Graphics*, 20(10):1367–1378, Oct 2014.

[47] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. Fusion4d: Real-time performance capture of challenging scenes. *ACM Trans. Graph.*, 35(4):114:1–114:13, July 2016.

[48] Mingsong Dou, Jonathan Taylor, Henry Fuchs, Andrew Fitzgibbon, and Shahram Izadi. 3D scanning deformable objects with a single RGBD sensor. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 493–501, June 2015.

[49] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43, Oct 1995.

[50] Jose I. Echevarria, Derek Bradley, Diego Gutierrez, and Thabo Beeler. Capturing and stylizing hair for 3d fabrication. *ACM Trans. Graph.*, 33(4): 125:1–125:11, July 2014.

[51] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the rgb-d slam system. In *2012 IEEE International Conference on Robotics and Automation*, pages 1691–1696, May 2012.

[52] Sean Follmer, Micah Johnson, Edward Adelson, and Hiroshi Ishii. deform: An interactive malleable surface for capturing 2.5d arbitrary objects, tools and touch. In *Proceedings of the 24th Annual ACM Symposium on User*

*Interface Software and Technology*, UIST '11, pages 527–536, New York, NY, USA, 2011. ACM.

[53] S. Fuhrmann, M. Kazhdan, and M. Goesele. Accurate isosurface interpolation with hermite data. In *2015 International Conference on 3D Vision*, pages 256–263, Oct 2015.

[54] Simon Fuhrmann and Michael Goesele. Fusion of depth maps with multiple scales. *ACM Trans. Graph.*, 30(6):148:1–148:8, December 2011.

[55] Simon Fuhrmann and Michael Goesele. Floating scale surface reconstruction. *ACM Trans. Graph.*, 33(4):46:1–46:11, July 2014.

[56] Simon Fuhrmann, Fabian Langguth, and Michael Goesele. Mve: A multi-view reconstruction environment. In *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage*, GCH '14, pages 11–18, Aire-la-Ville, Switzerland, Switzerland, 2014. Eurographics Association.

[57] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Manhattan-world stereo. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1422–1429, June 2009.

[58] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Reconstructing building interiors from images. In *2009 IEEE 12th International Conference on Computer Vision*, pages 80–87, Sep. 2009.

[59] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Reconstructing building interiors from images. In *2009 IEEE 12th International Conference on Computer Vision*, pages 80–87, Sep. 2009.

[60] Yasutaka Furukawa and Carlos Hernández. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015.

[61] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1362–1376, August 2010.

[62] Eric Galin, Eric Guérin, Adrien Peytavie, Guillaume Cordonnier, Marie-Paule Cani, Bedrich Benes, and James Gain. A review of digital terrain modeling. *Computer Graphics Forum*, 38(2):553–577, 2019.

[63] Pablo Garrido, Levi Valgaert, Chenglei Wu, and Christian Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph.*, 32(6):158:1–158:10, November 2013.

[64] Pablo Garrido, Michael Zollhöfer, Dan Casas, Levi Valgaerts, Kiran Varanasi, Patrick Pérez, and Christian Theobalt. Reconstruction of personalized 3d face rigs from monocular video. *ACM Trans. Graph.*, 35(3): 28:1–28:15, May 2016.

[65] Pablo Garrido, Michael Zollhöfer, Chenglei Wu, Derek Bradley, Patrick Pérez, Thabo Beeler, and Christian Theobalt. Corrective 3d reconstruction of lips from monocular video. *ACM Trans. Graph.*, 35(6):219:1–219:11, November 2016.

[66] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems 28*, pages 262–270. Curran Associates, Inc., 2015.

[67] Leon. A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, June 2016.

[68] Yotam Gingold, Takeo Igarashi, and Denis Zorin. Structured annotations for 2D-to-3D modeling. *ACM Trans. Graph.*, 28(5):148:1–148:9, December 2009.

[69] Kaiwen Guo, Feng Xu, Tao Yu, Xiaoyang Liu, Qionghai Dai, and Yebin Liu. Real-time geometry, albedo, and motion reconstruction using a single RGB-D camera. *ACM Trans. Graph.*, 36(4), June 2017.

[70] B. Haefner, Y. Quéau, T. Möllenhoff, and D. Cremers. Fight ill-posedness with ill-posedness: Single-shot variational depth super-resolution from shading. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 164–174, June 2018.

[71] Peter Hancock and Bernie Tiddeman. Stirling/ESRC 3d face database. `http://pics.stir.ac.uk/ESRC/`, 2013. Accessed: 2017-04-06.

[72] Nikolaus Hansen. The CMA evolution strategy: A tutorial. *CoRR*, abs/1604.00772, 2016.

[73] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[74] Stefan Hinterstoisser, Vincent Lepetit, Selim Benhimane, Pascal Fua, and Nassir Navab. Learning real-time perspective patch rectification. *International Journal of Computer Vision*, 91(1):107–130, Jan 2011.

[75] David A. Hirshberg, Matthew Loper, Eric Rachlin, and Michael J. Black. Coregistration: Simultaneous alignment and modeling of articulated 3d shape. In *Computer Vision – ECCV 2012*, pages 242–255, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[76] Dirk Holz, Stefan Holzer, Radu Bogdan Rusu, and Sven Behnke. Real-time plane segmentation using rgb-d cameras. In *RoboCup 2011: Robot Soccer World Cup XV*, pages 306–317, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[77] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Robust hair capture using simulated examples. *ACM Trans. Graph.*, 33(4):126:1–126:10, July 2014.

[78] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Single-view hair modeling using a hairstyle database. *ACM Trans. Graph.*, 34(4):125:1–125:9, July 2015.

[79] Liwen Hu, Chongyang Ma, Linjie Luo, Li-Yi Wei, and Hao Li. Capturing braided hairstyles. *ACM Trans. Graph.*, 33(6):225:1–225:9, November 2014.

[80] Hui Huang, Shihao Wu, Daniel Cohen-Or, Minglun Gong, Hao Zhang, Guiqing Li, and Baoquan Chen. L1-medial skeleton of point cloud. *ACM Trans. Graph.*, 32(4):65:1–65:8, July 2013.

[81] Jingwei Huang, Angela Dai, Leonidas Guibas, and Matthias Niessner. 3dlite: Towards commodity 3d scanning for content creation. *ACM Trans. Graph.*, 36(6):203:1–203:14, November 2017.

[82] HyeokHyen Kwon, Yu-Wing Tai, and S. Lin. Data-driven depth map refinement via multi-scale sparse representation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 159–167, June 2015.

[83] Alexandru Eugen Ichim, Sofien Bouaziz, and Mark Pauly. Dynamic 3d avatar creation from hand-held video input. *ACM Trans. Graph.*, 34(4): 45:1–45:14, July 2015.

[84] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[85] Takashi Ijiri, Shigeru Owada, Makoto Okabe, and Takeo Igarashi. Floral diagrams and inflorescences: Interactive flower modeling using botanical structural constraints. *ACM Trans. Graph.*, 24(3):720–726, July 2005.

[86] Satoshi Ikehata, Hang Yang, and Yasutaka Furukawa. Structured indoor modeling. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1323–1331, Dec 2015.

[87] Matthias Innmann, Michael Zollhöfer, Matthias Nießner, Christian Theobalt, and Marc Stamminger. Volumedeform: Real-time volumetric non-rigid reconstruction. In *Computer Vision – ECCV 2016*, pages 362–379, Cham, 2016. Springer International Publishing.

[88] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM.

[89] Jan Jachnik, Dan B. Goldman, Linjie Luo, and Andrew J. Davison. Interactive 3d face stylization using sculptural abstraction. *CoRR*, abs/1502.01954, 2015.

[90] Alec Jacobson. Geometry processing in the wild. `http://www.cs.toronto.edu/~jacobson/images/geometry-processing-in-the-wild-alec-jacobson.pdf`, 2019.

[91] Alec Jacobson, Ilya Baran, Ladislav Kavan, Jovan Popović, and Olga Sorkine. Fast automatic skinning transformations. *ACM Trans. Graph.*, 31(4):77:1–77:10, July 2012.

[92] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.*, 30(4): 78:1–78:8, July 2011.

[93] Zhongping Ji, Ligang Liu, and Yigang Wang. B-Mesh: A modeling system for base meshes of 3D articulated shapes. *Computer Graphics Forum*, 29(7): 2169–2177, 2010.

[94] Ming Jin, Dan Gopstein, Yotam Gingold, and Andrew Nealen. Animesh: Interleaved animation, modeling, and editing. *ACM Trans. Graph.*, 34(6): 207:1–207:8, October 2015.

[95] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision – ECCV 2016*, pages 694–711, Cham, 2016. Springer International Publishing.

[96] Micah K. Johnson, Forrester Cole, Alvin Raj, and Edward H. Adelson. Microgeometry capture using an elastomeric sensor. *ACM Trans. Graph.*, 30(4):46:1–46:8, July 2011.

[97] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python. `http://www.scipy.org/`, 2001–.

[98] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Computer Vision – ECCV 2018*, pages 386–402, Cham, 2018. Springer International Publishing.

[99] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, June 2018.

[100] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[101] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13, July 2013.

[102] Maik Keller, Damien Lefloch, Martin Lambers, Kolb Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 1–8, June 2013.

[103] Davis E. King. Dlib-ml: A machine learning toolkit. *J. Mach. Learn. Res.*, 10:1755–1758, December 2009.

[104] Andreas Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight cameras in computer graphics. *Computer Graphics Forum*, 29(1):141–159, 2010.

[105] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[106] Jacek Kustra, Andrei C. Jalba, and Alexandru C. Telea. Probabilistic view-based 3D curve skeleton computation on the GPU. In *VISAPP 2013 : Proceedings of the International Conference on Computer Vision Theory and Applications, Barcelona, Spain, February 21-24, 2013*, volume 2, pages 237–246. INSTICC Press, 2013.

[107] Tsz-Ho Kwok, Yunbo Zhang, and Charlie C. L. Wang. Efficient optimization of common base domains for cross parameterization. *IEEE Transactions on Visualization and Computer Graphics*, 18(10):1678–1692, Oct 2012.

[108] Fabian Langguth, Kalyan Sunkavalli, Sunil Hadap, and Michael Goesele. Shading-aware multi-view stereo. In *Computer Vision – ECCV 2016*, pages 469–485, Cham, 2016. Springer International Publishing.

[109] Edouard Lanteri. *Modelling and sculpting the human figure.* Courier Corporation, 2012.

[110] John Lasseter. Principles of traditional animation applied to 3d computer animation. *SIGGRAPH Comput. Graph.*, 21(4):35–44, August 1987.

[111] Vuong Le, Jonathan Brandt, Zhe Lin, Lubomir Bourdev, and Thomas S. Huang. *Interactive Facial Feature Localization*, pages 679–692. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[112] Jerome Lengyel, Emil Praun, Adam Finkelstein, and Hugues Hoppe. Real-time fur over arbitrary surfaces. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, I3D '01, pages 227–232, New York, NY, USA, 2001. ACM.

[113] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 131–144, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[114] Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. Robust single-view geometry and motion reconstruction. *ACM Trans. Graph.*, 28(5): 175:1–175:10, December 2009.

[115] Hao Li, Etienne Vouga, Anton Gudym, Linjie Luo, Jonathan T. Barron, and Gleb Gusev. 3d self-portraits. *ACM Trans. Graph.*, 32(6):187:1–187:9, November 2013.

[116] Pan Li, Bin Wang, Feng Sun, Xiaohu Guo, Caiming Zhang, and Wenping Wang. Q-mat: Computing medial axis transform by quadratic error minimization. *ACM Trans. Graph.*, 35(1):8:1–8:16, December 2015.

[117] Aristid Lindenmayer. Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of Theoretical Biology*, 18(3):280 – 299, 1968.

[118] Steven Longay, Adam Runions, Frédéric Boudon, and Przemyslaw Prusinkiewicz. Treesketch: Interactive procedural modeling of trees on a tablet. In *Proceedings of the International Symposium on Sketch-Based*

*Interfaces and Modeling*, SBIM '12, page 107–120, Goslar, DEU, 2012. Eurographics Association.

[119] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: A skinned multi-person linear model. *ACM Trans. Graph.*, 34(6):248:1–248:16, October 2015.

[120] Matthew M. Loper and Michael J. Black. Opendr: An approximate differentiable renderer. In *Computer Vision – ECCV 2014*, pages 154–169, Cham, 2014. Springer International Publishing.

[121] William E. Lorensen and Harvey E. Cline. Marching Cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, August 1987.

[122] David G. Lowe, David G. Lowe, and David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[123] Linjie Luo, Hao Li, and Szymon Rusinkiewicz. Structure-aware hair capture. *ACM Trans. Graph.*, 32(4):76:1–76:12, July 2013.

[124] Xinhui Ma, Simeon Keates, Yong Jiang, and Jiří Kosinka. Subdivision surface fitting to a dense mesh using ridges and umbilics. *Computer Aided Geometric Design*, 32:5 – 21, 2015.

[125] Eitan Marder-Eppstein. Project tango. In *ACM SIGGRAPH 2016 Real-Time Live!*, SIGGRAPH '16, New York, NY, USA, 2016. Association for Computing Machinery.

[126] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 343–352, June 2015.

[127] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 127–136, Washington, DC, USA, 2011. IEEE Computer Society.

[128] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Trans. Graph.*, 32(6):169:1–169:11, November 2013.

[129] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Trans. Graph.*, 38(6):203:1–203:17, November 2019.

[130] Gen Nishida, Adrien Bousseau, and Daniel G. Aliaga. Procedural modeling of a building from a single image. *Computer Graphics Forum*, 37(2):415–429, 2018.

[131] Seung-Tak Noh and Takeo Igarashi. Retouch transfer for 3D printed face replica with automatic alignment. In *Proceedings of the Computer Graphics International Conference*, CGI '17, pages 24:1–24:6, New York, NY, USA, 2017. ACM.

[132] Seung-Tak Noh, Kenichi Takahashi, Masahiko Adachi, and Takeo Igarashi. SkelSeg: Segmentation and rigging of raw-scanned 3D volume with user-specified skeleton. In *Proceedings of Graphics Interface 2019*, GI 2019. Canadian Information Processing Society, 2019.

[133] Luke Olsen and Faramarz F. Samavati. Image-assisted modeling from sketches. In *Proceedings of Graphics Interface 2010*, GI '10, pages 225–232, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society.

[134] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L. Davidson, Sameh Khamis, Mingsong Dou, Vladimir Tankovich, Charles Loop, Qin Cai, Philip A. Chou, Sarah Mennicken, Julien Valentin, Vivek Pradeep, Shenlong Wang, Sing Bing Kang, Pushmeet Kohli, Yuliya Lutchyn, Cem Keskin, and Shahram Izadi. Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 741–754, New York, NY, USA, 2016. ACM.

[135] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[136] Sylvain Paris, Hector M. Briceño, and François X. Sillion. Capture of hair geometry from multiple images. *ACM Trans. Graph.*, 23(3):712–719, August 2004.

[137] Sylvain Paris, Will Chang, Oleg I. Kozhushnyan, Wojciech Jarosz, Wojciech Matusik, Matthias Zwicker, and Frédo Durand. Hair photobooth: Geometric and photometric acquisition of real hairstyles. *ACM Trans. Graph.*, 27(3):30:1–30:9, August 2008.

[138] S. Peng, B. Haefner, Y. Quéau, and D. Cremers. Depth super-resolution meets uncalibrated photometric stereo. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2961–2968, Oct 2017.

[139] Hanspeter Pfister, Matthias Zwicker, Jeroen van Baar, and Markus Gross. Surfels: Surface elements as rendering primitives. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 335–342, New York, NY, USA, 2000. ACM Press/ Addison-Wesley Publishing Co.

[140] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4929–4937, June 2016.

[141] Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J. Black. Clothcap: Seamless 4d clothing capture and retargeting. *ACM Trans. Graph.*, 36(4):73:1–73:15, July 2017.

[142] Vivek Pradeep, Christoph Rhemann, Shahram Izadi, Christopher Zach, Michael Bleyer, and Steven Bathiche. Monofusion: Real-time 3d reconstruction of small scenes with a single web camera. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 83–88, Oct 2013.

[143] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, July 2017.

[144] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015.

[145] Bernhard Reinert, Tobias Ritschel, and Hans-Peter Seidel. Animated 3D creatures from single-view video by skeletal sketching. In *Proceedings of the 42Nd Graphics Interface Conference*, GI '16, pages 133–141, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2016. Canadian Human-Computer Communications Society.

[146] Dennie Reniers and Alexandru Telea. Skeleton-based hierarchical shape segmentation. In *IEEE International Conference on Shape Modeling and Applications 2007 (SMI '07)*, pages 179–188, June 2007.

[147] Marc Rioux. Laser range finder based on synchronized scanners. *Appl. Opt.*, 23(21):3837–3844, Nov 1984.

[148] K. M. Robinette, H. Daanen, and E. Paquet. The caesar project: a 3-d surface anthropometry survey. In *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, pages 380–386, Oct 1999.

[149] A. Romanoni, M. Ciccone, F. Visin, and M. Matteucci. Multi-view stereo with single-view semantic mesh refinement. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 706–715, Oct 2017.

[150] Francisco J. Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76:38 – 47, 2018.

[151] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3d model acquisition. *ACM Trans. Graph.*, 21(3):438–446, July 2002.

[152] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015.

[153] Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

[154] N. Savinov, L. Ladický, C. Häne, and M. Pollefeys. Discrete optimization of ray potentials for semantic 3d reconstruction. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5511–5518, June 2015.

[155] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, Apr 2002.

[156] Nico Schertler, Marco Tarini, Wenzel Jakob, Misha Kazhdan, Stefan Gumhold, and Daniele Panozzo. Field-aligned online surface reconstruction. *ACM Trans. Graph.*, 36(4):77:1–77:13, July 2017.

[157] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. Surfelmeshing: Online surfel-based mesh reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019.

[158] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 519–528, June 2006.

[159] J A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[160] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, Jan 2016.

[161] Ari Shapiro, Andrew Feng, Ruizhe Wang, Hao Li, Mark Bolas, Gerard Medioni, and Evan Suma. Rapid avatar capture and simulation using commodity depth sensors. *Computer Animation and Virtual Worlds*, 25(3-4): 201–211, 2014.

[162] Tianyang Shi, Yi Yuan, Changjie Fan, Zhengxia Zou, Zhenwei Shi, and Yong Liu. Face-to-parameter translation for game character auto-creation. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 161–170, October 2019.

[163] Hang Si. TetGen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw.*, 41(2):11:1–11:36, February 2015.

[164] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[165] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3):835–846, July 2006.

[166] Olga Sorkine and Daniel Cohen-Or. Least-squares meshes. In *Proceedings Shape Modeling Applications, 2004.*, pages 191–199, June 2004.

[167] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and Hans-Peter. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pages 175–184, New York, NY, USA, 2004. ACM.

[168] Jürgen Sturm, Erik Bylow, Fredrik Kahl, and Daniel Cremers. Copyme3d: Scanning and printing persons in 3d. In *Pattern Recognition*, pages 405–414, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[169] Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. *ACM Trans. Graph.*, 26(3), July 2007.

[170] Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, and Alexandru Telea. 3D skeletons: A state-of-the-art report. *Computer Graphics Forum*, 35(2):573–597, 2016.

[171] Kenshi Takayama, Ryan Schmidt, Karan Singh, Takeo Igarashi, Tamy Boubekeur, and Olga Sorkine. Geobrush: Interactive mesh geometry cloning. *Computer Graphics Forum*, 30(2):613–622, 2011.

[172] J. Rarael Tena, Moshe Mahler, Thabo Beeler, Max Grosse, Hengchin Yeh, and Iain Matthews. Fabricating 3d figurines with personalized faces. *IEEE Computer Graphics and Applications*, 33(6):36–46, Nov 2013.

[173] Christian Theobalt, Edilson de Aguiar, Carsten Stoll, Hans-Peter Seidel, and Sebastian Thrun. *Performance Capture from Multi-View Video*, pages 127–149. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[174] Jean-Marc Thiery, Émilie Guy, and Tamy Boubekeur. Sphere-meshes: Shape approximation using spherical quadric error metrics. *ACM Trans. Graph.*, 32(6):178:1–178:12, November 2013.

[175] Anastasia Tkach, Mark Pauly, and Andrea Tagliasacchi. Sphere-meshes for real-time hand modeling and tracking. *ACM Trans. Graph.*, 35(6):222:1–222:11, November 2016.

[176] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 311–318, New York, NY, USA, 1994. ACM.

[177] Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. SemanticPaint: Interactive 3D labeling and learning at your fingertips. *ACM Trans. Graph.*, 34(5):154:1–154:17, November 2015.

[178] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Computer Vision – ECCV 2018*, pages 55–71, Cham, 2018. Springer International Publishing.

[179] A. Webanck, Y. Cortial, E. Guérin, and E. Galin. Procedural cloudscapes. *Computer Graphics Forum*, 37(2):431–442, 2018.

[180] Yichen Wei, Eyal Ofek, Long Quan, Heung-Yeung Shum, and Heung-Yeung Shum. Modeling hair from multiple views. *ACM Trans. Graph.*, 24(3):816–820, July 2005.

[181] Chao Wen, Yinda Zhang, Zhuwen Li, and Yanwei Fu. Pixel2mesh++: Multi-view 3d mesh generation via deformation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[182] Thomas Whelan, Hordur Johannsson, John J. Leonard, John Mcdonald, Thomas Whelan, Hordur Johannsson, Michael Kaess, John J. Leonard, and John Mcdonald. Robust tracking for real-time dense rgb-d mapping with kintinuous. Technical report, 2012.

[183] Thomas Whelan, Michael Kaess, Hordur Johannsson, Maurice Fallon, John J. Leonard, and John McDonald. Real-time large-scale dense rgb-d slam with volumetric fusion. *The International Journal of Robotics Research*, 34(4-5):598–626, 2015.

[184] Robert J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19(1):139 – 144, 1980.

[185] C. Wu, B. Wilburn, Y. Matsushita, and C. Theobalt. High-quality shape from multi-view stereo and shading under general illumination. In *CVPR 2011*, pages 969–976, June 2011.

[186] Changchang Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 127–134, June 2013.

[187] Chenglei Wu, Derek Bradley, Pablo Garrido, Michael Zollhöfer, Christian Theobalt, Markus Gross, and Thabo Beeler. Model-based teeth reconstruction. *ACM Trans. Graph.*, 35(6):220:1–220:13, November 2016.

[188] Shihao Wu, Wei Sun, Pinxin Long, Hui Huang, Daniel Cohen-Or, Minglun Gong, Oliver Deussen, and Baoquan Chen. Quality-driven poisson-guided autoscanning. *ACM Trans. Graph.*, 33(6):203:1–203:12, November 2014.

[189] Jianxiong Xiao and Yasutaka Furukawa. Reconstructing the world's museums. *International Journal of Computer Vision*, 110(3):243–258, Dec 2014.

[190] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1696–1704. Curran Associates, Inc., 2016.

[191] Kangxue Yin, Hui Huang, Hao Zhang, Minglun Gong, Daniel Cohen-Or, and Baoquan Chen. Morfit: Interactive surface reconstruction from incomplete point clouds with curve-driven topology and geometry control. *ACM Trans. Graph.*, 33(6):202:1–202:12, November 2014.

[192] L. Yu, S. Yeung, Y. Tai, and S. Lin. Shading-based shape refinement of rgb-d images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1415–1422, June 2013.

[193] Ming Zeng, Fukai Zhao, Jiaxiang Zheng, and Xinguo Liu. Octree-based fusion for realtime 3d reconstruction. *Graphical Models*, 75(3):126 – 136, 2013. Computational Visual Media Conference 2012.

[194] Meng Zhang, Menglei Chai, Hongzhi Wu, Hao Yang, and Kun Zhou. A data-driven approach to four-view image-based hair modeling. *ACM Trans. Graph.*, 36(4):156:1–156:11, July 2017.

[195] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov 2000.

[196] Zhengyou Zhang. Microsoft kinect sensor and its effect, Feb 2012.

[197] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Trans. Graph.*, 33(4): 155:1–155:10, July 2014.

[198] Michael Zollhöfer, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Shading-based refinement on volumetric signed distance functions. *ACM Trans. Graph.*, 34(4):96:1–96:14, July 2015.

[199] S. Zuffi, A. Kanazawa, D. W. Jacobs, and M. J. Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5524–5532, July 2017.

[200] Matthias Zwicker, Mark Pauly, Oliver Knoll, and Markus Gross. Pointshop 3d: An interactive system for point-based surface editing. *ACM Trans. Graph.*, 21(3):322–329, July 2002.