NAIST-IS-DD1461015

Doctoral Dissertation

Novel View Synthesis and Augmented Reality for Assisting Human Action Learning

Fabian Lorenzo Dayrit

March 16, 2017

Department of Information Science
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Fabian Lorenzo Dayrit

Thesis Committee:

Professor Naokazu Yokoya	(Supervisor)
Professor Hirokazu Kato	(Co-supervisor)
Associate Professor Tomokazu Sato	(Co-supervisor)
Associate Professor Yuta Nakashima	(Co-supervisor)

Novel View Synthesis and Augmented Reality for Assisting Human Action Learning*

Fabian Lorenzo Dayrit

Abstract

When people wish to learn an action, such as in sports or in dance, for example, the most common way to do so is by imitating someone else performing the action. This can take one of two forms: either the learner observes a real, in-person teacher, or the learner watches a recording of the teacher performing the action. In-person observation allows the learner to view the action from any point of view, but it requires the teacher to be there. The video may be watched separate from the teacher, but it is limited to the original capturing point of view. We want to combine the advantages of these two by creating a new way to view such actions.

This study revolves around capturing human actions using depth cameras and rendering the actions from a novel viewpoint, focusing on the motion of the actions and not on the location or context. We call such novel views of actions, *reenactments*. We wish to use reenactments to help users comprehend and learn actions.

We explore practical ways of capturing and rendering reenactments that may be done using consumer depth cameras at home. The challenge is in adequately representing unseen areas and in defining consistent correspondences on the subject's body across the motion sequence. This thesis proposes two methods to represent reenactments: by a set of rigid body parts, and by a deformable statistical body model. For both of these methods, we have implemented an application and have conducted a user study to evaluate the reenactment quality, as well as the application's effectiveness, ease of use, and appeal.

*Doctoral Dissertation, Graduate School of Information Science,
Nara Institute of Science and Technology, NAIST-IS-DD1461015, March 16, 2017.

The rigid body parts method relies on approximating the shape of the subject's body parts with a 3D mesh for each. This method is straightforward, but is imprecise; we use view-dependent texture mapping to compensate for inaccuracies. We have implemented a mobile action viewer application that displays the reenactment using augmented reality. During our evaluation, users found it easier to comprehend ambiguous actions using our viewer than with the original video. It was confirmed that they were also able to compare reenactments with the real world more intuitively.

The deformable statistical body model method fits a model of the human body to the depth maps of the motion sequence. This method is more accurate, but requires plenty of training data. We have implemented a magic mirror action-learning system which directly helps users to learn actions by displaying the reenactment on top of a mirror of themselves. Using a skeleton tracker, the system is able to display a view of the reenactment that matches the user's body orientation, facilitating intuitive comparisons.

The contributions of this thesis can be summarized as follows. First, we present multiple methods of synthesizing novel views of an action using consumer RGB-D cameras. We also propose and implement several applications of our reenactments for the purpose of action learning. Finally, we demonstrate the value of reenactments, the applications, and the use of augmented reality for learning in general, by conducting user studies.

Keywords:

novel view synthesis, augmented reality, action learning, computer graphics, 3D reconstruction

Contents

List of Figures

1 Introduction

For physical activities such as dancing, sports, or martial arts, it is important for practicioners to practice and learn the correct motions in order to perform them properly. The best way to learn these motions is by copying a teacher who is performing them, since it may be hard to describe and understand descriptions of these kinds of motions accurately. The teacher may perform the motion in-person, together with the learner, and this is how motion learning was originally done. However, since the development of video, the teacher's motions may be recorded on camera and watched anytime, anywhere. This has become more prevalent in recent years due to the rise of online video sharing, and now relevant video demonstrations can be found quickly and easily.

However, these kinds of videos have a weakness in that they may only be viewed from a single viewpoint: the original capturing viewpoint. This means that if there are any ambiguous or hard-to-understand motions in the video, the learner will not be able to clarify the motion by changing his or her perspective. In person, it is much easier to view a different perspective, but this way is not as easily accessible as viewing videos.

Now, rendering a scene from an arbitrary perspective is in the domain of computer graphics. However, the information of the scene may not be readily available. Thus, this information must somehow be extracted.

Since we are capturing the performance of a human, we must do some sort of human motion capture. There are several existing methods for human motion capture: mechanical armatures that measure the performer's joint angles, magnetic and optical marker-based methods, and computer vision-based methods [36]. These methods provide varying degrees of ease-of-use and accuracy. For example, the motion capture systems exactly measure joint angles but heavily restrict motion. Marker-based methods provide more freedom, as the performer

only has to wear the considerably lighter markers, and these have a high degree of accuracy [51,56]. Purely computer vision-based methods are the most natural, as they allow capturing the scene "as-is," but may have less accuracy.

However, we do have further considerations. For one, we must also reconstruct the shape of our performer. Most systems do this by manually constructing a model of the performer in an offline process, but certain computer vision-based methods are able to do this using video and/or depth data. Another consideration is that we place great importance on accessibility for the general user, who is unable to build and complicated setups but may be able to purchase, for example, a consumer Microsoft Kinect consumer RGB-D camera. Thus, for this work, we limit ourselves to computer vision methods that utilize RGB-D data of human performers. This use of computer vision for representing a real scene from an arbitrary perspective is what we call novel view synthesis (NVS), the intersection of computer graphics and computer vision [18].

NVS systems attempt to synthesize a view of a scene from a previously-uncaptured viewpoint, making use of those viewpoints that were successfully captured. Such systems have various requirements, using inputs such as different views or depth images of the scene and so on. This thesis in particular focuses on NVS of moving humans, since motion cannot be conveyed with a static scene. Since this assumption is present, we consider that the most efficient method would be to represent our teacher with a human body model. This leads us into the fields of human motion capture and human shape reconstruction.

Human motion capture and human shape reconstruction, together, involve the processes of representing the human body and estimating human motion. Several representations of the human body exist. One of the more commonly used representations is a graph of rigid body parts, each with a varying degree of freedom as well as a number of connected body parts. On top of this, human statistical body models are able to generate plausible human body meshes from a relatively small set of attributes. Such models may be fit to the limited depth information that a consumer RGB-D camera provides in order to fill in missing regions with the most statistically probable shape. Due to limited view, we may not be able to reconstruct the actual shape or texture of our teacher. Thanks to the statistical methods, however, we may at least be able to generate plausible models that align

with the learner's imagination. In this case, the learner is able to comprehend the motion as being performed by a plausible model of the teacher. We call such generated motion sequences *reenactments*.

Reconstructing human motion and generating reenactments is only half of our work. A practical application requires an intuitive, easy-to-use user interface. Traditional mouse and keyboard input, for example, is well-suited to 2D applications, but is insufficient and unintuitive to use for 3D. For our own applications, we implement augmented reality (AR) techniques in order to more naturally convey motion.

An AR system presents a virtual object, such as our reenactments, in a real world environment, with real world context, in real time [85]. A large part of AR is interaction with the virtual object. For example, we consider a handheld AR virtual object viewer. Such a viewer would include a camera, allowing the learner to "see through" to the real world. The AR portion would render the virtual on the image of the real world, modifying the perspective appropriately as the viewer moves around, as if it were actually situated in the environment.

Finally, we propose the use of this technology to help users learn actions. One definition of learning is a change, resulting from practice, in the learner's capability to respond [74]. From the literature on motor learning, we adopt kinematic knowledge of results (KR) as a way to facilitate learning of physical actions. One way to present kinematic KR is by showing the user the pattern of his or her response sequence, at the same time showing the ideal response sequence [1]. The user can thus directly see the difference between the two and treat it as an error.

Towards this goal, we propose two AR applications. One is a handheld application for viewing reenactments on top of the original capturing location. We synchronize the capturing camera and the reenactment viewer's camera in order to do this, for the purpose of giving context to the reenactment. Using this viewer, users are able to see the difference between a real person performing the action, and a reenactment, which provides kinematic KR. The second AR application concerns users more directly. We propose a "magic mirror" application that overlays the reenactment's motions on top of the user's own body and presents it in a mirror-like fashion. As the user turns his or her body, the reenactment turns in the same way, thereby providing an easy method of comparison between

the user's motions and the reenactment's, fulfilling the need for kinematic KR. For both of these methods, we perform a user study on the effectiveness of this application-provided kinematic KR.

The rest of this thesis is organized as follows. Chapter 2 describes other works related to NVS for moving humans and AR for learning, contrasted with the contributions of this study. Chapter 3 and 4 describe our methodology: Chapter 3 describes our rigid body part-based reenactment method using view-dependent texture mapping (VDTM) for texturing, while Chapter 4 describes our statistical, nonrigid body model-based reenactment method. Finally, chapter 5 summarizes this thesis.

2 Related work

This work aims to generate arbitrary views of real human motion and display it in an intuitive manner, with the eventual goal of facilitating motor learning. We use novel view synthesis (NVS) techniques in order to capture human motion and shape from a small number of cameras, allowing the motion sequence to be rendered from an arbitrary point of view. In order to display the motion intuitively, we use augmented reality (AR) techniques. These techniques aim to replicate the natural motions of looking through a lens or using a mirror, increasing the ease of use. Finally, we apply concepts from the theory of motor learning in order to evaluate whether we can use such AR systems to help users learn actions.

2.1 Novel view synthesis

NVS is a field of research that aims to generate novel views of a scene from an arbitrary point of view, a combination of computer graphics and computer vision. Since our focus for this work is on views of human motion in particular, we introduce methods that are able to do this, from image-based NVS that can handle general scenes, to works that assume a human subject in order to build a model for reconstruction.

2.1.1 Image-based novel view synthesis

Image-based NVS systems basically use multiple captured images of a scene in order to generate a view of that scene from a viewpoint that is different from what was captured. Among the first of these systems used the image-based visual hull (IBVH) [59]. These build a 3D model of an object by capturing it from multiple viewpoints, generating the model from the sillhouettes in each image. Using a

single camera, as described in the paper, will only let one generate a static 3D model, as the camera must be moved around to cover different views and the object must remain static while doing so. Würmlin et al. [92], however, set up multiple cameras to capture a single subject, extending the technique to cover moving objects.

Similar to these are the systems based on voxels and marching cubes [22, 55]: Matsuyama and Takai [58] and Starck, Miller, and Hilton [80]. These systems first generate, using multiple cameras, a voxel representation of the subject, and then convert it into a 3D mesh using the marching cubes algorithm. The mesh is colored using view dependent textures in [58], while [80] blends the RGB frames from each camera into one integrated texture.

Another class of these systems interpolates captured views in order to generate novel ones. Zitnick et al. [99] and Karsten et al. [64] segment frames into layers and then blend the layers captured from two cameras in order to render an image from a virtual viewpoint that is somewhere in between the two cameras.

Other systems use a combination of depth and RGB data in order to generate free-viewpoint images. Dai and Yang [23] capture and render a subject in real time, from an arbitrary viewpoint, using multiple RGB-D cameras. Each camera's foregound layer is merged to produce the final result. Alexiadis, Zarpalas, and Daras [2] also capture a dynamic scene using multiple RGB-D sensors. Each camera's output is converted into 3D meshes and merged, taking care to remove redundant polygons.

Tong et al. [83] propose a setup using 3 RGB-D cameras to scan one subject, with two at the front and one in the back, in an arrangement such that they do not interfere with each other. The subject is then rotated and the depth input merged in order to generate a mesh.

Dou, Fuchs, and Frahm [26] use an RGB-D camera to reconstruct a moving person by first merging multiple point clouds with nonrigid registration to create a fused 3D model, then tracking that model.

Most recent methods of this type employ variants of the signed distance field technique, which is basically a registration problem of multiple depth maps and represents 3D shape using the zero-level iso-surface of the signed distance field. This approach was originally designed for rigid scenes. One of the more well-

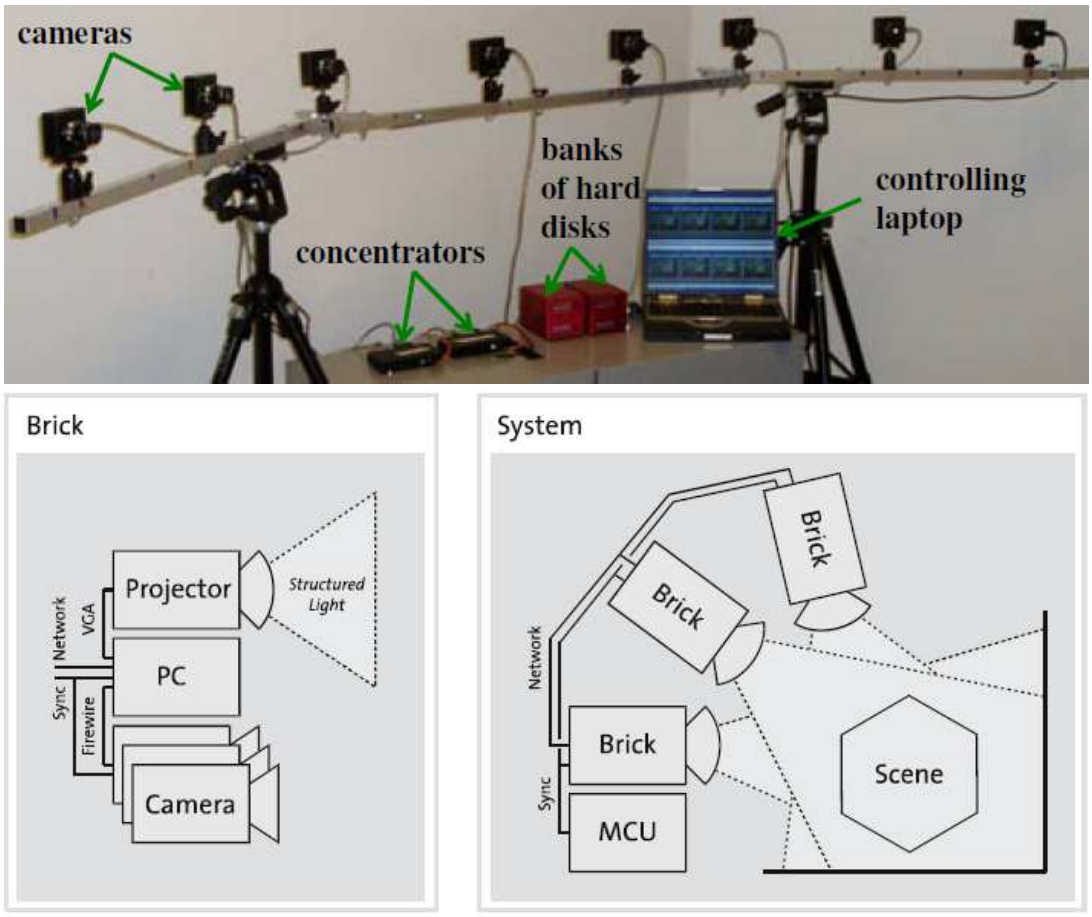


Figure 2.1: Top: Zitnick et al. [99]'s capturing setup. Bottom: Waschbüsch et al. [89]'s scene capture system using multiple cameras.

known examples is KinectFusion [45]. This approach was later extended to handle non-rigid objects by describing the deformation of objects with transformations of signed distance field [27, 28, 44, 66]. These methods can generate surprisingly high quality 3D shapes, but may lack tracking stability with regards to, e.g., occlusions.

Depth data also can be converted into 3D point clouds. Point clouds from multiple viewpoints can be integrated to form a representation of the entire scene. Waschbüsch et al. [89] capture an RGB-D stream from projector-camera combinations around the scene (Fig. 2.1 (bottom)), and then convert the color and depth data into 3D points, which they then integrate. Kainz et al. [47] also make

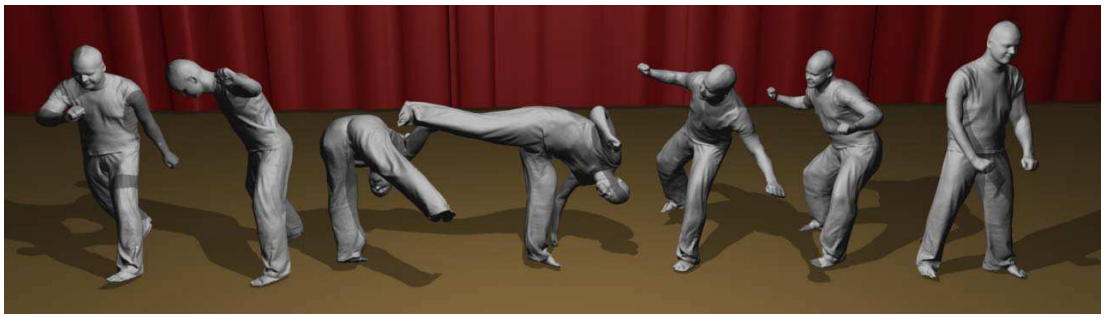


Figure 2.2: Detailed model and motion of a subject captured by De Aguiar et al. [24]

use of multiple RGB-D streams. They use a combination of point cloud integration and IBVH. The point cloud method usually produces noisy edges and IBVH does not detect concavities, but an intersection of the two produces a model with clean edges and proper concavities.

Zollhöfer et al. [100] use an RGB-D camera and capture deformations in the subject in real time. To do this, they first create a template mesh in an initialization step by capturing the subject from different angles, i.e. with various rigid transformations, then perform rigid and non-rigid fitting in real time.

2.1.2 Model-based human shape reconstruction

In contrast to the general approach, the model-based approach uses prior knowledge on the object, i.e. the human teacher, to be captured to facilitate entire object reconstruction, and most existing methods that take this approach are designed for human body reconstruction.

One frequently used model is a 3D mesh with an underlying pose. The mesh is created and fit to the pose using skinning. From there, this mesh-and-pose model can be fit to the motion sequence. Fitting a 3D mesh to a monocular motion sequence is difficult, so most works rely on multiple camera setups or depth cameras. For example, Carranza et al. [18] first initialize a general 3D model to the body shape of a subject (Fig. 2.3). Then, by capturing that subject using multiple cameras, they are able to obtain sillhouettes from multiple viewpoints over multiple frames. For each frame, they then find the pose of the 3D model

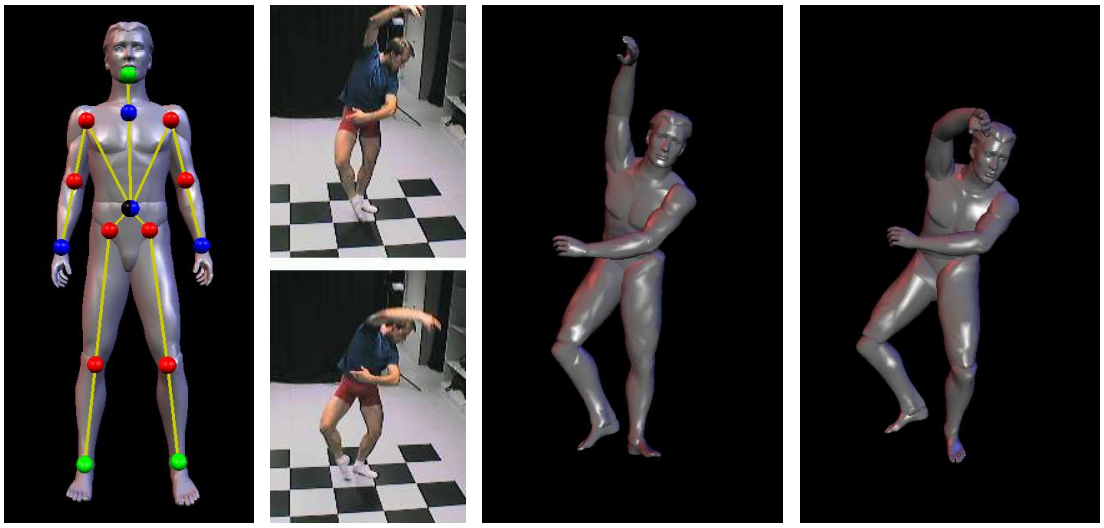


Figure 2.3: Human pose estimation and free-viewpoint image generation system by Carranza et al. [18] First column: general 3D model. Second column: two frames of input. Third and fourth column: 3D model fit to the input.

that fits to each sillhouette. De Aguiar et al. [24], in addition to using RGB frames, also use a laser scanner to construct a 3D mesh model of the subject in advance. They then capture the subject's motion and use keypoints in each frame to transform the model. In order to locate the 3D positions of the keypoints, they capture from multiple cameras simultaneously. Using this method, they are able to capture a detailed mesh with motion (Fig. 2.2). Hofmann and Gavrila [42] propose a multicamera method to do this in complex, dynamic environments by a combination of background modeling, volume carving, and finally culling unsuitable voxel volumes. Similar other works [32, 86] use a visual hull of the multiple cameras.

Ganapathi et al. [33] and Baak et al. [9] propose a real time mesh-and-pose-based method of estimating human motion using a single depth camera. Cagniart, Boyer, and Ilic [16, 17] forgo the pose parameters and simply deform the mesh using correspondences from multiple cameras. These methods work well, but having to use an initial mesh increases the burden on the user of the system. Some may still work with a generic template mesh, but performance may suffer.

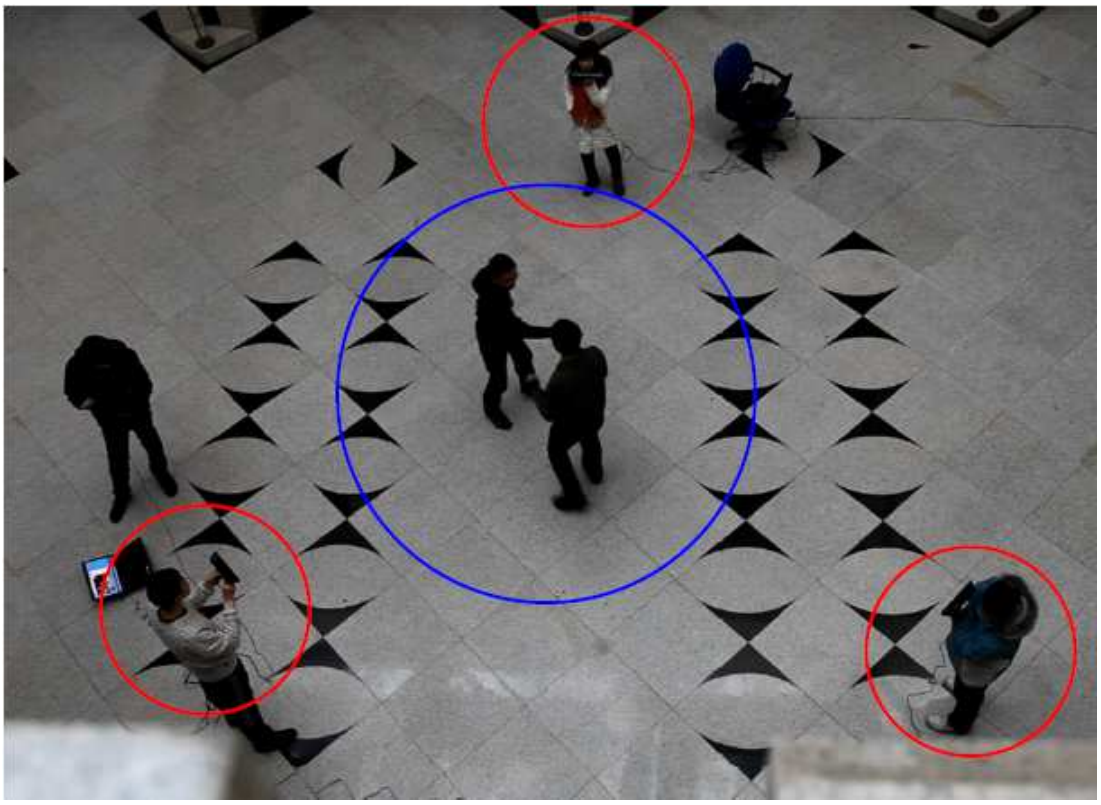


Figure 2.4: Capturing two users with three handheld Kinects simultaneously [95].

Shotton et al. [79] also developed an algorithm that estimates human motion in real time with a single depth camera, but requiring no initialization. This is the algorithm that the first version of the Microsoft Kinect RGB-D camera uses. In [95], three Kinects are used in order to generate free-viewpoint images of human motion (Fig. 2.4). Each Kinect captures a point cloud of the scene, similar to the systems above. In order to correctly integrate the point cloud, they use a number of constraints, such as the extrinsic parameters of each Kinect, and the pose of the subjects. Using this method, they are able to generate free-viewpoint image sequences of up to two subjects. In [57], on the other hand, only a single Kinect is used. They build a 3D model of a subject using voxels and apply the subject's motion to the model in order to generate a free-viewpoint image sequence. To accomplish this, they capture the subject's pose in each frame, and then assign voxels to defined body parts.

Stoll et al. [81] use a body model made up of sums of Gaussians that may be computed in a preprocessing step before proceeding to estimate the pose of the subject from multiple cameras. Liu et al. [54] and Rhodin et al. [72] use the mesh-and-pose model mentioned above, but additionally deform their template mesh to fit the visual hull from multiple cameras. Similar systems [96, 97] also exist based on the output of single RGB-D cameras.

Other methods use a statistical parametric model of human body shape in order to fit arbitrary body shapes, such as SCAPE [6], S-SCAPE [46], the work of Wuhrer et al. [91], and TenBo [20]. These methods in particular describe plausible body shapes using pose and shape parameters that control the human body's attributes like weight, height, etc. These parameters and the way they affect the model are calculated statistically, from a mesh dataset such as [37]. Some methods that adopt the model-based approach basically fit one of these models to a point cloud of depth observations. For example, Weiss et al. [90] proposed a reconstruction system using SCAPE, where the fitting process is initialized with skeletal tracking results. Yang et al. [93] use S-SCAPE as an underlying model to reconstruct subjects wearing loose or baggy clothing. Bogo et al. [14] use SCAPE with several modifications including multi-resolution mesh fitting and using displacement maps for finer details. Due to the modification of multiple resolution meshes, their system no longer relies on skeletal tracking, which is error-prone.

2.2 Augmented reality

Our system aims to display a virtual subject with AR. An AR system aims to present a virtual object, e.g. some textual information or a rendering of a real person, in a real world environment, in real time [7, 85]. AR has plenty of applications in medicine, manufacturing, entertainment, etc., but it is especially suited to learning [29], as well as specifically helping its users learn motions. AR can be delivered through multiple kinds of media, such as head-worn displays, handheld devices, or direct projection [8]. Here, we pay special attention to the "glasses" metaphor of AR [75], in which a user views a lens in front of his or her eyes, which may be further classified into head-mounted, handheld, and projected

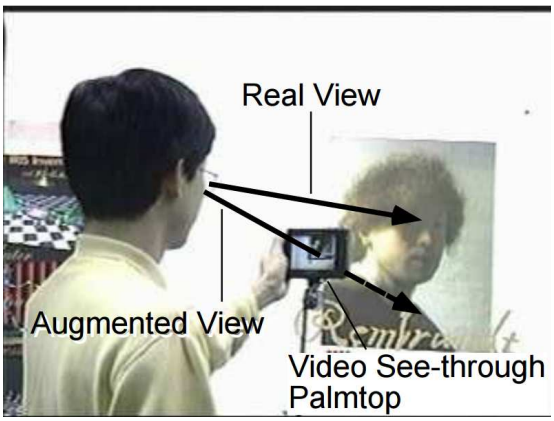


Figure 2.5: The Navicam system [70], one of the first handheld AR systems. It consists of a display with a gyroscope and a camera for video see-through.

AR. In particular, we focus on handheld AR, due to its portability and ease of use. With this metaphor, the device simulates a lens through which the user views the world, as seen for example in Fig. 2.5. The device in this case implements an optical or see-through display using a camera on the back side, augmenting the captured video stream in real time. AR systems implementing this metaphor often use some sort of pose estimation to calculate how the user is holding the device, which should have an effect on how the system renders its output. Thus, users are allowed to intuitively control the viewpoint with the way they hold the device. We also reference AR with the "mirror" metaphor [75], an AR metaphor that aims to simulate an augmented mirror. With this metaphor, on the other hand, the device simulates a mirror using a display and a camera, both facing the user, for example as in Fig. 2.6. AR systems implementing this metaphor will often detect the user and render objects in relation to the user. For example, a virtual clothes try-on system might render clothes on top of users to let them see how the clothes look on them. This metaphor is more suited to applications where users want to view how something might affect themselves, and is intuitive due to the similarity with a regular mirror.

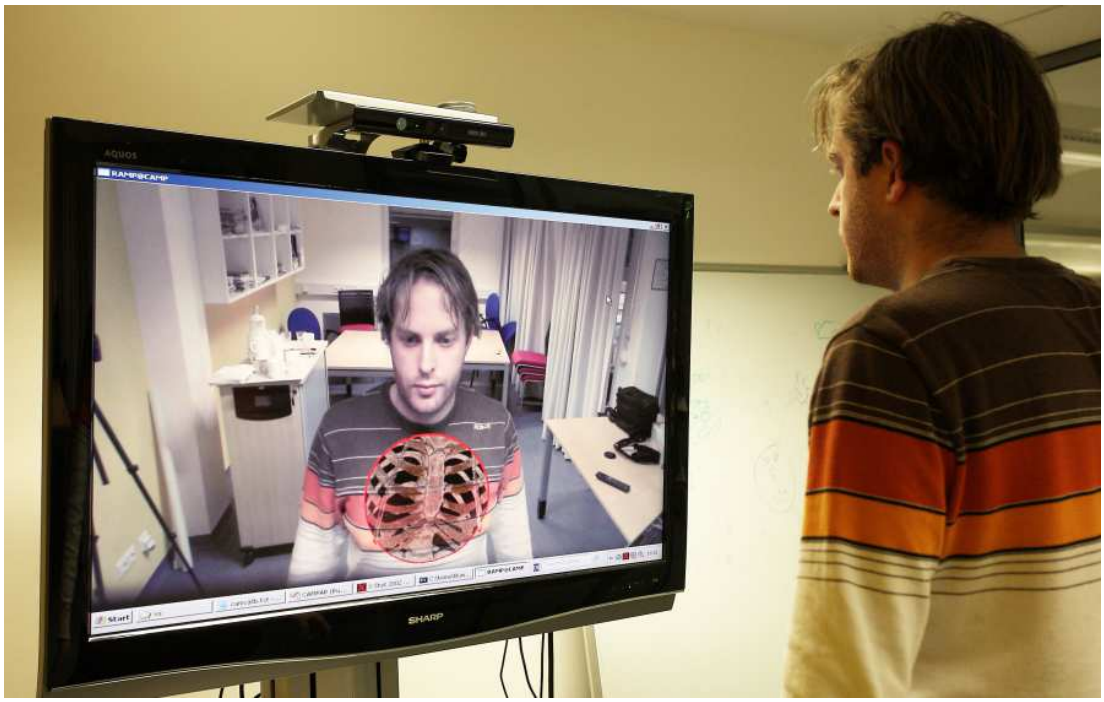


Figure 2.6: The mirracle system [13], a mirror AR system that helps users learn about anatomy.

2.2.1 Learning through augmented reality

Learning is commonly defined as a change that results from practice or experience in the capability of the learner to respond. In this thesis, we focus specifically on motor learning, or the learning of physical actions. Knowledge of results (KR), widely regarded as a critical step in learning, is defined by Salmoni et al. [74] as verbal, terminal, augmented feedback. Verbal means able to be verbalized or expressed in language. Terminal means that the feedback comes after the action, as opposed to concurrently with the action. Augmented means that the feedback is explicit and direct. However, Adams [1] puts forth the idea of *kinematic knowledge of results*, with 3 different ways of achieving this:

1. Showing the subject his or her motion sequence. Error is inferred.

2. Showing the subject his or her motion sequence, along with the target motion sequence. Error is directly, implicitly shown.

3. Giving the subject explicit error information for divisions of the motion sequence.

In this thesis, we are mostly interested in way 2. In our user studies, we use AR to directly compare the motions of real humans with our target reenactments.

AR has many aspects that can be used to enhance learning. As an example, commonly, teachers use props in order to demonstrate lessons or convey meanings. By using AR, students can interact with tangible virtual objects in real world contexts [12]. Several flashcard-like AR systems for memory tasks augment entries with the corresponding 3D object for easy recall [87].

Kancherla et al. [48] describe a system that allows students to visualize the underlying skeleton of a human body, e.g. the 3D position of a patient's bones. This demonstrates the power of AR to convey 3D information in context.

Caarls et al. [15] (Fig 2.7) use AR to enhance museum displays by allowing users to interact with history, showcasing the interactive nature of AR. Using these displays, they were able to attract and hold people's attention, drawing interest to the content.

Santos et al. [76] propose and evaluate a handheld AR system that aims to allow users to visualize virtual objects in context. Their findings were that the system conferred no radical advantages over simply viewing the virtual objects in terms of realism, depth perception, and visibility. However, as AR offers a form of experiential learning, or learning from experience [52], these results imply that AR can become a valid alternative for teachers and students most suited to experiential learning.

Another strength of AR is that it can have a large positive impact on users' physical skills [98]. Tsuchida, Terada, and Tsukamoto [84] propose a learning support system specifically for dancers in a formation. In the place of a missing dancer, they used a self-propelled robot with a screen displaying the appearance of the dancer (Fig. 2.8). The robot moves in space according to how the dancer would have moved. Users who tried the system danced more accurately, i.e. closer to the actual trajectory, with the robot than without.

The system proposed by Henderson and Feiner [39, 40] shows the user instructions on how to do a specific procedure, by way of arrows and labels in 3D space attached to key objects (Fig. 2.9). Users wore an optical see-through head-

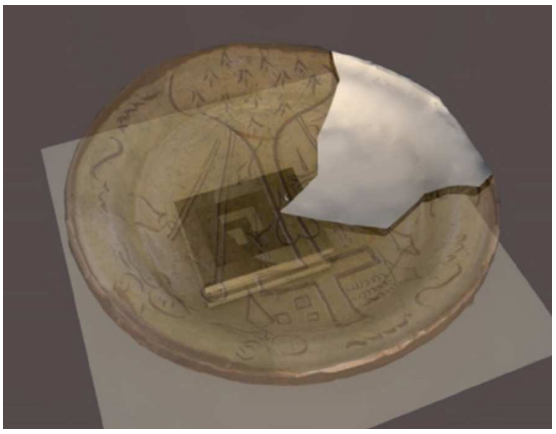


Figure 2.7: Visualizing history with AR [15].

mounted display, and the instructions were overlayed on top of their view. Users who were surveyed preferred to use the proposed system over an instructional video with similar content displayed on an LCD monitor.

Yang and Kim [94] developed a first-person motion-training system which made use of the "ghost" metaphor. The ghost in the system is a semitransparent 3D model which the user must imitate. Being able to perfectly follow the ghost means that the user has performed the motion correctly, but we consider that the first-person view means that the user may not be able to easily see the ghost's actions, especially if the user and the ghost are on the same spot. Nevertheless, we make much use of the ghost metaphor in this thesis.

2.2.2 Handheld augmented reality

Handheld AR systems offer portability, comparative ease of use, and availability. These, however, come at the cost of power, which may have a critical impact on the performance and usability of the system. A secondary challenge that all handheld AR systems face is that of tracking the pose of the device. Some rely on non-visual data, such as from gyroscopic sensors. Many rely on color-coded stickers, AR markers, and other fiducial objects. Others make use of visual features of the environment itself.

Amselem's work [3] makes use of a handheld display with a Polhemus tracker to estimate the display's position and orientation. Similarly, Rekimoto's Transvi-

Figure 2.8: Dancing with a robot [84]. The robot carries a screen that simulates the appearance of a missing dancer.

sion [69] and Navicam [70] (Fig. 2.5) consist of a screen with an attached gyroscope, with a separate workstation. In these systems, the display itself does not process information; instead, the appropriate output is streamed via a wired connection based on the current tracking status. This limits these systems' usable environment. The mPARD system [68] relaxes this limitation somewhat, replacing the wired connection with a wireless one. Aside from this, other works aim to reduce the load of the workstation by splitting the processing ([34, 67, 78]). However, they still do not completely eliminate the reliance on an external workstation.

With that said, these are comparatively early works. Due to the recent advances in smartphones and tablet PCs, several systems and frameworks have managed to build working systems on purely mobile devices that provide adequate performance ([30, 41, 43]).

Fiducual markers are another option for tracking, taking the square marker introduced in [71] as an example. Wagner and Schmalstieg [88] [77] developed a framework for handheld AR applications by porting ARToolkit [49], one of the

Figure 2.9: The AR instruction system proposed by Henderson and Feiner [40]. The AR arrows instruct the user on how to move in 3D space.

more widely distributed marker tracking systems for desktop PCs, to the Windows CE mobile operating system. They demonstrated their framework's practical value by implementing several educational games. Others developed their own low-cost marker tracking system specifically for consumer cellphones [63, 73].

2.2.3 Mirror augmented reality

The mirror metaphor presents a screen as a mirror reflection facing the user, which is then augmented with virtual objects. This metaphor is well-suited to applications which focus on the user's own body and immediate environment, which are reflected in the mirror. A common application of this metaphor is in virtual try-on systems, applications that allow users to virtually try on clothes without having to handle the physical objects ([35, 38, 61]). Fiala developed a framework for such systems that utilizes fiducial markers on users' bodies [31] (Fig. 2.10).

In order for users to visualize how their own muscles work, Murai et al. [65] developed a system to display muscles on top of users of a mirror system, to make it easier for users to observe which muscle was [65]. For educational purposes, Blum et al. [13] developed a mirror AR system for learning anatomy (Fig. 2.6). The user stands in front of the system and internal organs and skeletons appear on top of the appropriate place on the user's body. Meng et al. [60] developed a similar system, with an emphasis on anatomical accuracy. Bauer et al. [10, 11] contribute another such system, calibrating the skeleton bones and organs to individual users and e.g. maintaining consistent bone lengths, for anatomical

Figure 2.10: Fiala's magic mirror system [31]. A user wearing AR markers is reflected and augmented virtually by the mirror.

plausibility (Fig. 2.11).

Mercrier et al. [62] created a mirror system called Mind-Mirror, which superimposes a virtual brain onto the user's head. This virtual brain displays the user's brain activity accurately by using EEG sensors and is placed in the correct location thanks to skeleton tracking.

Kwon and Gross [53] developed a motion learning system that displayed the motion on a screen, recorded the learner's own motions, and compared them. The system itself explicitly gave visual feedback based on the learner's motions. We consider, however, if the same sort of system were to be implemented as an AR mirror, for example by overlaying the teacher onto the student. In this case, even without the system giving explicit feedback, learners can find out by themselves the exact region they are making mistakes, decreasing the risk of miscommunication. The YouMove system [4] is one such application. It first records and tracks the motion of a teacher using an RGB-D camera. Afterwards, learners stand in front of the augmented mirror, (Fig. 2.12) and the system overlays the

Figure 2.11: Superimposing e.g. 3D skeletons, muscles, and internal organs on top of learners' bodies in order to facilitate anatomy education [10].

subject's motions on their reflection. In this way, viewers can more easily copy difficult motions. The system also provides a comparison between the subject and a viewer using 3D stick figures, which the viewer can rotate in order to view the motions from different directions.

Mixed-reality physical therapy systems are related to motion learning, as the system is guiding a user's motion [50]. Tang et al. [82] develop a system for assisting physical therapy at home by demonstrating motions to users (Fig. 2.13). Recommended motions are displayed as wedge-shaped overlays on top of a mirror display, on the side of which a top view is included.

2.3 Contributions of this thesis

This thesis proposes multiple methods for capturing, synthesizing, and viewing reenactments (see Table 2.1), as well as two applications of reenactments towards action learning. Our main contributions are:

- We introduce the concept of *reenactments*, free-viewpoint images of sequences of human motion. Reenactments potentially have a wide range of applications, including watching performances, recording sports, etc., but

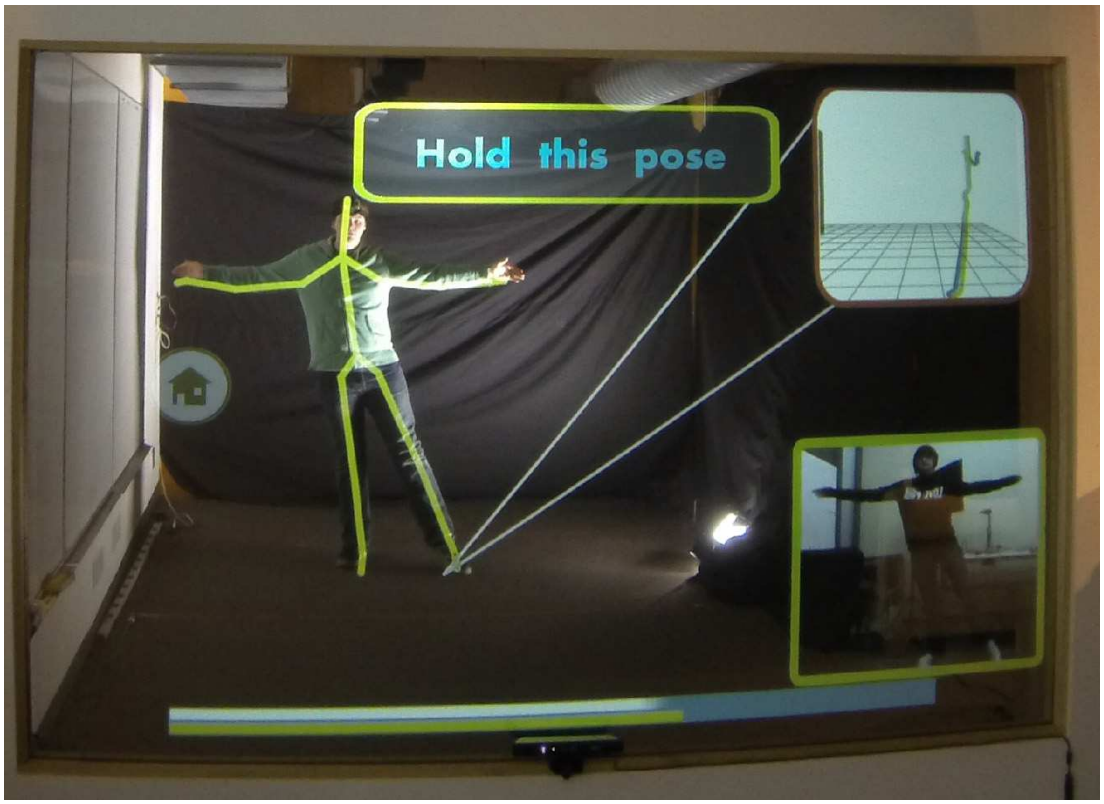


Figure 2.12: YouMove, AR system using the mirror metaphor [4]. The ideal motions are overlayed on top of a mirror image of the user, reflected on a screen.

in this thesis we focus on learning motions. Our proposed systems implement reenactments with AR techniques, which lets users intuitively choose the viewpoint, and displays the reenactments in a natural way in the same location that the actor was captured. This kind of presentation through AR is novel, when compared to the existing methods of NVS.

- We propose a novel method for NVS utilizing only a single RGB-D camera, representing the human body as a set of rigid body parts. We shape each body part using the depth images and exploit skeleton tracking in order to represent the motion. We texture each body part individually using VDTM, taking into account the pose of the person in each captured RGB image as well as the camera positions.

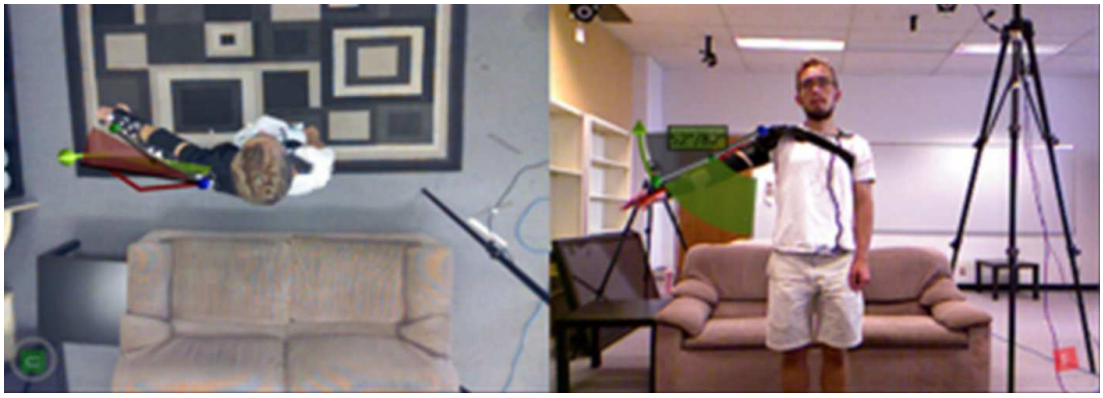


Figure 2.13: The Physio@Home system [82], a physical therapy system. Users must follow the motion of the colored wedges, with the front view on the right and the top view on the left.

- We propose a second novel method for NVS that uses a statistical nonrigid body model to represent the human body. We fit the model's parameters to the entire set of depth images, which gives us the model's shape and motion. On the other hand, given particular shape and motion parameters, the model generates an appropriately deforming mesh, which accounts for missing regions in the observation. This method is flexible and may use one or more RGB-D cameras.

- We implement a handheld AR reenactment system that allows users to view reenactments on a handheld device. We quantitatively evaluate the performance of our system in terms of its effectiveness in learning specific poses. We show that by viewing the AR reenactments, users are more easily able to comprehend ambiguous poses. We also subjectively survey the visual quality of the synthesized reenactment. We found that while the visual quality is not at the level of standard video, it is much improved, and is enough to be easily comprehensible.

- We implement a mirror AR reenactment system that allows users to view reenactments on a "magic mirror," allowing them to easily imitate and learn motions. We quantitatively evaluate the performance of our system by comparing how well users learn actions on it as opposed to regular video.

Table 2.1: Comparisons between reenactment systems. Our rough shape-based, voxel carving-based, statistical body-based systems have desirable characteristics.

	Capturing equipment	Shape reconstruction	Clothing handled
Rough shape (Section 3.3)	1 RGB-D camera	Scale pre-built model	Pants, shirts
Voxel carving (Section 3.4)	1 RGB-D camera	Simultaneous	Pants, shirts
Statistical body (Section 4)	2 RGB-D cameras	Simultaneous	Pants, shirts
[18]	7 RGB cameras	Scale pre-built model	Skintight
[99]	8 RGB cameras	No	Any
[24]	8 RGB cameras	Need laser scan	Any
[95]	3 RGB-D cameras	No	Any
[14]	1 RGB-D camera	Simultaneous	Skintight
[93]	68 RGB cameras	Simultaneous, through clothing	Any

We show that users are more accurately and easily able to perform the recorded actions by using our system.

3 Rigid body part-based reenactment with view-dependent texture mapping

3.1 Overview

One way to represent and render the human body is as a collection of rigid body parts. The shape of each body part is determined beforehand, and is assigned a transformation, i.e. rotation and translation, in each frame of the sequence. Finally, in order to render a natural-looking human body, we make use of view-dependent texture mapping [25] to texture each body part according to its transformation. VDTM is a texturing method that uses a large amount of color images to texture a virtual shape, by searching for the closest image to the virtual camera, which would show the viewer the correct angle of the object. It was originally proposed to render geometrically simple shapes that give the illusion of detail, and we use it here in a similar way.

This chapter proposes two methods that use this representation. The first method uses cylinders to approximate the shape of each body part. Cylinders do not follow the shape of the human body exactly, but we made use of VDTM in order to compensate for this.

The second method uses voxel volumes to represent each body part. We perform a voxel carving stage in order to fit each body part volume to the actual body part, relying on the truncated signed depth field [45].

In order to evaluate this method of reenactment, we developed a mobile AR

reenactment system and performed a user study. The system allowed users to view the reenactments on a mobile device equipped with a camera, letting them change the view of the reenactment by physically changing the device's view. The user study evaluated the quality of the reenactments, the power of the system to let learners differentiate ambiguous motions, and the usefulness of the application.

3.2 RGB-D sensor pose estimation

Our first step is estimating the location of the actor within the world. We assume that the videographer is using a single RGB-D camera, capturing a number of frames consisting of one RGB and depth image each. For the n-th RGB image of the captured stream, we first estimate the RGB-D sensor's pose as extrinsic camera parameters $\mathbf{C}_n$ with respect to the world coordinate system using the RGB image with a simultaneous localization and mapping (SLAM) technique.

Figure 3.1 shows the coordinate systems in use. In the system, the world coordinate system is defined as a unique base of the coordinate system for both the capturing and reenactment stage, and is set as the camera pose in the first frame in the capturing stage. The camera pose is treated as a transform from a sensor coordinate system (i.e. RGB-D sensor or viewer's camera) to the world coordinate system.

Here, it should be noted that in practice, 3D points regained from the depth sensor on the RGB-D sensor and those in the SLAM system's map are usually in different coordinate systems. Additionally, the depth sensor is distinct from the RGB camera, and thus there may be some slight translation or rotation between them. In order to correctly render our reenactment with the model, we must calibrate the transformation parameters, i.e. rotation $\mathbf{R}$, translation $\mathbf{t}$ and scale s, among the coordinate systems.

Fortunately, PTAMM [19] tracks a number of map points, which are feature points with estimated 3D coordinates in the world coordinate system. We can project each map point into the depth image to get the corresponding pairs of 3D points, which then gives us the transformation parameters. Given M map points, with $\mathbf{p}_m$ as the position of the m-th map point relative to the RGB camera and $\mathbf{q}_m$ as the corresponding point based on the depth image, we obtain the

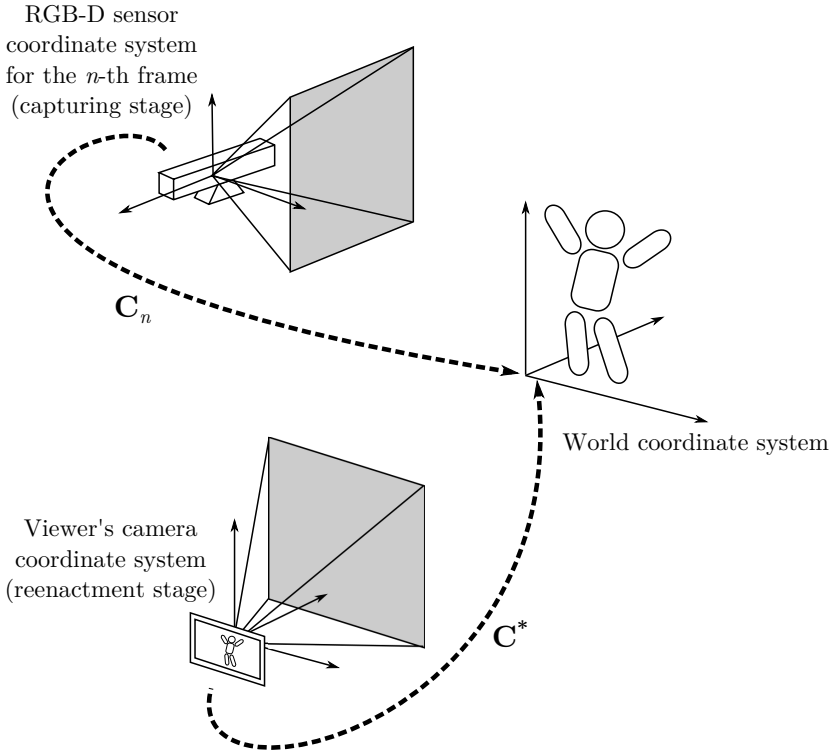


Figure 3.1: Relationship among RGB-D sensor coordinates, viewer's camera coordinates, and world coordinates.

transformation from the skeleton tracker coordinate system to the RGB camera coordinate system as follows:

$$(\bar{\mathbf{R}}, \bar{\mathbf{t}}, \bar{s}) = \underset{(\mathbf{R},\mathbf{t},s)}{\arg\min} \sum_{m=1}^{M} \|\mathbf{p}_m - (s\mathbf{R}\mathbf{q}_m + \mathbf{t})\|^2. \tag{3.1}$$

This least squares problem can be solved by using singular value decomposition. From this point on, all points based on the depth sensor are assumed to have been transformed into the unique world coordinate system defined in the RGB camera coordinate system, i.e. the camera pose estimated from SLAM.

The next sections detail the flow of rough shape-based reenactment, and then voxel carving-based reenactment.

3.3 Rough shape-based reenactment

3.3.1 Capturing stage

For the rough shape-based reenactment, we rely on a skeleton tracker to capture the actor's motion. The skeleton tracker assumes a model of the human body consisting of a set of joints. Each joint has a 3D position which the skeleton tracker estimates in each frame. After estimating the skeleton in a sequence of frames, we then apply shape to the skeleton by defining a set of body parts from the joint positions and assigning a cylinder to each body part.

Skeleton tracking

Figure 3.2(a) shows the N_{J} joints that compose a skeleton, where N_{BP} vectors identified by specific pairs of the joints are referred to as body parts. Each body part can be viewed as a vector formed by the pair of the joints in a specific order. The skeleton of the actor's body in the n-th frame can be extracted and tracked using an existing technique [79]. Assuming a single actor in the scene, we denote the skeleton in the n-th frame by

$$\mathbf{S}_n = \{\mathbf{s}_{n,i} | i = 1, \ldots, N_{\mathrm{J}}\}, \tag{3.2}$$

where $\mathbf{s}_{n,i}$ is the 3D position of the i-th joint of the skeleton in the RGB-D sensor's coordinate system shown in Fig. 3.1.

Using the inverse of $\mathbf{C}_n$, which transforms the 3D coordinates in the world coordinate system to the RGB-D sensor's one, we transform the 3D joint positions in $\mathbf{S}_n$ by $\mathbf{s}'_{n,i} = \mathbf{C}_n^{-1}\mathbf{s}_{n,i}$ for all i in $\mathbf{S}_n$ and define the skeleton in the world coordinate system as $\mathbf{S}'_n = \{\mathbf{s}'_{n,i} | i = 1, \ldots, N_{\mathrm{J}}\}$, so as to store the skeleton in the world coordinate system.

We store the n-th video frame, i.e., skeleton $\mathbf{S}_n$, the RGB image I_n, and depth image D_n in the database.

Rough 3D model preparation

To render the reenactment of the actor, we prepare a 3D model for generating a novel viewpoint image of the actor. We use a cylinder to represent each body part. Since the heights of the cylinders are trivially determined from the length of the body part vector, all we need to determine the cylinders are their radii. For this, we first find the index of a single representative frame $\hat{n}$ from the recorded

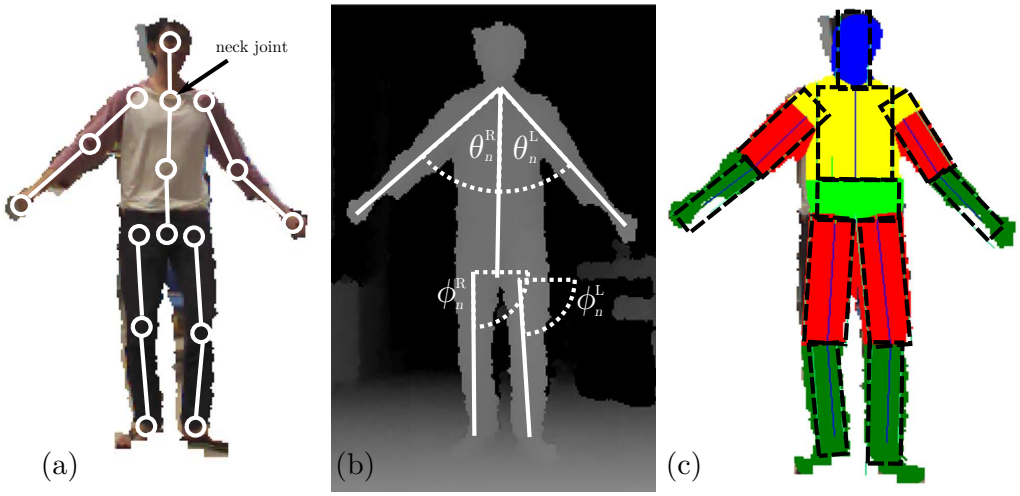


Figure 3.2: (a) The skeleton representation. Circles are joints, and segments are body parts. (b) Corresponding depth image with definitions of some angles. (c) Rectangles fitted to each body part.

video stream and then fit rectangles to the actor's region in the depth image of the representative frame $D_{\hat{n}}$, which can viewed as a projection of the cylinders onto the image plane of the RGB-D sensor.

To obtain radii and heights of the cylinders based on the rectangles that are their projection, the directions of their heights must be perpendicular to the optical axis of the RGB-D sensor. This means that the representative frames should contain the actor's appearance that meet the following requirements: (i) both arms should be away from the body, (ii) the line segments formed by the joints corresponding to both hands should be parallel to the image plane as possible, and (iii) the legs should be uncrossed. These requirements ensure that the representative frame has body parts that are separate from each other as shown in Fig. 3.2(a), making it easier to build an accurate model of the actor's body. Such a pose may be specifically requested of the actor, but it may also be captured during the normal course of recording. We find such a pose by inspecting the angles formed by the body parts.

As shown in Fig. 3.2(b), we denote the angles between the torso and the left and right arms in $\mathbf{S}'_n$ by θ_n^{L} and θ_n^{R}, respectively. We also define term $g(\phi_n^{\mathrm{R}}, \phi_n^{\mathrm{L}})$

that gives a positive value when legs are uncrossed as

$$g(\phi_n^{\mathrm{R}}, \phi_n^{\mathrm{L}}) = \begin{cases} 1 & \text{if } \phi_n^{\mathrm{R}} > \phi_n^{\mathrm{L}} \\ 0 & \text{otherwise} \end{cases}, \tag{3.3}$$

where ϕ_n^{L} and ϕ_n^{R} are the angles between $[1\ 0\ 0]^T$ and the vectors of the left leg and right leg, respectively. This representative frame selection is done in the RGB-D sensor's coordinate system, assuming that the user who capture the video stream does not rotate it very much. The above requirements can be empirically encoded in the criterion

$$E(n) = \theta_n^{\mathrm{L}} a_n^{\mathrm{L}} + \theta_n^{\mathrm{R}} a_n^{\mathrm{R}} + \lambda g(\phi_n^{\mathrm{R}}, \phi_n^{\mathrm{L}}), \tag{3.4}$$

where a_n^{L} and a_n^{R} are the x-components of the left and right arm vectors, whose lengths are normalized to 1 and λ is an empirically-defined constant. The first and second terms ensure that the arms are lifted away from the torso and that they are parallel to the x-axis of the RGB-D sensor's coordinate system. We obtain the index of the most appropriate frame in sense of the above criterion by maximizing E, i.e.,

$$\hat{n} = \arg\max_n E(n). \tag{3.5}$$

We then find the rectangle that fit to each body part in $D_{\hat{n}}$, as in Fig. 3.2(c). The radius r of the cylinder for the body part is then given as the length of the line segments perpendicular to the body part segment. For compensating the slight differences in the body part segment length from frame to frame, we store in the database the radius rate given by r/l for each body part, where l is the length of the body part segment.

3.3.2 Reenactment stage

Since our 3D model of the actor is rough and no color is assigned to it as in Fig. 3.3 (a), we apply textures to our 3D model so as to improve its visual quality. For a static scene, view-dependent texture mapping proposed by Debevec et al. [25] works well for this purpose by assigning as textures those images which

were captured from the viewpoint close to that of the novel image to be synthesized. However, we cannot adopt it naively because the proposed system captures a moving actor and uses only a single RGB-D sensor and thus there are no video frames that capture the same scene at the same time from different viewpoints. Our idea for solving this problem is based on our observation that there still are several video frames that capture a similar actor's pose, which means that we can select a frame such that the joint positions in the selected frame are close to those in the novel image to be synthesized.

Applying camera pose

When reenacting the actor's appearance from the skeleton currently being rendered, $\mathbf{S}'$, we first transform joint $\mathbf{s}'_i$ in the world coordinate system into the viewer camera's coordinate system using $\mathbf{C}_{\mathrm{S}}$, giving us $\mathbf{S}^*$. We also transform $\mathbf{S}'_n$ for all n into its original RGB-D sensor's coordinate system using $\mathbf{C}_n$, giving us $\mathbf{S}_n$.

Appropriate texture search

Since the position of the actor in the world coordinate system varies frame by frame, to make the selection translation invariant, the position of a specific joint is subtracted from the all joint's position so that the specific joint coincide the origin. In this work, we choose the neck joint shown in Fig. 3.2(a) as the origin. We select the appropriate video frame, of which associated skeleton $\mathbf{S}_n$ in the original RGB-D sensor's coordinate system is closest to the $\mathbf{S}^*$ in the viewer camera's coordinate system. To summarize, we find the appropriate frame index $\bar{n}$ by

$$\bar{n} = \arg\min_n \sum_{i=1}^{N_{\mathrm{J}}} \|(\mathbf{s}^*_i - \mathbf{s}^*_{\mathrm{neck}}) - (\mathbf{s}_{n,i} - \mathbf{s}_{n,\mathrm{neck}})\|, \tag{3.6}$$

where $\mathbf{s}^*_{\mathrm{neck}}$ and $\mathbf{s}_{n,\mathrm{neck}}$ are the neck joint positions of $\mathbf{S}^*$ and $\mathbf{S}_n$, respectively.

The limitation of this texture selection is its inability to preserve the facial expression of the actor. However, we consider that it is sufficient to make the actor's motion comprehensible.

Applying textures

Although we selected the appropriate frame for coloring the cylinder, since the poses represented by $\mathbf{S}^*$ and $\mathbf{S}_{\bar{n}}$ are not exactly the same, naively projecting the cylinder to the selected RGB frame can lead to inconsistency between the

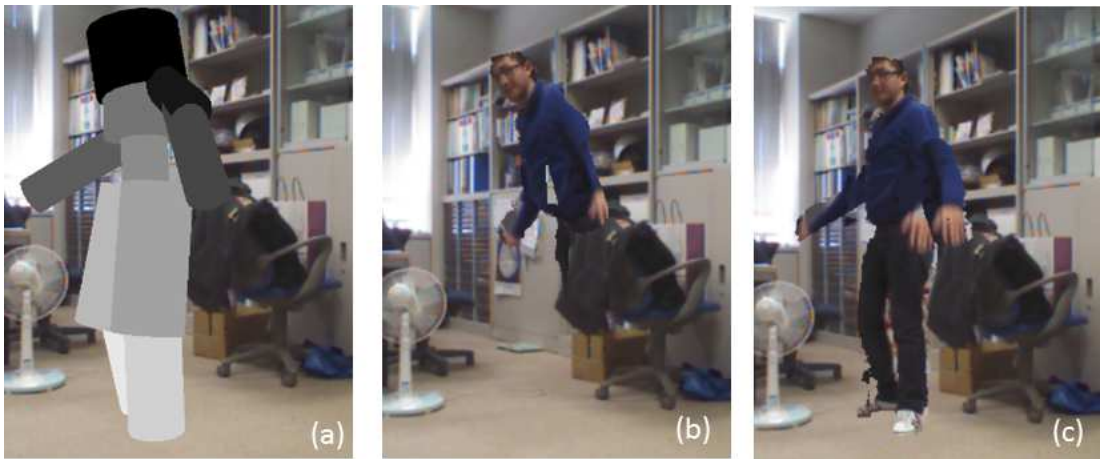


Figure 3.3: (a) The cylinder model. Cylinders are colored for visibility. (b) The colored cylinders, without an individual mapping for each cylinder. (c) The colored cylinders corrected to have an individual mapping for each cylinder.

cylinders and the frame as shown in Fig. 3.3(b). We thus find a projection individually for each cylinder that compensates the actor's poses in $\mathbf{S}^*$ and $\mathbf{S}_{\bar{n}}$, and use the projection to determine the color on each 3D point on that cylinder (Fig. 3.3(c)). Finally, we superimpose the reenactment on the real-time RGB video frame from the viewer's camera.

3.4 Voxel carving-based reenactment

3.4.1 Capturing stage

In this section, we detail the processes which estimate the camera's and actor's pose in each frame and reconstruct the actor's body model. As the input for the processes, a user captures a sequence of RGB-D frames of an actor performing a motion sequence, consisting of RGB images $\{I_n|n = 1, ..., N\}$ and depth images $\{D_n|n = 1, ..., N\}$.

RGB image segmentation

We segment the actor from the background of the RGB images in order to achieve correct body part registration and correct texturing in the reenactment stage.

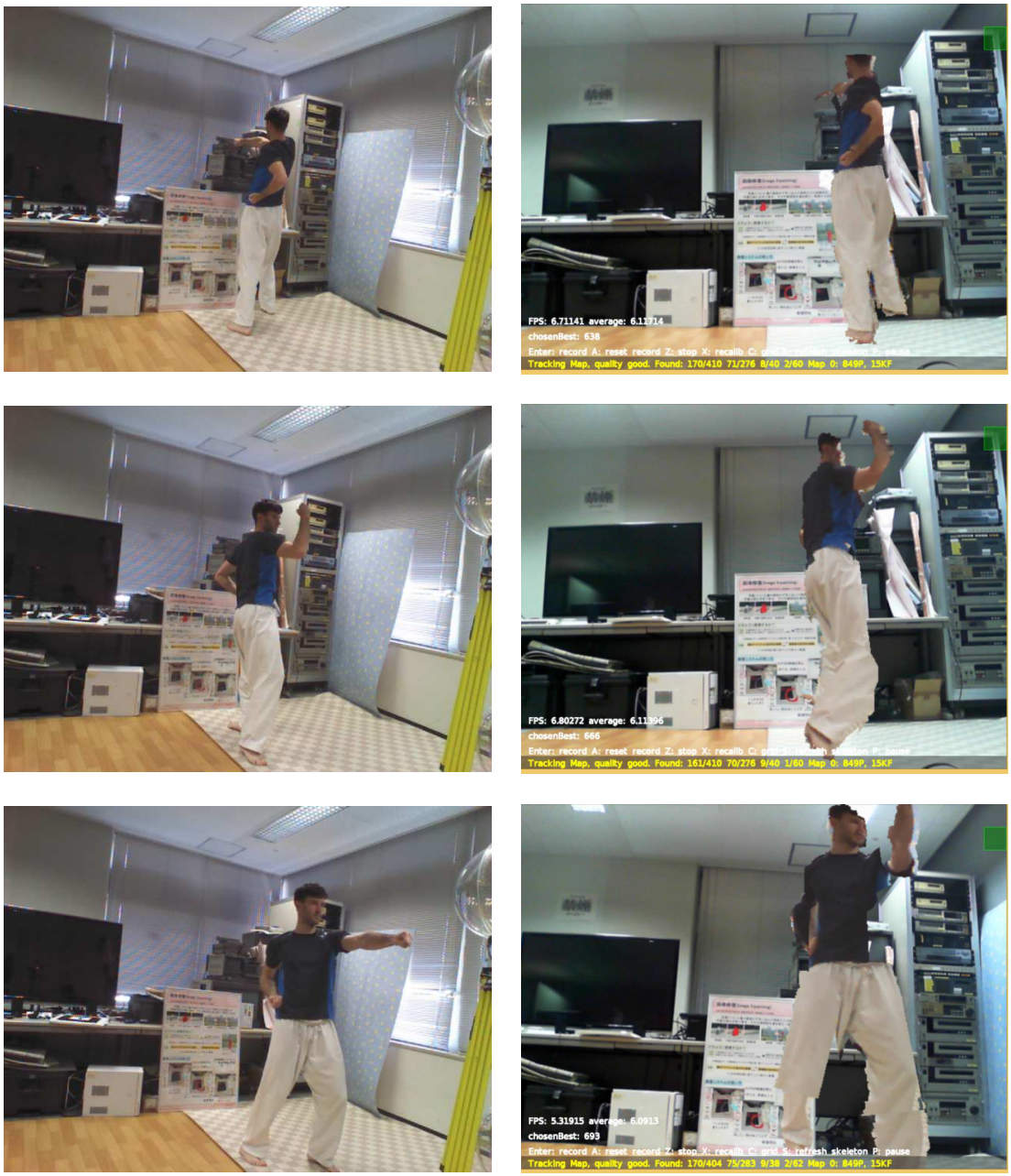


Figure 3.4: Left: frames from the videos. Right: the same frames from a different angle, reenacted by our system.

Here, we employ the "BodyIndexFrame" functionality from the Kinect SDK, which tells us which depth pixels belong to our actor and which belong to the

background.

RGB-D sensor pose tracking and mapping

To obtain camera pose $\mathbf{C}_n$ for the n-th frame, we use PTAMM [19]. $\mathbf{C}_n$ can also be interpreted as the transformation from the current camera coordinate system to the world coordinate system. PTAMM also provides a map of the environment that is constructed of 3D points gained during visual SLAM, as well as their descriptors. This map is in the world coordinate system, which in our system is equivalent to the camera pose in the first frame, i.e. $\mathbf{C}_1$ is the 4×4 identity matrix.

Body part registration

In order to build an accurate model of the actor's body, we use Malleson et al.'s method, described in [57]. They define a model of the actor's body that consists of body parts. In each frame n, each body part b has transform $\mathbf{T}_{b,n}$, which defines its pose, i.e., its rotation and translation, for that frame. Each body part also has a voxel volume V_b, which defines its shape. Pose and shape are closely related, because the accuracy of the reconstructed shape depends on the accuracy of the estimated transforms: in order to correctly shape each body part, each volume must be correctly aligned in each depth image, and this process is called body part registration.

For body part registration, Malleson et al. use a combination of point-to-point and point-to-plane ICP, with an additional constraint given by Kinect skeletal pose estimation, in order to register each body part in each frame. ICP works better with incremental transforms, and so given the previous frame's transform $\mathbf{T}_{b,n-1}$, the current frame's transform $\mathbf{T}_{b,n}$ is defined using a transform delta $\Delta\mathbf{T}$:

$$\mathbf{T}_{b,n} = \Delta\mathbf{T}\mathbf{T}_{b,n-1}. \tag{3.7}$$

$\Delta\mathbf{T}$ is calculated over a number of iterations, until convergence, by minimizing the cost function:

$$E_{b,n}(\Delta\mathbf{T}) = E^p_{b,n}(\Delta\mathbf{T}) + w_o E^o_{b,n}(\Delta\mathbf{T}) + w_s E^s_{b,n}(\Delta\mathbf{T}), \tag{3.8}$$

where $E^p_{b,n}(\Delta\mathbf{T})$ is the point-to-plane term, $E^o_{b,n}(\Delta\mathbf{T})$ is the point-to-point term, and $E^s_{b,n}(\Delta\mathbf{T})$ is the skeletal pose constraint term. Relative weighting coefficients w_o and w_s are applied to the terms. For our system, w_o is set to 1 and w_s is set to

half of the number of voxels in V_b. For the point-to-plane term and point-to-point terms, we register the body part by attempting to fit the 3D points belonging to the body part on frame $n-1$ to the 3D points on frame n, taking into account the difference in camera pose. The 3D points that belong to the body part are obtained by calculating the 3D coordinate of each depth pixel in depth image D_{n-1} and taking those 3D points that are within the volume corresponding to the body part. Each volume has predefined dimensions according to the body part and takes the body part transform $\mathbf{T}_{b,n-1}$. For the first frame, we set each body part transform to the one estimated by the Kinect skeleton tracker.

Point-to-plane ICP term: The point-to-plane ICP term $E^p_{b,n}(\Delta\mathbf{T})$ returns the sum of squared distances between each 3D point belonging to body part b on frame $n-1$, which is regained from depth image D_{n-1}, and the tangent plane on the corresponding point on the surface of frame n, which is a set of 3D points regained from depth image D_n. The point correspondences for point-to-plane ICP are defined as the point pairs having the same depth pixel coordinates across D_n and D_{n-1}, taking into account the difference in camera pose between $\mathbf{C}_{n-1}$ and $\mathbf{C}_n$ and the body part transform delta $\Delta\mathbf{T}$.

Point-to-point ICP term: The point-to-point ICP term $E^o_{b,n}(\Delta\mathbf{T})$ similarly returns the sum of squared distances between each 3D point belonging to body part b on frame $n-1$, which is regained from depth image D_{n-1}, and the corresponding point on the surface of frame n, which is a set of 3D points regained from depth image D_n, with the difference being that the point correspondences are calculated using optical flow between color images I_{n-1} and I_n.

Skeleton constraint term: The skeleton constraint term $E^s_{b,n}(\Delta\mathbf{T})$ returns a measure of distance between the calculated body part transform $\mathbf{T}_{b,n}$ and the estimated body part transform $\mathbf{T}^*_{b,n}$ acquired from the Kinect skeleton tracker.

In order to be able to solve the cost function linearly the small rotation angle

assumption is used to define the transform $\Delta\mathbf{T}$ as:

$$\Delta\mathbf{T} = [\Delta\mathbf{R}|\Delta\mathbf{t}] = \begin{bmatrix} 1 & \alpha & -\gamma & t_x \\ -\alpha & 1 & \beta & t_y \\ \gamma & -\beta & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.9}$$

The derivative of $E_{b,n}(\Delta\mathbf{T})$ can then be computed for each component $(\alpha, \beta, \gamma, t_x, t_y, t_z)$, obtaining a 6×6 symmetric linear system, which is solved as in [45]. $\Delta\mathbf{T}$ is composed onto $\mathbf{T}_{b,n}$ after each iteration.

Body part mesh reconstruction

After estimating transform $\mathbf{T}_{b,n}$ for body part b in frame n, the corresponding depth image D_n is then used to reconstruct its 3D shape as a mesh model. For this process, we basically follow the method [57], with a slight modification. Here, the 3D shape of each body part is reconstructed as a surface model using the voxel-space signed distance function (SDF) [45] and the marching cubes algorithm [55]. Voxel volume V_b has predefined width W_b, height H_b, and depth D_b and contains $W_b \times H_b \times D_b$ voxels. For each voxel, scores can be calculated indicating the average observed signed distance from the surface. Due to such uncertainties as fluctuating depth measurements, each depth image's contribution should be limited. Thus, the SDF is truncated to the range $[-\mu, \mu]$. In addition to this, signed distances beneath the opposite side of the surface will usually be incorrect, as the opposite side is unobserved; therefore, to make the truncated SDF calculation more robust, each frame's contribution that are less than $-\mu$ is ignored in order to avoid interfering with any possible surfaces on the other side. More concretely, the score is defined as follows:

$$F(\mathbf{v}) = \sum_{n=1}^{N} \frac{F_{D_n}(\mathbf{v})}{N^*(\mathbf{v})}, \tag{3.10}$$

$$F_{D_n}(\mathbf{v}) = \begin{cases} \mu & : \quad \mu \leq \eta(\mathbf{v}) \\ \eta(\mathbf{v}) & : \quad -\mu \leq \eta(\mathbf{v}) < \mu \\ 0 & : \quad \eta(\mathbf{v}) < -\mu \end{cases}, \tag{3.11}$$

where $\eta(\mathbf{v})$ is the signed distance from the surface to voxel $\mathbf{v}$ taking into account the transform $\mathbf{T}_{b,n}$, μ is a predefined constant to truncate the SDF, and $N^*(\mathbf{v})$ is the number of frames excluding those with $\eta(\mathbf{v}) < -\mu$.

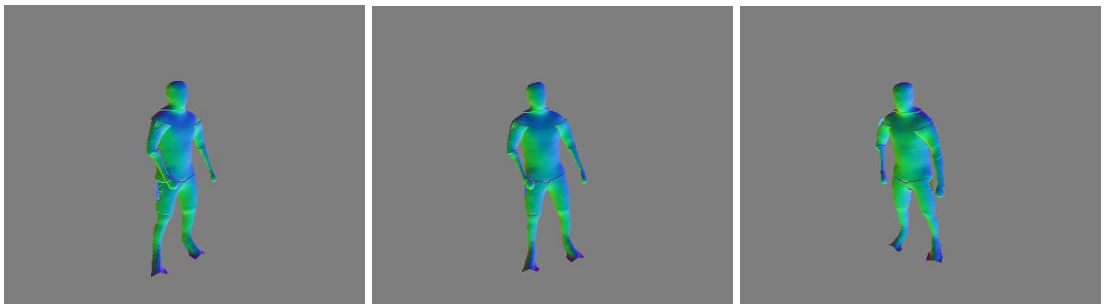


Figure 3.5: The generated body part meshes from different angles.

Finding the zero-crossings will thus give an estimate of surface locations. We apply the marching cubes algorithm [55] in order to convert these voxels into a mesh for each body part, as in Fig. 3.5.

3.4.2 Reenactment stage

This section details the process for reenacting the pose (i.e., the pose of each body part b, $\mathbf{T}_{b,n}$) in the n-th frame overlaid on the real-time image I^*.

Applying camera pose

We use the tracked camera pose $\mathbf{C}^*$ in order to transform each body part to the viewer camera coordinates:

$$\mathbf{T}_b^* = \mathbf{C}^* \mathbf{C}_n^{-1} \mathbf{T}_{b,n}, \tag{3.12}$$

where b is the body part id and n is the frame.

Appropriate texture search

We then apply the appearance of the actor to the transformed body parts by using view-dependent texture mapping. Most existing techniques for NVS use multiple RGB/RGB-D cameras and sensors in order to reduce invisible regions due to occlusion [99] [18] [24] [89]. Since our system captures from a single RGB-D sensor, it instead uses appropriate RGB images over the course of the entire recording. We find appropriate textures for each body part using the similarity of the rotation components of their transforms as a metric. As in Fig. 3.6, we want to find frame n with the rotation that is closest to the rotation computed

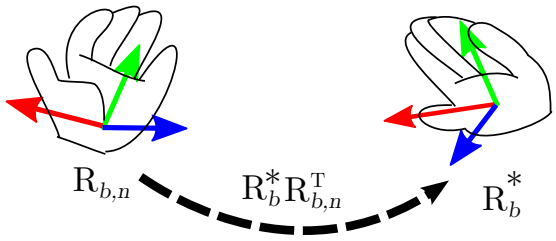


Figure 3.6: The difference between captured body part rotation $\mathbf{R}_{b,n}$ and virtual rotation $\mathbf{R}_b^*$ is expressed as another rotation $\mathbf{R}_b^* \mathbf{R}_{b,n}^{\mathrm{T}}$.

by equation (3.12):

$$\bar{n}_b = \arg\min_n \Phi(\mathbf{R}_b^* \mathbf{R}_{b,n}^{\mathrm{T}}), \tag{3.13}$$

where $\Phi(\mathbf{R})$ converts rotation matrix $\mathbf{R}$ into its axis-angle form and returns the angle, i.e., the magnitude of the rotation.

Applying textures

We map all $\mathbf{x}^*$, the 3D positions of all visible pixels on the surface of the body parts, onto the corresponding transformed mesh as $\mathbf{x}_{\bar{n}_b}$, which are then projected onto the 2D image in order to get the color at the corresponding pixel of RGB image $I_{\bar{n}_b}$.

$$\mathbf{x}_{\bar{n}_b} = \mathbf{T}_{b,\bar{n}_b} \mathbf{T}_b^{*-1} \mathbf{x}^*, \tag{3.14}$$

$$\mathbf{x}_{2\mathrm{D}} = \rho(\mathbf{x}_{\bar{n}_b}), \tag{3.15}$$

where $\rho(\mathbf{x})$ transforms a point into pixel coordinates by multiplying by the camera matrix and dividing by the z-coordinate.

Since the actor is reenacted from a viewpoint different from those at which the textures were originally captured, it should be noted that $\mathbf{x}_{\bar{n}_b}$ can be occluded by other body parts as shown in Fig. 3.7. Background pixels can be detected by referring to the results of actor/background segmentation. In this case, we consider it to be an extraneous part caused by the simplified geometry model, and we show instead the corresponding pixel on the real-time image. To handle occlusion, we take the following strategy. First, the system detects the occlusion in $I_{\bar{n}_b}$ for body part b by projecting each body part in the appropriate pose for

Figure 3.7: Left: checking for occlusions by projecting different body part volumes onto a texture. Right: unoccluded regions for the chest body part.

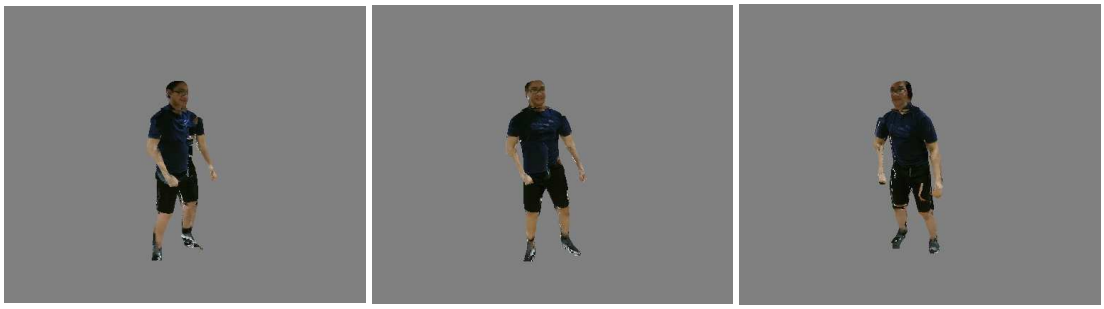


Figure 3.8: Textured meshes for the surface model shown in Fig 3.5.

the $\bar{n}_b$-th frame, i.e. $\mathbf{T}_{b,\bar{n}_b}$ onto the $I_{\bar{n}_b}$, testing for depth map rendered for all body parts(see Fig. 3.7). If the body part is not occluded, the projected body part and the depth map coincide. Otherwise the body part lies farther than the depth map and the system finds the next-best frame instead of $I_{\bar{n}_b}$ and repeats the process until it finds one in which the corresponding pixel is not occluded. The output is shown in Fig. 3.8.

Finally, we overlay the environment image with the synthesized reenactment, as shown in Fig. 3.9.

3.5 User study

We implemented the proposed reenactment viewing system on a mobile device and evaluated its effect on users' comprehension of actor's poses. In this experiment, the effectiveness of the system is evaluated by checking the pose errors

Figure 3.9: Environment image overlayed with reenactment.

defined between the true pose and the pose recognized by subjects from the system's output. We then confirm the quality and applicability of the proposed reenactment systems.

3.5.1 Mobile reenactment viewer

We captured motion sequences of performances using a Microsoft Kinect 2. We implemented our AR reenactment system on a Microsoft Surface Pro 2 with 4GB RAM and 1.60GHz processor. For skeleton tracking as well as actor-background segmentation, we relied on the implementation in the Kinect SDK [79]. Our body model contains 15 body parts, seen in Fig. 3.10. With this configuration, we achieved an interactive FPS ranging from 8 to 12 frames per second during reenactment.

3.5.2 Evaluation

In order to evaluate the system, we experimentally tested users' comprehension of actor's poses with the reenactment compared with their comprehension with conventional 2D images and video using 21 subjects. The experiment consists of two parts.

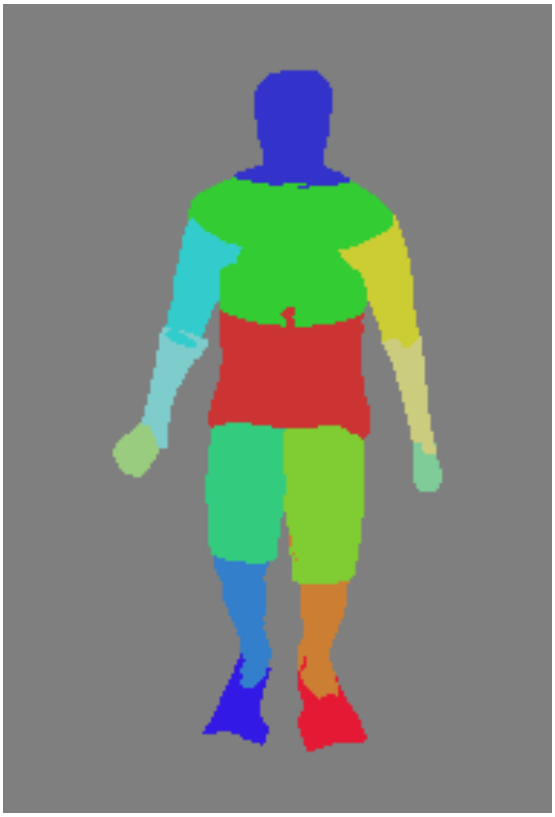


Figure 3.10: 15 body parts used for body modeling.

3.5.3 Experimental setup

Pose angle estimation

In the first part of experiments, users were tasked with estimating the angle of the actor's arm. The actor was asked to form four different poses with specific angles between his arm and torso, and we captured these poses with both our proposed system and a conventional camera, as shown in Fig. 3.11. Each pose was captured from a different viewing angle, as illustrated in Fig. 3.12 and detailed in Table 3.1, in order to test the effect of viewing direction on angle comprehension. In order to aid our system in collecting textures, we also captured the actor from different points of view, asking him to hold the pose as still as he could. For each pose, we showed half of our users the conventional image, and the other half were made to view the pose as an AR reenactment using our proposed system.

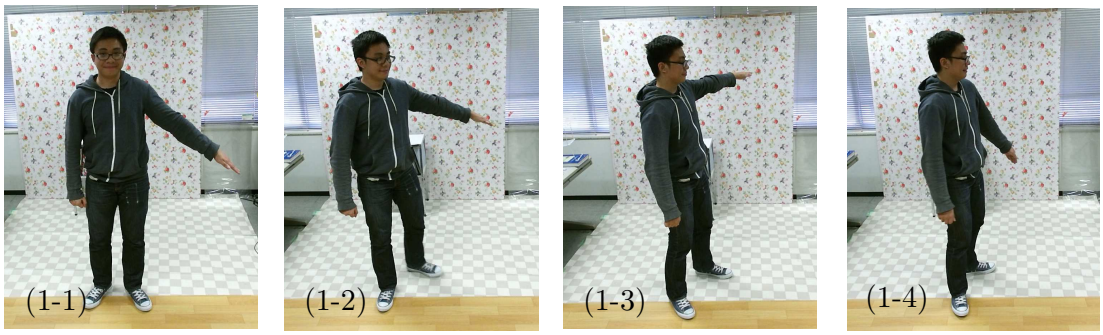


Figure 3.11: Conventional images depicting the poses which were shown to the users for pose angle estimation. In each pose, the actor forms a different angle with his arm. Each image is also taken from a different viewing angle.

Table 3.1: Pose angle estimation results. For the users' answers, the mean absolute errors (MAE) for both the conventional images (conv.) and the proposed system (prop.) were calculated.

	Arm angle	Viewing angle	Conv. MAE	Prop. MAE
Pose (1-1)	47°	0°	7.25°	7.62°
Pose (1-2)	68°	26°	6.70°	9.01°
Pose (1-3)	95°	46°	10.48°	3.11°
Pose (1-4)	32°	57°	10.59°	4.90°

Users alternately viewed either the conventional image or the AR reenactment per pose. Specifically, users were divided into Group A and Group B. Users in Group A were shown Pose (1-1) and (1-3) in conventional images and Pose (1-2) and (1-4) using the proposed system, while those in Group B were shown the opposite.

Users were asked to form the angle using a compass while viewing the pose. We then calculated the mean absolute error (MAE) for all users for the viewers of the conventional image and of the proposed system.

Table 3.1 also shows the results of the experiment. The proposed system's errors were generally lower than the conventional result. We can see that as the

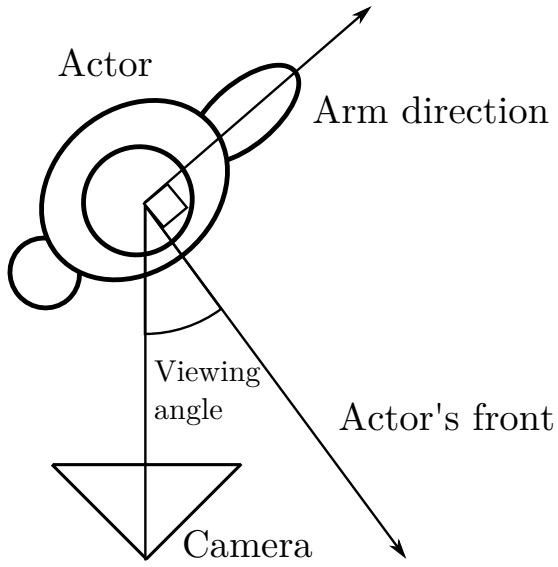


Figure 3.12: Viewing angle shown from the top. A value of 0° means that the actor is facing the camera. Arm direction is always perpendicular to the actor's front.

viewing angle of the conventional image increases, the arm angle estimation error also tends to increase. Because the proposed system allows users to view the actor's pose from wherever they want (see Fig. 3.13), they could choose to view it from the viewpoint that allows for the easiest estimation, i.e. from directly in front of the actor.

We also note that for the proposed system, the MAE is higher for Poses (1-1) and (1-2) than for (1-3) and (1-4). We consider that this may be caused by the order of poses which are shown to users: group A users are shown Pose (1-1) first, then Pose (1-3), while group B users are shown Pose (1-2) first, then (1-4). This means that it takes some time to get used to our system.

Pose matching

In the second part of experiments, users were tasked with discerning the actor's pose. We formed four poses with a small mannequin and had the actor perform these poses, which we captured both with a conventional camera and our proposed system (Figs. 3.14 and 3.15). Similarly to the angle estimation, we captured the actor from different points of view in order to aid our texture selection,

Figure 3.13: Poses (1-1)–(1-4) for pose angle estimation from the front (top row) and side (bottom row), viewed using the AR reenactment system.

Table 3.2: Pose matching results. Conventional and proposed system results refer to the rate of correct answers.

	Correct answer	Conventional result	Proposed system result
Pose (2-1)	C	80%	82%
Pose (2-2)	B	40%	73%
Pose (2-3)	C	55%	100%
Pose (2-4)	A	36%	80%

asking him to hold the pose as still as he could. For each of the four initial poses, we also formed two similar poses with the mannequin, shown in Fig. 3.16, making 12 poses in all. We alternately showed users the conventional image, and the AR reenactment. Users in Group A viewed Pose (2-1) and (2-2) using the conventional images and (2-3) and (2-4) using the proposed system, and users in Group B viewed the opposite.

Users chose the closest pose from three mannequins' poses (Fig. 3.16). We decided to let the users choose between mannequin poses because these would not contain cues, e.g. clothing folds, shadows, etc., that would relate them to the conventional image. Users were encouraged to view the AR reenactment from different viewpoints.

Table 3.2 shows the results for this experiment. Users scored higher with our system than with conventional images for all poses. We consider that this is because the poses are not very discriminative from the frontal views that were shown to the users, while our system can provide side views.

3.5.4 Survey

We gave users a survey on the quality and applicability of the system, comparing it to the quality and applicability of the rough shape model-based system. First, users were shown the "Exercise" motion sequence rendered using rough shape-based reenactment, as in Fig. 3.17 (left). Users were then asked to answer the survey in Table 3.3. Next, users were shown the same sequence rendered using

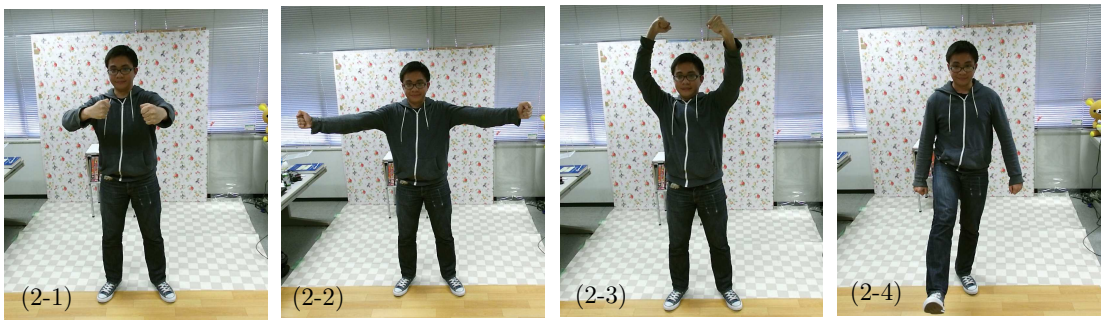


Figure 3.14: Conventional images depicting the four poses that were shown to the users for pose matching.

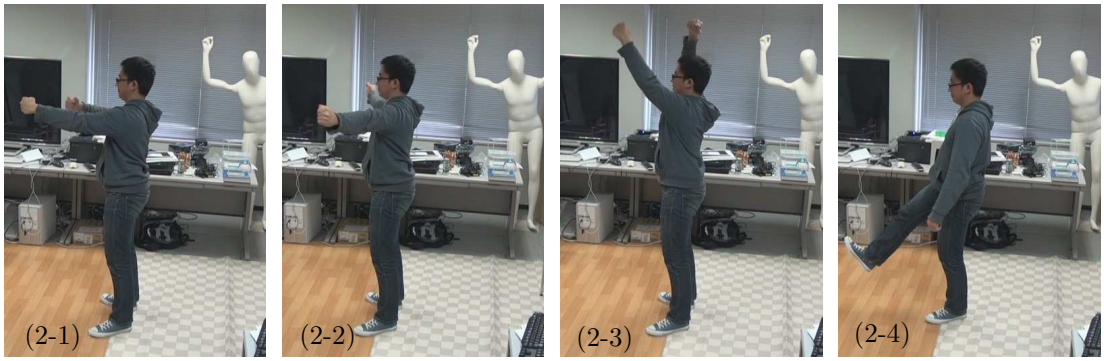


Figure 3.15: Poses in Fig. 3.14 viewed from the side.

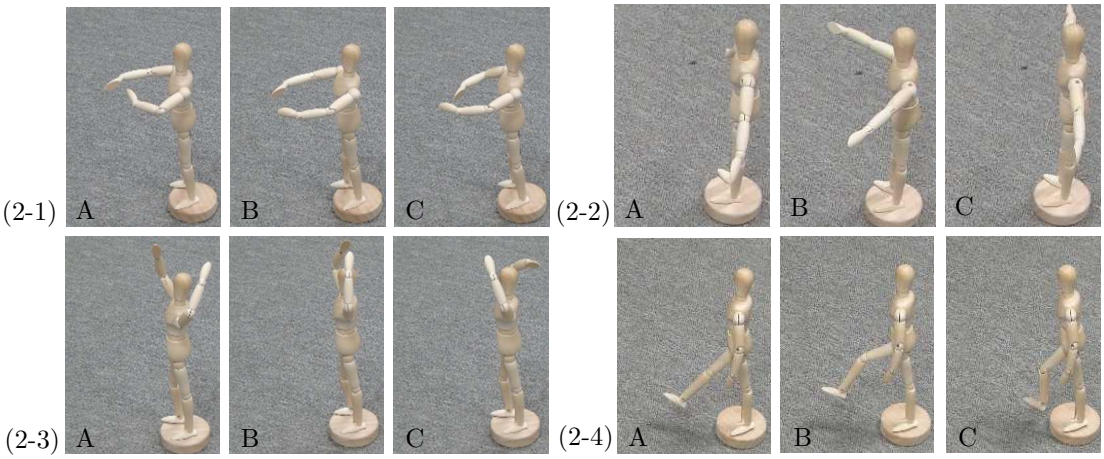


Figure 3.16: The mannequins to match the poses to. Correct answers are C for 1, B for 2, C for 3, and A for 4.

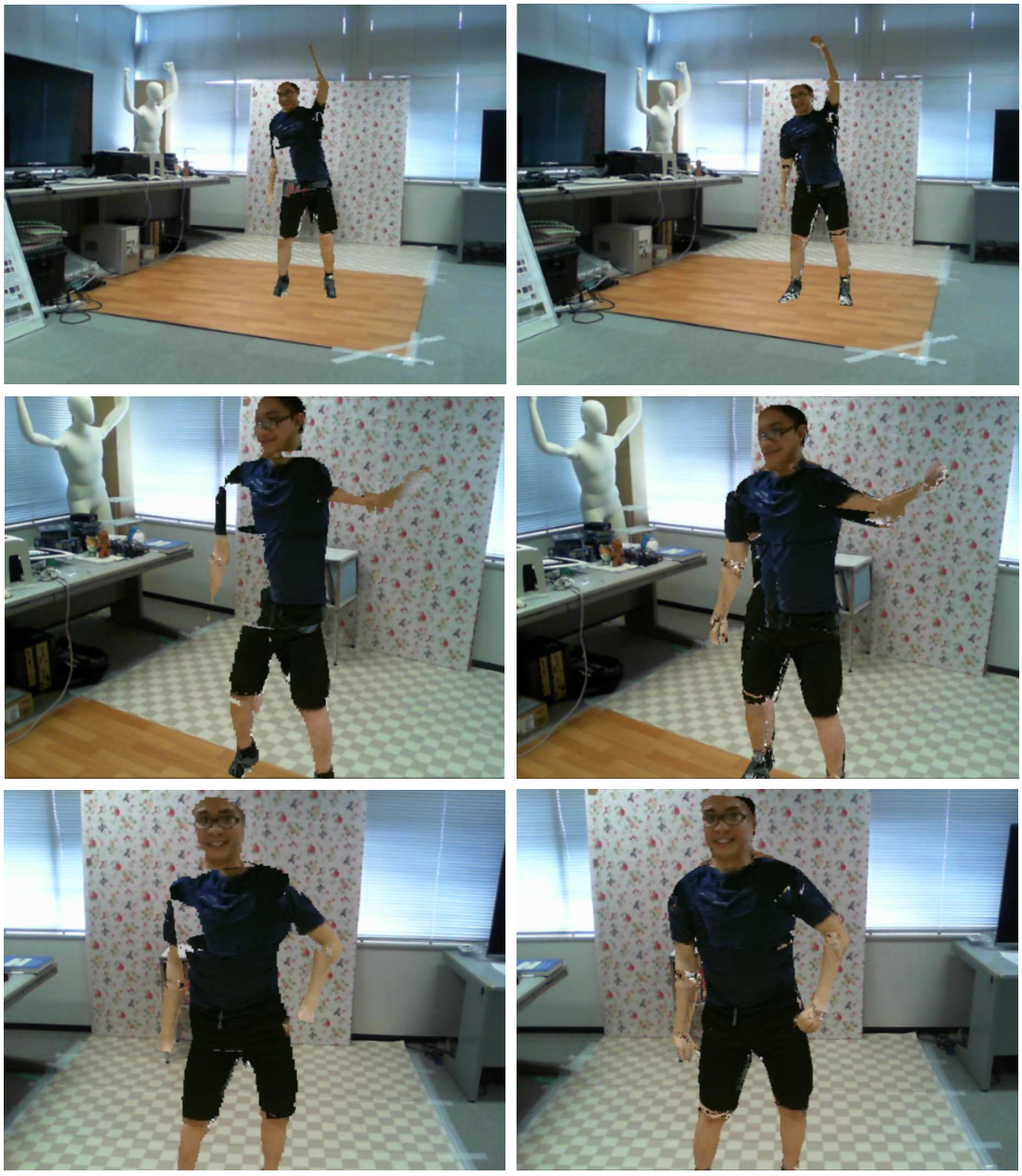


Figure 3.17: Left: "Exercise" sequence viewed with the previous, cylinder-based system. Right: The same frames viewed with the proposed system.

Part 1. For each reenactment (rough shape-based, voxel carving-based):

Q1 I am not bothered by holes and artifacts in the reenactment.

Q2 The reenactment's motion is smooth.

Q3 The reenactment resembles the conventional video.

Part 2. For each reenactment (rough shape-based, voxel carving-based): I would prefer to watch the reenactment over the conventional video for...

Q4 ...performances.

Q5 ...training videos.

Q6 ...sports recordings.

Q7 ...videos of daily life.

Figure 3.18: Questions asked in our user study. Users answered from 1 (strongly disagree) to 5 (strongly agree).

voxel carving-based reenactment, as in Fig. 3.17 (right) Users were then asked to answer the same questions a second time.

The survey shows that while users were not entirely satisfied with the quality, they were positive toward the reenactment. The answers to Q1 shows that enough holes and artifacts exist in the rendering that they disturb the users' experience of the previous system. These holes are the result of the rough 3D modeling of the target. The output quality has been improved for the proposed system by employing the state of the art body modeling method [57]. Q2 shows that most of the users thought that the motion was smooth enough, with the proposed system scoring higher. Q3 asks whether the synthesized reenactment looks like the original video. If viewed from the original capture point, it should strongly resemble the video since it is using the same video frames as textures. If viewed

Table 3.3: Users' averaged answers for the survey in Fig. 3.18 for rough shape model-based reenactment (A) and voxel carving-based reenactment (B). Users answered from 1 (strongly disagree) to 5 (strongly agree).

	A	B
Q1	2.62 ± 0.86	$\mathbf{3.86} \pm 1.03$
Q2	3.71 ± 1.00	$\mathbf{4.29} \pm 0.76$
Q3	3.81 ± 1.11	$\mathbf{4.24} \pm 0.76$
Q4	3.10 ± 1.40	$\mathbf{3.62} \pm 1.26$
Q5	4.05 ± 1.00	$\mathbf{4.57} \pm 0.73$
Q6	3.52 ± 1.30	$\mathbf{4.05} \pm 1.08$
Q7	2.43 ± 1.52	$\mathbf{3.29} \pm 1.22$

from elsewhere, however, it must be believable enough to look like it was captured from that viewpoint, and as the answers to Q3 show, most users felt that it accomplished this task, with the proposed system's output being closer to the conventional video due to having a more accurate body model. Reactions to the listed applications were also positive. The highest-scoring application were training videos and sports recordings. Users scored our proposed system higher in all aspects compared to our previous system, which shows a marked improvement in quality.

3.6 Summary

This section shows two methods to capture human motion as a reenactment, as well as an application for viewing the reenactment in the form of a handheld AR reenactment viewer. For both methods, the process of capturing only requires a single RGB-D camera, which makes it easier for non-expert users. The reenactments are rendered by reconstructing the actor's body parts using 3D mesh models and texturing them using the RGB video sequence. The reenactment's virtual view is based on a map of feature points in the environment which we generate using visual SLAM during capturing and reuse in order to render the reenactment relative to its original capturing location. The reenactments are

comprehensible by users and generally resemble the video they were based on. Users of the system are able to more precisely estimate body angles at any viewing angle. For cases involving ambiguous poses, the proposed system benefits the users by allowing them to view the pose from multiple angles.

4 Statistical, nonrigid body model-based reenactment

4.1 Overview

Our previous approach was able to generate views of a human in motion. However, this method had drawbacks. It does not perfectly represent the deformation that a true human body undergoes when it moves. This is somewhat alleviated by the view-dependent textures that we apply to the body parts, but these rely heavily on the RGB images that are captured. If this kind of deformation has not been captured, the result may be lacking.

Statistical body models are a way to handle this kind of deformation based on motion. The human body deforms in many subtle ways that are impossible to manually specify, but a large amount of body scans are analyzed, these deformations may be solved for.

One of the earlier examples of statistical body models is SCAPE [6]. SCAPE, standing for shape completion and animation of people, describes human body shape using two learned statistical models: pose deformation and body shape variation. It is based off of the model that moving human bodies deform in the same way for the same poses but also that each body part will roughly stay the same shape. We based our work off of the tensor-based human body model, or TenBo [20]. In contrast to SCAPE, which learns two separate statistical models, TenBo integrates both models into one formula, giving greater reconstruction accuracy as well as requiring less training data.

Figure 4.1: Our ReMagicMirror system. The learner is mirrored on the left in the screen, and the reenactment of the teacher is shown on the right.

4.2 Capturing stage

In the capturing stage, our system records an action of the teacher using a pair of RGB-D sensors facing each other. The relative pose between these two sensors is calculated, and they are manually synchronized. Since we require the depth and color pixels belonging to the teacher, separate from the background, we extract the teacher's region using such a method as [79]. After extracting the teacher's region, we regain the 3D position of each depth pixel to form a point cloud. We merge the two point clouds from the pair of sensors using the relative pose calculated above.

We denote the f-th frame point cloud with N_f points, by

$$Z_f = \{\mathbf{z}_{fn} | n = 1, \ldots, N_f\}, \tag{4.1}$$

and the RGB images from first and second sensors as I_f^1 and I_f^2, respectively.

4.3 Fitting stage

Figure 4.3 (a, top) shows examples of merged point clouds. Generally, even though we capture the teacher from both his front and back, the point cloud can

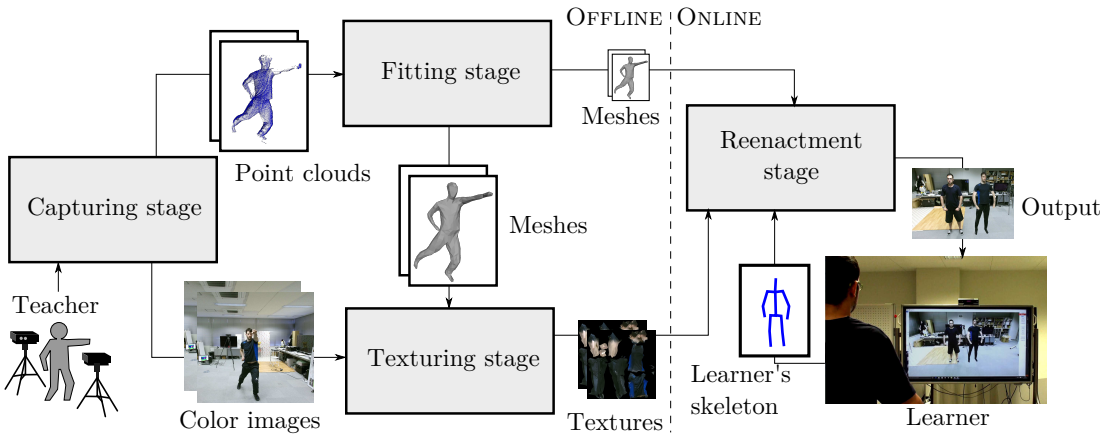


Figure 4.2: System overview.

be incomplete because of occlusion or difficult-to-capture regions such as hair. In addition, some body parts can partially be out of the sensors' field of view. To reconstruct the complete shape of his body, we fit the TenBo model [20], which is a state-of-the-art statistical human shape model, to each point cloud.

4.3.1 Mesh definition

Training a statistical human shape model, usually requires a large amount of registered meshes of multiple subjects in various poses. We used the MPII dataset [37], which contains over 500 registered meshes. For stable fitting, we selected a mesh and reduced the number of vertices in it from 6,449 to 502 using the quadric edge collapse decimation algorithm [21]. From here we treat the decimated mesh as the reference.

This decimation is transferred to all other meshes in the dataset as they are registered, i.e., we keep the same vertices in a mesh as the reference and use the edges in the reference instead of the original ones. We refer to the reference as

$$M_{\mathrm{X}} = \{X, E\}, \tag{4.2}$$

where $X = \{\mathbf{x}_j | j = 1, \ldots, J\}$, $\mathbf{x}_j$ being the j-th vertex, and E contains the pairs of vertex indices that form the edges of the reference. The TenBo model also requires segmenting the mesh into body parts so that each body part is not

(a) (b)

Figure 4.3: (a) Top: Example input point clouds. Middle: Examples of fit meshes. Bottom: Textured meshes. (b) Segmented reference mesh, front and back. Each color in (b) represents one of the 13 body parts: head, shoulders, upper arms, lower arms, torso, abdomen, upper legs, and lower legs.

subjected to excessive deformation. Instead of using an automatic approach, such as [5], we manually segmented the mesh as in Fig. 4.3 (b).

The TenBo model, like other parametric shape models such as [6], regresses a deformation matrix of each triangle in the reference given the body part poses Θ and shape parameter $\mathbf{v}$, where $\Theta = \{\boldsymbol{\theta}_l | l = 1, \ldots, L\}$ is a set of rotation representations for all body parts. The body part rotation matrices $\mathbf{R}(\boldsymbol{\theta}_l)$ are simply each $\boldsymbol{\theta}_l$ in rotation matrix form. We derive the deformation matrices $\mathbf{D}_k(\Theta, \mathbf{v})$ from the TenBo model, with the idea being that deformation for each triangle k is affected uniquely and individually by the shape and pose parameters.

The deformed triangle k's edges, which are called triangle vectors, $\triangle \mathbf{y}_{k1}$ and $\triangle \mathbf{y}_{k2}$ can thus be given by

$$\begin{aligned} \triangle \mathbf{y}_{k1} &= \mathbf{R}(\boldsymbol{\theta}_l)\mathbf{D}_k(\Theta, \mathbf{v})\triangle \mathbf{x}_{k1} \\ \triangle \mathbf{y}_{k2} &= \mathbf{R}(\boldsymbol{\theta}_l)\mathbf{D}_k(\Theta, \mathbf{v})\triangle \mathbf{x}_{k2}, \end{aligned}$$

where $\triangle \mathbf{x}_{km} = \mathbf{x}_{km} - \mathbf{x}_{k0}$ and $\mathbf{x}_{km}$ $(m = 0, 1, 2)$ is in X and forms a triangle of the mesh. In the above equation, l is the body part that triangle k belongs to.

Here is the formulation for the deformation matrices:

$$d_l = \mathcal{G} \times_1 \mathbf{v}^T A^T \times_2 \boldsymbol{\theta}_l \times_3 B. \tag{4.3}$$

$\mathcal{G}$, A, and B are internal TenBo matrices, learned through training on a set of meshes. d_l gives all the values of each $\mathbf{D}_k$, reshaped into a 1-column vector.

4.3.2 Optimization

The fitting algorithm tries to find the body part poses Θ and the shape parameter $\mathbf{v}$. We modify the fitting algorithm in [20] to take advantage of the temporal continuity of meshes in successive frames. More specifically, we apply an additional smoothness term for the pose parameters that penalizes pose differences between adjacent frames, as well as modifying the shape parameter fitting to simultaneously take multiple frames into account. The optimization involves three terms: the model error term $\mathcal{M}$, the point cloud error term $\mathcal{P}$, and the temporal pose smoothness term $\mathcal{R}$.

Cost functions

The model error term penalizes the difference between the TenBo model-based body shape prediction and the deformed mesh Y_f in frame f. $\triangle \mathbf{y}_{fkt}$ is triangle vector $t \in \{1, 2\}$ of triangle k in frame f, the term is given by

$$\mathcal{M}(Y_f, \Theta_f, \mathbf{v}) = \sum_{k=1}^{K} \sum_{t} \|\mathbf{R}(\boldsymbol{\theta}_{fl})\mathbf{D}_k(\Theta_f, \mathbf{v})\triangle \mathbf{x}_{kt} - \triangle \mathbf{y}_{fkt}\|^2. \tag{4.4}$$

The point cloud error term $\mathcal{P}$ for frame f is the difference between the deformed mesh Y_f and the point cloud Z_f. As there are no explicit correspondences between the deformed mesh and the point cloud, we first use the rigid iterative

closest point (ICP) algorithm to bring the mesh into rough alignment, then assign correspondences by nearest neighbor. Using $\tilde{\mathbf{y}}_f(\mathbf{z}_{fn})$ as the nearest vertex in Y_f to point cloud point $\mathbf{z}_{fn}$, the point cloud error term is

$$\mathcal{P}(Y_f) = \sum_n \|\tilde{\mathbf{y}}_f(\mathbf{z}_{fn}) - \mathbf{z}_{fn}\|^2. \tag{4.5}$$

The pose smoothness term $\mathcal{R}$ for frame f penalizes large differences in pose between frames. Due to our assumption of fitting depth image sequences, we do not want subsequent frames to vary wildly. This term increases fitting robustness. The term is defined as the sum of squared Frobenius norms:

$$\mathcal{R}(\Theta_f, \Theta_{f+1}) = \sum_l \|\mathbf{R}(\boldsymbol{\theta}_{fl}) - \mathbf{R}(\boldsymbol{\theta}_{(f+1)l})\|_{\text{fro}}^2. \tag{4.6}$$

The final meshes $M_{\text{Y},f} = \{Y_f, E\}$ can be found by minimizing the following objective with respect to Y_f and Θ_f for $f = 1, \ldots, F$ as well as $\mathbf{v}$:

$$\sum_{f=1}^{F} [\mathcal{M}(Y_f, \Theta_f, \mathbf{v}) + w_z \mathcal{P}(Y_f)] + w_r \sum_{f=1}^{F-1} \mathcal{R}(\Theta_f, \Theta_{(f+1)}). \tag{4.7}$$

We cannot handle all frames at once because of memory requirements. We instead use a sliding window of three frames at a time with the second and third frames' parameters being updated (frames 1 and 2 are independently minimized). Figure 4.3 (a, middle) shows examples of fit meshes.

Implementation

In order to perform the optimization, we used coordinate descent, i.e. optimizing one group of variables at a time and holding the rest constant. In this case, the groups of variables to be optimized are:

1. The meshes, Y_f,
2. The TenBo body pose parameters, Θ_f,
3. And the TenBo shape parameters, $\mathbf{v}$.

To solve for the meshes, each point in each frame's point cloud must have a corresponding mesh vertex in that frame's mesh, as the point cloud term minimizes the sum of these distances. We use a modified nearest neighbor term

that additionally takes into account the similarity of the normals for increased stability:

$$\tilde{\mathbf{y}}_f(\mathbf{z}_{fn}) = \arg\min_{\mathbf{y}_f}(|\mathbf{y}_f - \mathbf{z}_{fn}|)(\epsilon - \hat{\mathbf{y}_f} \bullet \hat{\mathbf{z}_{fn}}), \tag{4.8}$$

where $\hat{\mathbf{y}_f}$ is the normal of mesh point $\mathbf{y}_f$, $\hat{\mathbf{z}_{fn}}$ is the normal of point cloud point $\mathbf{z}_{fn}$, which is calculated by using corresponding adjacent depth pixels in the depth image, and ϵ is some value > 1 that controls the weight of normal similarity.

After determining correspondences, we can solve for the meshes using a least-squares solver. In order to represent our optimization in the form of $Ax = b$, we think of x as all mesh vertices in $Y_{1\ldots F}$, reshaped to 1 column. b will contain elements from the TenBo formulation $\mathbf{R}(\boldsymbol{\theta}_{fl})\mathbf{D}_k(\Theta_f, \mathbf{v})\triangle\mathbf{x}_{kt}$ as well as each point in the point clouds that has a corresponding mesh vertex. A relates x to b, which means that it depends on the configuration of faces, as well as the depth point correspondences.

To solve for the rotations, we handle the rotation deltas using the small angle assumption as in Eq. 3.9. We can then solve for the rotations linearly, using a least-squares solver, similarly to how we solve for the meshes.

Finally, the TenBo shape parameters can be solved for by taking the derivative of the cost function with regards to $\mathbf{v}$ and solving at 0.

We repeat these steps until convergence, updating all $\mathbf{D}_{fk}$ every time after body part rotations or shape parameters get updated.

4.4 Texturing stage

We texture our mesh using values from the RGB images. Since we can now project the mesh into each RGB image, we simply find the correspondences.

Our system extracts textures from RGB images I_f^1 and I_f^2 from the first and second sensors using $M_{\mathrm{Y},f}$ $(f = 1, \ldots, F)$. For each triangle in frame f, we project its vertices $\mathbf{y}_{km}$ to I_f^1 and I_f^2. Since the image region corresponding to a triangle may not necessarily be visible (e.g., an arm may be occluding the body), we must detect and handle such regions.

To do this, we generate a depth map of $M_{\mathrm{Y},f}$ for each sensor that captures I_f^1 and I_f^2, and project a vertex to them. If the depth component of one of the

vertices in a triangle is inconsistent with the corresponding depth value by a threshold T, we deem the triangle not visible. If the triangle is not visible from both sensors, we use the averaged texture calculated over corresponding visible triangles in the entire sequence. Figure 4.3 (a, bottom) shows some examples of textured meshes.

4.5 Reenactment stage

In the reenactment stage, the system reenacts the captured action and presents it to the learner through our interface with the mirror metaphor. This section describes reenactment generation and the interface in detail.

4.5.1 Action learning through magic mirror

In order to help learners perform actions, we bring in the magic mirror metaphor. As before, we are faced with a user interface problem: it is difficult to manipulate the reenactment to find the desired 3D view. The mobile reenactment viewer we previously designed lets learners intuitively find this view, but this required the learners to use both hands, which does not let them perform the action at the same time. According to the experiential learning model [52], people learn best with real experience, which means that letting our learners actively copy the actions themselves would be better than just having them passively watch a motion sequence.

Thus, we implement our AR mirror system. For the display, used a large screen and mounted a Microsoft Kinect v2 on top of it. It displays a mirror image of the learner, like a real mirror, and it overlays the reenactment on top of the image of the learner. To facilitate easy comparisons, we detect which way the learner is facing, and rotate the reenactment in the same way, so that the reenactment and the learner are facing the same way. In this way, learners will be able to directly compare their own motions to the reenactments while they are performing the action.

For this, we use a skeleton tracker (e.g., [79]) to obtain the learner's shoulders' position and compute the learner's direction. After a fixed amount of time, the system fixes the rotation of the teacher's reenactment and starts playing the

action. Once the action plays completely, it resets and the learner can adjust his or her facing again.

Figure 4.1 shows the configuration of our system's learning interface. The interface has one RGB-D sensor to capture the learner and the environment as well as a screen to present the captured live video stream from the sensor and the reenactment of the teacher. The RGB image in the live video stream is flipped before it is presented to the learner so that it appears like a mirror. Note that the image is not a true mirror image as the RGB-D sensor is on top of the screen. We however consider it similar enough to the learner's mental model of a mirror.

4.6 User study

To implement our system, we used two Microsoft Kinect v2s as our RGB-D sensors. We used Kinect v2 SDK for extracting the teacher's region in depth maps and for skeleton tracking. The fitting stage is implemented on a Windows PC with 3.20GHz CPU and 32GB memory. Optimization process (Eq. (4.7)) takes around 5 minutes per frame. We use $w_z = 1$, $w_r = 0.05$, and $T = 10$ cm. For the reenactment stage, the screen is 165 $\times$ 97 cm. The system was implemented on a Windows PC with 3.40GHz CPU and 8GB memory. It runs at 20FPS.

We conducted an objective evaluation to demonstrate how well our system helps users learn actions and a survey to subjectively evaluate our system in terms of ease of use, effectiveness, graphics quality, and appeal.

4.6.1 Experimental setup

We compared the system against the process of learning by imitating a video. We recorded four Taekwondo actions (A, B, C, and D) for this purpose, ranging from 4-12 seconds long. We divided the actions into two groups: Group 1, consisting of actions A and B, where the teacher mainly faced forward, and group 2, consisting of actions C and D, with no restriction. Users learned one action from each group using the system, and the other with the video.

For this evaluation, we recruited 14 users with ages ranging from 20-30, with 3 female and 11 male users. The process of learning an action is as follows: First,

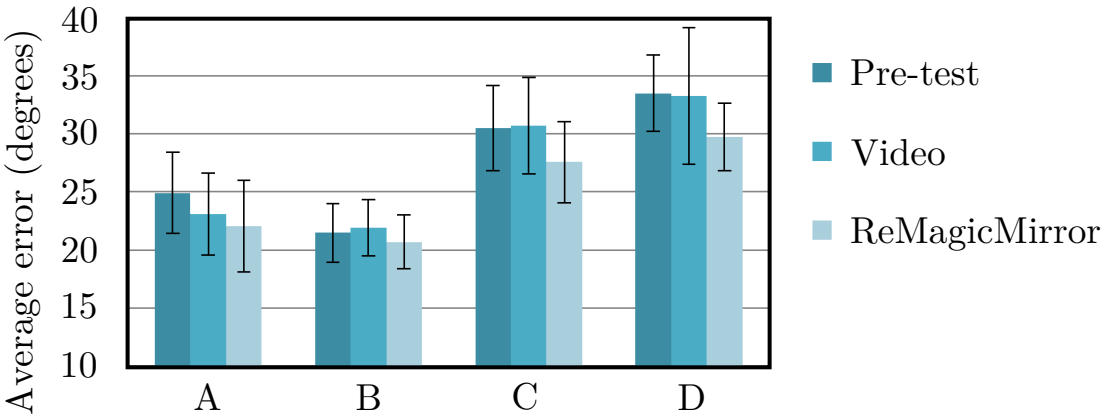


Figure 4.4: Average error in degrees per joint, per frame, between the user and the teacher, for action sequences A, B, C, and D.

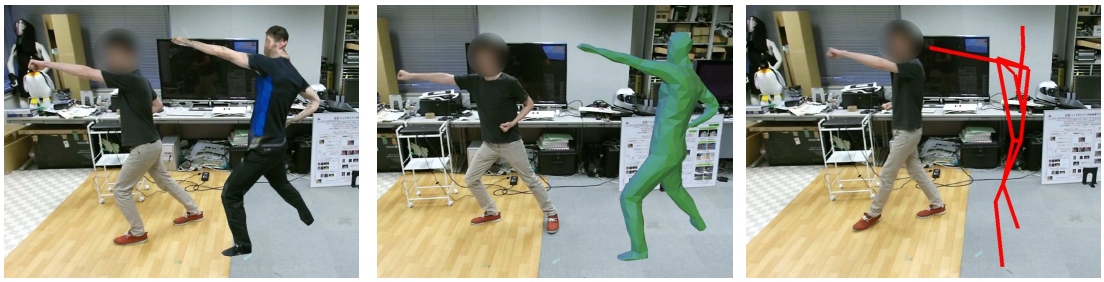
we show a video of the action to the user. Next, we establish a baseline by having the user perform the action and recording it, while the video plays again. After that, the user learns the action by practicing it over and over. The practice is accompanied either with a video of the action looping repeatedly, or with our system looping the reenactment repeatedly. For our system, the user can freely change the viewing direction before every repetition. Finally, we test the user's learning by playing the video or the reenactment one last time and recording, comparing it to the baseline.

We measured the error by recording the users' motion using a Kinect v2. Since we play the video or the reenactment at the same time that the users perform the action, we are able to match body pose frames up one to one and compare each frame directly. We compare body part orientations, normalizing all orientations relative to the spine.

4.6.2 Results

Figure 4.4 summarizes the results of our experiment. For the "easy" sequences A and B, the average errors were lower in general compared to the "hard" sequences C and D, consistent with our expectations.

For all sequences, those using our system were able to follow our teacher's motions more closely compared to the pre-test and those learning from a video.



(a) (b) (c)

Figure 4.5: (a) Textured full mesh reenactment. (b) Untextured full mesh reenactment. (c) Teacher skeleton reenactment.

In fact, those learning from the video barely changed from the pre-test. We consider that one factor could be the mirror self-correction factor. In the video, the user is not able to see their mistakes. On the other hand, while our system does not explicitly point out mistakes either, users are able to see the difference themselves. This allows them to adjust their motions to better copy the teacher's by observing the teacher from desired directions.

Simply the fact that they are able to see themselves allows this. In the future, we would like to see if the novel view synthesis truly has an effect on learning, for example by adding a splitscreen mirror panel to the video playback.

4.6.3 Survey

We asked the same users to try out 2 other reenactment methods: the untextured full mesh, and the skeleton of the teacher (Fig. 4.5). Finally, our users answered a survey consisting of 8 questions with the goal of evaluating the system's perceived ease of use, effectiveness, quality, and appeal (Fig. 4.6).

Table 4.1 summarizes our users' responses. Most users preferred the reenactment with a fully textured mesh for all questions, even for the equivalent video questions. This means that users found our system easy to use, effective at helping them learn actions, having high output quality, and most would use a similar system given the chance. Many users also appreciated the mirroring as it was more difficult to tell left from right by watching the video.

Part 1. For each reenactment (Full mesh with full textures, untextured full mesh, skeletons only) and the video:

Q1 Was the reenactment/video comprehensible?

Q2 Was it easy to learn the motions using the system/video?

Part 2. For each reenactment (Full mesh with full textures, untextured full mesh, skeletons only):

Q3 Did the reenactment have good quality?

Q4 Did the reenactment resemble the original video?

Q5 Was it easy to manipulate the viewpoint to your desired one?

Q6 Were the differences between yourself and the reenactment clear?

Q7 Did changing the viewpoint help you learn the action?

Q8 Would you use this system in the future?

Figure 4.6: Questions asked in our user study. Users answered from 1 (strongly disagree) to 5 (strongly agree).

4.7 Summary

We have proposed and implemented an augmented reality system for helping users learn actions. The actions are performed by a teacher, and the system reconstructs the body and motion of the teacher using two RGB-D sensors. Using the reconstruction, the system overlays reenactments, onto a screen which also mirrors the learner. Learners are then are able to control the viewpoint intuitively by moving their own body.

To evaluate our system, we conducted a user study and found that this system allows for easy comparisons between learner and teacher, and users were able to perform more accurate motions using the system than with video. They appreciated the ability to intuitively control the point of view while comparing motions,

Table 4.1: Users' averaged answers for the survey in Fig. 4.6, for full mesh with full textures (R1), untextured full mesh (R2), skeletons only (R3), and video (V). Users answered from 1 (strongly disagree) to 5 (strongly agree).

	R1	R2	R3	V
Q1	**4.11** $\pm$ 0.66	3.86 $\pm$ 0.77	2.29 $\pm$ 1.07	3.86 $\pm$ 0.77
Q2	**4.29** $\pm$ 0.73	3.71 $\pm$ 0.91	2.29 $\pm$ 0.83	2.93 $\pm$ 0.83
Q3	**4.00** $\pm$ 0.68	3.71 $\pm$ 0.99	2.64 $\pm$ 1.22	—
Q4	**4.50** $\pm$ 0.65	3.64 $\pm$ 1.08	2.50 $\pm$ 1.16	—
Q5	**3.93** $\pm$ 1.00	**3.93** $\pm$ 1.00	3.14 $\pm$ 1.29	—
Q6	**4.07** $\pm$ 1.21	3.50 $\pm$ 1.22	2.29 $\pm$ 1.33	—
Q7	**4.00** $\pm$ 0.96	3.79 $\pm$ 0.89	2.93 $\pm$ 1.14	—
Q8	**4.43** $\pm$ 0.85	3.57 $\pm$ 1.02	2.00 $\pm$ 1.11	—

which to our knowledge is unique to our system. Finally, in general our users preferred learning using the system over watching a video.

Future work can go in many directions. Currently, the system is rudimentary, requiring mouse input for all of its functions. With motion-based controls, the system could become something that consumers would legitimately want to use. We would also like to see the effect of different rendering methods of the teacher's reenactment. For example, instead of displacing the reenactment, what if it was instead overlayed directly on top of the user? Or, what if it was semi-transparent?

Also, as we discussed in the results, we would like to see if the increase in accuracy for our system came from being able to choose a novel viewpoint or merely because of the existence of the mirror. The mirror is currently the only way that the user can receive feedback from the system. The user study could be conducted again, but this time also adding a mirror to the conventional video portion.

Another feature that was often requested was direct feedback. During training, the user's body and the teacher's body would be directly compared on the spot, highlighting differences in a different color, for example. One other possibility for training that we'd like to see the effect of is training in a different context,

in a sort of “challenge mode”. In this mode, the system sets a view for the user instead of the user choosing a view for him or herself. This forces the user to adapt and hopefully remember the action better.

5 Conclusion

In this thesis, we have put forward the idea of *reenactments* to aid in action learning. Reenactments are motion sequences of a human performing an action that may be viewed from arbitrary viewpoints for increased ease of comprehending the action. Reenactments do not have to be perfectly true-to-life, but they must be plausible and comprehensible.

We have proposed two novel methods of capturing and generating reenactments. Both may be done using only a single RGB-D camera, for convenience and ease for general users.

The first method exploits human skeleton tracking from depth images as well as simultaneous localization and mapping systems in order to capture a motion sequence that is located in the world. The motion sequence is represented as a set of body parts over multiple frames. To render it, each body part is represented by a rigid shape, either a rough volume that approximates the body part's actual shape such as a cylinder, or by a more accurate voxel volume, acquired through voxel carving. To texture each shape, we make use of VDTM [25], using information over the entire RGB sequence.

We also implemented a mobile AR reenactment viewer that is able to display the reenactment. Using the mobile device as an AR see-through display, the learner is able to look through the display to see a "ghost" of the teacher. Since we make use of the same environment that we captured in, the learner is then able to intuitively move the device in order to be able to watch the motion sequence from the desired viewpoint.

We evaluated the system by performing user studies. First, we quantitatively evaluated the learners' ability to estimate using the reenactment. The learners scored higher with the system than with a perspective. Next, we qualitatively evaluated the output quality. The learners gave low scores for our this reen-

actment method's output quality, and we aimed to address this with the next method.

The second method is built upon the tensor-based human body model [20] which is a statistical body model that parameterizes human body shape as well as pose. When trained, the body model is able to output an appropriately deformed mesh given shape and pose parameters, and in the other direction, is able to output shape and pose parameters given a mesh. We are thus able to fit a model to an observation which may be incomplete, such as from our RGB-D sensor.

We also implemented an AR mirror-based motion viewer that implements the AR "mirror" metaphor to augment a mirror image of the learner with a view of the reenactment, so that the learner can easily copy the motion sequence. The reenactment view should mirror the learner's own, since this way it is easiest to compare pose. The learner is thus able to control the view by turning his or her body in the desired direction, which is easy to grasp and make use of. The system is based on the theory of motor learning, specifically kinematic knowledge of results, of which one way is to show the learner their own motion, along with the ideal motion. This is easily accomplished by overlaying the reenactment on the mirror image of the learner.

We evaluated the system by performing user studies, quantitatively by measuring motion accuracy and qualititatively by taking a survey. We measured motion accuracy by comparing the leaners' motions to the teacher's motions after a short training period the motion sequence, meaning either the RGB only video sequence or the reenactment. In all cases, learners who used the reenactment followed the teacher's motions more closely, demonstrating our system's usefulness. We then surveyed our system's percieved effectiveness, ease-of-use, and reenactment quality, with a generally favorable response.

We can conclude that AR-based reenactments are a worthwhile and effective way of learning actions. From here, this research can go in a number of interesting directions. One direction is in developing reenactment capture. Consumer RGB-D cameras are easier to acquire and use than the past methods of NVS, but they are still not as ubiquitous as, for example, conventional video capture devices, which are a staple of smartphones. If a reenactment could be created from such a video, it would be a large step towards mainstream use.

Another direction is in improving the output quality of the reenactment. Currently, reenactments are comprehensible and plausible. However, if, for example, movie-quality reenactments were available, it would widen the scope of reenactments to not just motion training but also entertainment. Imagine watching a movie from within the scene on your handheld reenactment viewer. Also, a wide database or reenactment sharing site, similar to video-sharing social networks such as YouTube, would be a boon to learners.

Other theories of learning can also be implemented and evaluated. For example, how do the different ways of kinematic knowledge of results benefit learning? A comparison can be done between different methods, e.g. showing just a mirror of the user, versus showing a mirror of the user with a reenactment overlaid, versus explicitly giving feedback on each step of the motion, whether automatically or done by experts. Retention of the knowledge is also significant when measuring learning; for this purpose, we may do longer-term experiments.

Acknowledgements

This thesis would not have been possible without the hard work of my thesis supervisors: Professors Naokazu Yokoya, Hirokazu Kato, Tomokazu Sato, and Yuta Nakashima. They basically shaped the thesis into what it is today. Thank you for the numerous lab meetings which gave me direction, the insights that offered me new perspectives, and the corrections which were hard but necessary. Additionally, I performed the bulk of the work in Professor Yokoya's laboratory. I am grateful to him for admitting me. Yokoya-lab closes this year, which I am quite sad about; however, I know that Professor Yokoya will do an excellent job as NAIST president.

Thank you to the teachers and staff of NAIST, for making me feel welcome at a new university and a new country. Professor Hiroyuki Seki, Ms. Ayako Ohta, and Mr. Norito Hamada were my first contacts, and they made a warm and welcoming first impression. Mr. Tadashi Nakano, Ms. Haruna Hatoyama, Ms. Rika Sunamoto, Ms. Kaori Kamiya, and Ms. Yuko Sumitani from the International Student Office were also very helpful throughout my stay. Ms. Yumi Ishitani and Ms. Azusa Minami also tirelessly worked to take care of all my documents, giving me a smooth ride.

I would also like to thank the people at Microsoft Research Asia, who offered me an internship, and the rest of the MSCORE project team: Mr. Ambrosio Blanco, Professor Katsushi Ikeuchi, Professor Hiroshi Kawasaki, and Ryosuke Kimura. MSRA offered me a great environment for research and a great opportunity to produce something worthwhile. Even after the internship, they continued to work closely with me through countless online meetings.

I must also thank the teachers at the Ateneo de Manila, especially at DISCS: Professor Regina Estuar, my undergraduate thesis adviser, Professor Jon Fernandez, Professor Didith Rodrigo, Professor John Paul Vergara, Professor Jessica Sugay, among others. I still remember all your lessons! Thank you for setting me on this road, nine years later.

I would also like to thank my upperclassmen at Yokoya-lab: Hideyuki Kume, Fumio Okura, and Takahito Aoto, for leading the way. I would also be remiss if I did not mention my fellow PhD candidates Antonio Tejero de Pablos, Hikari

Takehara, and Mayu Otani, for journeying together with me. Thanks for sitting through all those lab meetings and for the suggestions and feedback.

Thanks to my friends at NAIST and back home! You are great!

Finally, to my loving family, thank you for your continued support through all these years.

All things are possible through God.

References

[1] J. Adams. Historical review and appraisal of research on the learning, retention, and transfer of human motor skills. *Psychological Bulletin*, 101(1):41–74, 1987.

[2] D. Alexiadis, D. Zarpalas, and P. Daras. Real-time, full 3D reconstruction of moving foreground objects from multiple consumer depth cameras. *IEEE Trans. Multimedia*, 15(2):339–358, 2013.

[3] D. Amselem. A window on shared virtual environments. *Presence: Teleoperators and Virtual Environments*, 4(2):130–145, 1995.

[4] F. Anderson, T. Grossman, J. Matejka, and G. Fitzmaurice. YouMove: Enhancing movement training with an augmented reality mirror. In *Proc. ACM Symposium on User Interface Software and Technology*, pages 311–320, 2013.

[5] D. Anguelov, D. Koller, H.-C. Pang, P. Srinivasan, and S. Thrun. Recovering articulated object models from 3D range data. In *Proc. Conf. Uncertainty in Artificial Intelligence*, pages 18–26, 2004.

[6] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. SCAPE: Shape completion and animation of people. *ACM Trans. Graphics*, 24(3):408–416, 2005.

[7] R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 1997.

[8] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, 21(6):34–47, 2001.

[9] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Consumer Depth Cameras for Computer Vision*, pages 71–98. 2013.

[10] A. Bauer, A.-H. Dicko, F. Faure, O. Palombi, and J. Troccaz. Anatomical mirroring: real-time user-specific anatomy in motion using a commodity depth camera. In *Proc. Int. Conf. Motion in Games*, pages 113–122, 2016.

[11] A. Bauer, A.-H. Dicko, O. Palombi, F. Faure, and J. Troccaz. Living book of anatomy project: See your insides in motion! In *Proc. Emerging Technologies*, pages 1–4, 2015.

[12] M. Billinghurst. Augmented reality in education. *New Horizons for Learning*, 9(1):1–5, 2002.

[13] T. Blum, V. Kleeberger, C. Bichlmeier, and N. Navab. mirracle: An augmented reality magic mirror system for anatomy education. In *Proc. IEEE Virtual Reality Workshops*, pages 115–116, 2012.

[14] F. Bogo, M.J. Black, M. Loper, and J. Romero. Detailed full-body reconstructions of moving people from monocular RGB-D sequences. In *Proc. IEEE Int. Conf. Computer Vision*, pages 2300–2308, 2015.

[15] J. Caarls, P. Jonker, Y. Kolstee, J. Rotteveel, and W. van Eck. Augmented reality for art, design and cultural heritage system design and evaluation. *EURASIP Journal on Image and Video Processing*, 2009:1–16, 2010.

[16] C. Cagniart, E. Boyer, and S. Ilic. Free-form mesh tracking: a patch-based approach. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 1339–1346, 2010.

[17] C. Cagniart, E. Boyer, and S. Ilic. Probabilistic deformable surface tracking from multiple videos. In *Proc. European Conf. Computer Vision*, pages 326–339, 2010.

[18] J. Carranza, C. Theobalt, M. Magnor, and H. Seidel. Free-viewpoint video of human actors. *ACM Trans. Graphics*, 22(3):569–577, 2003.

[19] R. Castle, G. Klein, and D. Murray. Video-rate localization in multiple maps for wearable augmented reality. In *Proc. IEEE Int. Symp. Wearable Computers*, pages 15–22, 2008.

[20] Y. Chen, Z. Liu, and Z. Zhang. Tensor-based human body modeling. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 105–112, 2013.

[21] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. Meshlab: an open-source mesh processing tool. In *Proc. Eurographics Italian Chapter Conf.*, pages 129–136, 2008.

[22] W.B. Culbertson, T. Malzbender, and G. Slabaugh. Generalized voxel coloring. In *Proc. Int. Workshop on Vision Algorithms: Theory and Practice*, pages 100–115, 2000.

[23] B. Dai and X. Yang. A low-latency 3D teleconferencing system with image based approach. In *Proc. ACM SIGGRAPH Int. Conf. Virtual-Reality Continuum and Its Applications in Industry*, pages 243–248, 2013.

[24] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H. Seidel, and S. Thrun. Performance capture from sparse multi-view video. *ACM Trans. Graphics*, 27(3):1–10, 2008.

[25] P. Debevec, C. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proc. ACM SIGGRAPH*, pages 11–20, 1996.

[26] M. Dou, H. Fuchs, and J.-M. Frahm. Scanning and tracking dynamic objects with commodity depth cameras. In *Proc. IEEE Int. Symposium on Mixed and Augmented Reality*, pages 99–106, 2013.

[27] M. Dou, S. Khamis, Y. Degtyarev, P. Davidson, S.R. Fanello, A. Kowdle, S.O. Escolano, C. Rhemann, D. Kim, J. Taylor, P. Kohli, V. Tankovich, and S. Izadi. Fusion4d: Real-time performance capture of challenging scenes. *ACM Trans. Graphics*, 35(4):1–13, 2016.

[28] M. Dou, J. Taylor, H. Fuchs, A. Fitzgibbon, and S. Izadi. 3D scanning deformable objects with a single RGB-D sensor. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 493–501, 2015.

[29] M. Dunleavy and C. Dede. Augmented reality teaching and learning. In *Handbook of Research on Educational Communications and Technology*, pages 735–745. 2014.

[30] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster. A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment. *Personal Technologies*, 1(4):208–217, 1997.

[31] M. Fiala. Magic mirror system with hand-held and wearable augmentations. In *Proc. IEEE Conf. Virtual Reality*, pages 251–254, 2007.

[32] J. Gall, C. Stoll, E. De Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel. Motion capture using joint skeleton tracking and surface estimation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 1746–1753, 2009.

[33] V. Ganapathi, C. Plagemann, D. Koller, and S.Sebastian Thrun. Real time motion capture using a single time-of-flight camera. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 755–762, 2010.

[34] J. Gausemeier, J. Fruend, C. Matysczok, B. Bruederlin, and D. Beier. Development of a real time image based object recognition method for mobile AR-devices. In *Proc. Int. Conf. Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, pages 133–139, 2003.

[35] S. Giovanni, Y.C. Choi, J. Huang, E.T. Khoo, and K. Yin. Virtual try-on using Kinect and HD camera. In *Proc. Int. Conf. Motion in Games*, pages 55–65, 2012.

[36] M. Gleicher. Animation from observation: Motion capture and motion editing. *ACM SIGGRAPH Computer Graphics*, 33(4):51–54, 1999.

[37] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H.-P. Seidel. A statistical model of human pose and body shape. *Computer Graphics Forum*, 28(2):337–346, 2009.

[38] S. Hauswiesner, M. Straka, and G. Reitmayr. Free viewpoint virtual try-on with commodity depth cameras. In *Proc. Int. Conf. Virtual Reality Continuum and Its Applications in Industry*, pages 23–30, 2011.

[39] S. Henderson and S. Feiner. Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret. In *Proc. IEEE Int. Symp. Mixed and Augmented Reality*, pages 135–144, 2009.

[40] S. Henderson and S. Feiner. Augmented reality in the psychomotor phase of a procedural task. In *Proc. IEEE Int. Symp. Mixed and Augmented Reality*, pages 191–200, 2011.

[41] A. Henrysson, M. Billinghurst, and M. Ollila. Face to face collaborative AR on mobile phones. In *Proc. IEEE and ACM Int. Symp. Mixed and Augmented Reality*, pages 80–89, 2005.

[42] M. Hofmann and D.M. Gavrila. Multi-view 3D human pose estimation in complex environment. *International Journal of Computer Vision*, 96(1):103–124, 2012.

[43] T. Höllerer and S. Feiner. Mobile augmented reality. *Telegeoinformatics: Location-Based Computing and Services*, 21:1–39, 2004.

[44] M. Innmann, M. Zollhöfer, M. Nießner, C. Theobalt, and M. Stamminger. Volumedeform: Real-time volumetric non-rigid reconstruction. In *European Conf. Computer Vision*, pages 362–379, 2016.

[45] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera. In *Proc. ACM Symp. User Interface Software and Technology*, pages 559–568, 2011.

[46] A. Jain, T. Thormählen, H.-P. Seidel, and C. Theobalt. Moviereshape: Tracking and reshaping of humans in videos. *ACM Trans. Graphics*, 29(6):1–9, 2010.

[47] B. Kainz, S. Hauswiesner, G. Reitmayr, M. Steinberger, R. Grasset, L. Gruber, E. Veas, D. Kalkofen, H. Seichter, and D. Schmalstieg. OmniKinect: Real-time dense volumetric data acquisition and applications. In *Proc. ACM Symp. Virtual Reality Software and Technology*, pages 25–32, 2012.

[48] A.R. Kancherla, J.P. Rolland, D.L. Wright, and G. Burdea. A novel virtual reality tool for teaching dynamic 3D anatomy. In *Proc. Computer Vision, Virtual Reality and Robotics in Medicine*, pages 163–169, 1995.

[49] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proc. IEEE and ACM Int. Workshop on Augmented Reality*, pages 85–94, 1999.

[50] G.J. Kim and A. Rizzo. A SWOT analysis of the field of virtual reality rehabilitation and therapy. *Presence: Teleoperators and Virtual Environments*, 14(2):119–146, 2005.

[51] A. Kirk, J. O'Brien, and D. Forsyth. Skeletal parameter estimation from optical motion capture data. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 782–788, 2005.

[52] D.A. Kolb. *Experiential learning: Experience as the source of learning and development*. 2014.

[53] D.Y. Kwon and M. Gross. Combining body sensors and visual sensors for motion training. In *Proc. ACM SIGCHI Int. Conf. Advances in Computer Entertainment Technology*, pages 94–101, 2005.

[54] Y. Liu, J. Gall, C. Stoll, Q. Dai, H.-P. Seidel, and C. Theobalt. Markerless motion capture of multiple characters using multiview image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 35(11):2720–2735, 2013.

[55] W. Lorensen and H. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proc. ACM SIGGRAPH*, pages 163–169, 1987.

[56] L. Maletsky, J. Sun, and N. Morton. Accuracy of an optical active-marker system to track the relative motion of rigid bodies. *Journal of biomechanics*, 40(3):682–685, 2007.

[57] C. Malleson, M. Klaudiny, A. Hilton, and J.-Y. Guillemaut. Single-view RGB-D-based reconstruction of dynamic human geometry. In *Proc. Int. Workshop on Dynamic Shape Capture and Analysis*, pages 307–314, 2013.

[58] T. Matsuyama and T. Takai. Generation, visualization, and editing of 3D video. In *Proc. Int. Symp. 3D Data Processing Visualization and Transmission*, pages 234–245, 2002.

[59] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan. Image-based visual hulls. In *Proc. ACM SIGGRAPH*, pages 369–374, 2000.

[60] M. Meng, P. Fallavollita, T. Blum, U. Eck, C. Sandor, S. Weidert, J. Waschke, and N. Navab. Kinect for interactive AR anatomy learning. In *Proc. IEEE Int. Symp. Mixed and Augmented Reality*, pages 277–278, 2013.

[61] Y. Meng, P.Y. Mok, and X. Jin. Interactive virtual try-on clothing design systems. *Computer-Aided Design*, 42(4):310–321, 2010.

[62] J. Mercier-Ganady, F. Lotte, E. Loup-Escande, M. Marchal, and A. Lécuyer. The mind-mirror: See your brain in action in your head using eeg and augmented reality. In *IEEE Conf. Virtual Reality*, pages 33–38, 2014.

[63] M. Mohring, C. Lessig, and O. Bimber. Video see-through AR on consumer cell-phones. In *Proc. IEEE and ACM Int. Symp. Mixed and Augmented Reality*, pages 252–253, 2004.

[64] K. Müller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand. View synthesis for advanced 3D video systems. *EURASIP Journal on Image and Video Processing*, 2008(1):1–11, 2009.

[65] A. Murai, K. Kurosaki, K. Yamane, and Y. Nakamura. Musculoskeletal-see-through mirror: Computational modeling and algorithm for whole-body muscle activity visualization in real time. *Progress in Biophysics and Molecular Biology*, 103(2):310–317, 2010.

[66] R. Newcombe, D. Fox, and S. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 343–352, 2015.

[67] J. Newman, D. Ingram, and A. Hopper. Augmented reality in a wide area sentient environment. In *Proc. IEEE and ACM Int. Symp. Augmented Reality*, pages 77–86, 2001.

[68] H.T. Regenbrecht and R. Specht. A mobile passive augmented reality device-mPARD. In *Proc. IEEE and ACM Int. Symp. Augmented Reality*, pages 81–84, 2000.

[69] J. Rekimoto. Transvision: A hand-held augmented reality system for collaborative design. In *Proc. Virtual Systems and Multimedia*, pages 85–90, 1996.

[70] J. Rekimoto. Navicam: A magnifying glass approach to augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):399–412, 1997.

[71] J. Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. In *Proc. Computer Human Interaction*, pages 63–68, 1998.

[72] H. Rhodin, N. Robertini, D. Casas, C. Richardt, H.-P. Seidel, and C. Theobalt. General automatic human shape and motion capture using volumetric contour cues. In *Proc. European Conf. Computer Vision*, pages 509–526, 2016.

[73] M. Rohs and B. Gfeller. Using camera-equipped mobile phones for interacting with real-world objects. In *Proc. Advances in Pervasive Computing*, pages 265–271, 2004.

[74] A. Salmoni, R. Schmidt, and C. Walter. Knowledge of results and motor learning: a review and critical reappraisal. *Psychological Bulletin*, 95(3):355–386, 1984.

[75] M.E. Santos, A. Chen, T. Taketomi, G. Yamamoto, J. Miyazaki, and H. Kato. Augmented reality learning experiences: Survey of prototype design and evaluation. *IEEE Trans. Learning Technologies*, 7(1):38–56, 2014.

[76] M.E. Santos, A. Chen, M. Terawaki, G. Yamamoto, T. Taketomi, J. Miyazaki, and H. Kato. Augmented reality X-ray interaction in K-12 education: Theory, student perception and teacher evaluation. In *Proc. IEEE Int. Conf. Advanced Learning Technologies*, pages 141–145, 2013.

[77] D. Schmalstieg and D. Wagner. Experiences with handheld augmented reality. In *Proc. IEEE and ACM Int. Symp. Mixed and Augmented Reality*, pages 3–18, 2007.

[78] F. Shibata, T. Hashimoto, K. Furuno, A. Kimura, and H. Tamura. Scalable architecture and content description language for mobile mixed reality systems. In *Advances in Artificial Reality and Tele-Existence*, pages 122–131. 2006.

[79] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[80] J. Starck, G. Miller, and A. Hilton. Video-based character animation. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 49–58, 2005.

[81] C. Stoll, N. Hasler, J. Gall, H.-P. Seidel, and C. Theobalt. Fast articulated motion tracking using a Sums of Gaussians body model. In *Proc. IEEE Int. Conf. Computer Vision*, pages 951–958, 2011.

[82] R. Tang, X.-D. Yang, S. Bateman, J. Jorge, and A. Tang. Physio@ home: Exploring visual guidance and feedback techniques for physiotherapy exer-

cises. In *Proc. ACM Conf. Human Factors in Computing Systems*, pages 4123–4132, 2015.

[83] J. Tong, .J Zhou, L. Liu, Z. Pan, and H. Yan. Scanning 3D full human bodies using Kinects. *IEEE Trans. Visualization and Computer Graphics*, 18(4):643–650, 2012.

[84] S. Tsuchida, T. Terada, and M. Tsukamoto. A system for practicing formations in dance performance supported by self-propelled screen. In *Proc. Augmented Human Int. Conf.*, pages 178–185, 2013.

[85] D.W.F. Van Krevelen and R. Poelman. A survey of augmented reality technologies, applications and limitations. *International Journal of Virtual Reality*, 9(2):1–21, 2010.

[86] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. *ACM Trans. Graphics*, 27(3):1–9, 2008.

[87] D. Wagner and I. Barakonyi. Augmented reality kanji learning. In *Proc. IEEE and ACM Int. Symp. Mixed and Augmented Reality*, pages 1–2, 2003.

[88] D. Wagner and D. Schmalstieg. First steps towards handheld augmented reality. In *Proc. IEEE Int. Symp. Wearable Computers*, pages 1–9, 2003.

[89] M. Waschbüsch, S. Würmlin, D. Cotting, F. Sadlo, and M. Gross. Scalable 3D video of dynamic scenes. *The Visual Computer*, 21(8):629–638, 2005.

[90] A. Weiss, D. Hirshberg, and M.J. Black. Home 3D body scans from noisy image and range data. In *Proc. IEEE Int. Conf. Computer Vision*, pages 1951–1958, 2011.

[91] S. Wuhrer, L. Pishchulin, A. Brunton, C. Shu, and J. Lang. Estimation of human body shape and posture under clothing. *Computer Vision and Image Understanding*, 127:31–42, 2014.

[92] S. Würmlin, E. Lamboray, O. Staadt, and M. Gross. 3D video recorder. In *Proc. Pacific Conf. Computer Graphics and Applications*, pages 325–334, 2002.

[93] J. Yang, J.-S. Franco, F. Hétroy-Wheeler, and S. Wuhrer. Estimation of human body shape in motion with wide clothing. In *Proc. European Conf. Computer Vision*, pages 439–454, 2016.

[94] U. Yang and G. Kim. Implementation and evaluation of "Just Follow Me": An immersive, VR-based, motion-training system. *Presence: Teleoperators and Virtual Environments*, pages 304–323, 2002.

[95] G. Ye, Y. Liu, Y. Deng, N. Hasler, X. Ji, Q. Dai, and C. Theobalt. Free-viewpoint video of human actors using multiple handheld Kinects. *IEEE Trans. Cybernetics*, 43(5):1370–1382, 2013.

[96] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *Proc. IEEE Int. Conf. Computer Vision*, pages 731–738, 2011.

[97] M. Ye and R. Yang. Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2345–2352, 2014.

[98] S. Yuen, G. Yaoyuneyong, and E. Johnson. Augmented reality: An overview and five directions for AR in education. *Journal of Educational Technology Development and Exchange*, 4(1):119–140, 2011.

[99] C. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graphics*, 23(3):600–608, 2004.

[100] M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Trans. Graphics*, 33(4):1–12, 2014.

Publication List

Journal Papers

1. Fabian Lorenzo Dayrit, Yuta Nakashima, Tomokazu Sato, Naokazu Yokoya. Increasing pose comprehension through augmented reality reenactment. Multimedia Tools and Applications vol. 76(1), pages 1291–1312. December 2015. (Section 3.4)

International Conferences

1. Fabian Lorenzo Dayrit, Ryosuke Kimura, Yuta Nakashima, Ambrosio Blanco, Hiroshi Kawasaki, Katsushi Ikeuchi, Tomokazu Sato, Naokazu Yokoya. ReMagicMirror: Action Learning using Human Reenactment with the Mirror Metaphor. International Conference on Multimedia Modeling, pages 303–315. January 2017. (Section 4)

2. Fabian Lorenzo Dayrit, Yuta Nakashima, Tomokazu Sato, Naokazu Yokoya. Free-viewpoint AR human-motion reenactment based on a single RGB-D video stream. IEEE International Conference on Multimedia and Expo, 6 pages. July 2014. (Section 3.3)

Domestic Conferences

1. Fabian Lorenzo Dayrit, Yuta Nakashima, Tomokazu Sato, Naokazu Yokoya. Single RGB-D video stream-based human-motion reenactment. 映像情報メディア学会. 10 pages. February 2014. (Section 3.3)