Estimating the Pose of a Medical Manikin for Haptic Augmentation of a Virtual Patient in Mixed Reality Training

David Scherfgen

Jonas Schild

david.scherfgen@hs-hannover.de

jonas.schild@hs-hannover.de

Interactive Reality Experiences, Hanover University of Applied Sciences and Arts
Hanover, Germany

Figure 1: Estimating the pose of a manikin via deep learning-based tracking can be used for haptic support in medical training.

ABSTRACT

Virtual medical emergency training provides complex while safe interactions with virtual patients. Classical non-virtual medical training includes the use of medical manikins. Haptically integrating such a manikin into virtual training has the potential to improve the interaction with a virtual patient (e.g. regarding social presence) and the resulting training experience. This work analyzes user requirements and several technological approaches. The presented system estimates the pose of a medical manikin in order to haptically augment a 3D human model in a virtual reality (VR) training environment, allowing users to physically touch a virtual patient. The system uses a Convolutional Neural Networks-based (CNN) body keypoint detector to locate relevant keypoints of the manikin in the images of the stereo camera built into a head-mounted display (HMD). The system retrieves manikin position, orientation and joint angles using non-linear optimization. A manual precision analysis reports a mean error distance of 48 mm. Unless further optimized, we recommend applying this method for haptically augmented social interaction with virtual patients but not for practicing precise medical treatments.

CCS CONCEPTS

• **Human-centered computing** → **Mixed / augmented reality**; **Virtual reality**.

KEYWORDS

haptics, mixed reality, virtual medical training, medical manikin, body pose estimation

ACM Reference Format:

David Scherfgen and Jonas Schild. 2021. Estimating the Pose of a Medical Manikin for Haptic Augmentation of a Virtual Patient in Mixed Reality Training. In *Symposium on Virtual and Augmented Reality (SVR'21), October 18–21, 2021, Virtual Event, Brazil*. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3488162.3488166>

1 INTRODUCTION

Medical emergency training through means of VR allows users to safely train relevant medical aspects like diagnostics and treatment of varying patient conditions and even surgical procedures in complex situations. In comparison to classical training environments with medical training manikins, a virtual patient can display varying visual symptoms (e.g. skin color changes, rash, wounds, blood flow, shiver, sweat, facial emotions, etc.) that can be altered through parameters, resulting in rich and complex scenarios of rare and critical medical cases. With existing systems, the overall training experience has shown to correlate with the experienced presence and usability [15]. The use of Mixed Reality (MR) technology may improve presence and realism [10], potentially contributing to better learning and integration into existing medical training facilities [19].

An MR approach requires mixing an existing training environment with a medical manikin, medical equipment, cables, wires,

This work is licensed under a Creative Commons Attribution International 4.0 License.

SVR'21, October 18–21, 2021, Virtual Event, Brazil

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-9552-6/21/10.

<https://doi.org/10.1145/3488162.3488166>

hoses, diagnostic instruments, and consumables for treatment (e.g. patches, syringes, etc.) with a virtual training scenario (e.g. virtual location, virtual patient persona, virtual tools). Basically, two approaches can contribute to a solution: Augmented Reality (AR) and Augmented Virtuality (AV).

In an AR approach, the manikin could be overlaid with visual symptoms or emotions using a see-through HMD (e.g. Microsoft HoloLens). Such displays currently support only limited fields of view and the approach would emphasize the haptic use of material which the simulation would hence need to track in order to support training analysis. On the other hand, in an AV approach, users could experience a virtual scenario through a fully immersive HMD and haptically feel and use selected objects whose real-world counterparts need to be tracked. Information regarding what tools or instruments the trainees apply is already available within the simulation, which facilitates training data analysis.

This work follows the AV approach. We focus on social interaction with a virtual patient where several previous experiments and prototypes have found positive effects on perceived agency and co-presence [12] or empathy [14], further proposing that observing an empathetic reaction from a haptic touch may lead to an increased engagement in training [18].

There are technological challenges involved with previously described prototypes [5, 9], requiring a systematic approach to enable haptic interaction between trainees and a virtual patient embodied by a medical manikin that is present in the same physical space. If the virtual patient closely matches the manikin with respect to location, orientation, body pose and body geometry, then a convincing haptic experience can be achieved, e.g. when touching the patient, assuming the existence of some form of hand tracking.

As contributions, this work:

- describes requirements on the system capabilities and the problem decomposition,
- lists and discusses possible technical approaches,
- constructs a design and implementation for body pose estimation via optical keypoint detection,
- analyzes precision and technical effectiveness of our implementation.

2 COMBINING A PHYSICAL MANIKIN AND A VIRTUAL PATIENT THROUGH MIXED REALITY

2.1 Related systems

There are several approaches that partially implement haptic interaction with a simulation manikin by only using selected body parts. To name a few, Hochreiter et al. [11] and Daher [6] projected the face of a virtual patient onto a translucent and touch-sensitive head-shaped shell in an optical see-through AR application with the aim to physically interact with the patient's head. However, this approach does not allow full body interaction nor conducting a complex virtual training scenario. Almousa et al. [2] integrated a half-torso manikin into an MR training environment with gamification elements to train cardiopulmonary resuscitation.

Only a few studies comprise a complex training prototype in a fully immersive virtual environment and simultaneously integrate a full body simulation manikin. Hanzaki et al. [10] statically combined a simulation manikin with a virtual character and compared this setting in an MR and VR environment. The MR scenario achieved higher believability scores and higher realistic experience scores. Their system relied on VIVE trackers to track only a single arm of the manikin. Girau et al. [9] created a virtual counterpart of a simulation manikin by scanning a real person with photogrammetry technology. They integrated the physical manikin and virtual character at the same position inside the MR environment. Leap Motion hand tracking was used to allow for interaction with the virtual patient. In a user study, medical students and laypeople reported high presence scores (spatial presence, involvement and experienced realism). However, the process of adjusting the virtual patient's pose to match the manikin was a completely manual one. Chirico et al. [5] extended the prototype of Girau et al. by integrating critical emergencies scenarios, e.g., first-aid or choking. These scenarios allow the training of life support procedures and basic skills that require haptic feedback.

2.2 System requirements

A manikin tracking system suitable for haptically augmenting a virtual patient is required to accurately determine the medical manikin's location and orientation in space as well as its body pose, which can be described by the rotation angles of the mechanical joints connecting its limbs. The VR application can then apply the corresponding transformations to the 3D model that represents the patient. In more detail, the solution should meet the requirements on the system capabilities described in the following.

Quantitative requirements:

- *Accuracy*: The parameters measured by the system should be accurate enough to allow for a convincing haptic experience. That means that when the user touches the manikin with their hand, then the virtual representation of their hand should also touch the virtual patient, and vice versa. (It is difficult to exactly define what is "accurate enough", as it depends on the concrete tasks to be performed by the trainees.)
- *Low cost*: Ideally, no additional equipment should be required. The system should work with a typical VR setup, i.e. a tracked HMD with an integrated camera, and VR controllers.
- *Interactive update rates*: The system should ideally be able to measure the manikin's parameters at interactive rates, making it possible for the trainees to perform physical manipulations, e.g. putting the patient into recovery position.

Qualitative requirements:

- *Ease of use and low complexity*: If the system is to be integrated into educational practice, the system should be easy to set up and use. A complicated setup process would increase the entire training system's complexity and introduce new points of failure.
- *Versatility*: The system should be adaptable to work with a variety of medical manikins, which can differ in body proportions and degrees of freedom. Also, clothes worn by the manikin should not cause the system to fail.

2.3 Problem decomposition

Medical manikins usually consist of several body parts or limbs made of plastic that are connected by mechanical joints, allowing for rotation around one axis (hinge joints) or free rotation (ball joints). Some parts such as fingers, face and neck are often made of a deformable material (silicone).

In computer graphics, a well-established representation of a human body is a *skinned mesh*. The mesh captures the geometry of the body in a neutral pose (“T pose”). Inside the mesh, a skeleton is formed by a hierarchical structure of bones, similarly to the human skeleton, although usually with less detail. Bones can only be rotated, not translated, relatively to each other. Limits can be imposed on the rotation angles in order to exclude unnatural poses (for example, the elbow joint only permits the lower arm to be rotated inwards, not outwards). Each vertex of the mesh is influenced by at least one bone and moves along with it as the bone is rotated, simulating the connection between flesh and bones. Vertices located in proximity to joints are usually influenced by the two bones connected by the joint, which results in a soft deformation of the mesh when the bones are rotated.

The problem to be solved is to capture a physical manikin’s pose and apply it to the 3D model of a virtual patient. We decompose the problem into three sub-problems:

- (1) Creating a 3D model for the virtual patient that closely resembles the manikin with respect to body proportions and geometry. Since most medical manikins look somewhat unnatural, compromises must be made. A virtual patient modeled exactly after the manikin will look like a manikin in VR as well.
- (2) Determining body pose of the manikin, i.e. the relative rotations between the bones.
- (3) Determining absolute position and orientation of the manikin with respect to the VR tracking system’s coordinate space.

The remainder of this work will focus on the second and third sub-problem.

2.4 Manikin pose estimation approaches

In this section, we will present some of the solution approaches that were considered and partially explored in prototypes to briefly discuss their advantages and drawbacks.

2.4.1 Tracking limbs using VIVE trackers. The SteamVR tracking system (“lighthouses”) supports standalone VIVE trackers that can be attached to the limbs of the manikin. VIVE trackers employ the same tracking technology as VIVE controllers and HMDs do. This has been implemented in [10], although only for a single movable limb. This approach benefits from a high precision and low latency of the tracking system. On the other hand, the number of trackers required (at least one per movable part) and the dimensions of the trackers limit practicability of this approach. With fewer trackers, calculating inverse kinematics for an appropriate body pose results in multiple ambiguous solutions. Users unintentionally touching the trackers seems likely. Each tracker must also be registered precisely in relation to its respective body part, and that procedure would have to be repeated each time after a tracker is replaced or

(accidentally) moved. Moreover, it is necessary to charge the tracker batteries frequently.

2.4.2 Measuring joint angles using IMUs. Another approach is to measure the joint angles, i.e. the relative rotations between the limbs, by attaching inertial measurement units (IMUs). The position of the manikin as a whole would have to be determined in a different way. While IMUs are much more compact than a VIVE tracker, they tend to suffer from drift (accumulation of small errors over time) or can be disturbed by magnetic fields. They also must be charged regularly.

2.4.3 Tracking the manikin using a depth sensor. Depth sensors such as the Microsoft Kinect often come with software libraries for detecting human bodies and fitting 3D skeletons to them. A medical manikin that looks sufficiently human can therefore be tracked by such a sensor (see [17] for a detailed evaluation of the Kinect’s pose estimation performance). However, depth sensors can interfere with the SteamVR tracking system when they use infrared light of a similar wavelength. It may also prove difficult to find a suitable position for the depth sensor so that it has an unobstructed view of the manikin, especially when users are standing close to it.

2.4.4 Tracking limbs using fiducial markers. Fiducial markers, such as ARTags [8], are frequently used for tracking objects in AR applications. In order to gather spatial information, each of the manikin limbs must be equipped with several markers, and the markers must be precisely registered with the model.

The markers are very cheap to produce, as they are merely adhesive stickers, and ready-to-use solutions exist for detecting and tracking them in camera images.

Issues to be expected are noise-related errors due to poor illumination conditions, motion blur, or occlusions. In order to uniquely identify each marker, a code must be embedded. Thus, the markers need to be of a certain size to guarantee that the code can be reliably read even from some distance and under non-ideal conditions. Attaching markers of several centimeters in size to a medical manikin can prove difficult due to the lack of flat surfaces. Additionally, adhesive stickers don’t stick well to silicone. Finally, this approach doesn’t work well when the manikin is wearing clothes, as they could potentially move.

2.4.5 Optical tracking of the manikin via body keypoint detection. The advent of *Deep Learning* and *Convolutional Neural Networks* (CNNs) has led to considerable progress in image-based human body pose estimation. Using a technique called “Part Affinity Fields”, which is implemented in the open-source OpenPose [4] library, CNNs can reliably identify the pixel coordinates in an image where certain *keypoints* of a human body are located. Keypoints are prominent body features such as eyes, nose, shoulders, elbows, knees, or ankles. After training the CNN on a large set of annotated images, it can process images of humans differing in height, age, gender, skin color, clothing, body pose, etc.

Pose detection CNNs can successfully identify keypoints on human-like medical manikins. Most contemporary HMDs feature at least one integrated camera that can provide the input images to be fed into the CNN. The main advantage of this approach is that it does not require any additional equipment, and there is no need to physically alter the manikin. Clothing also does not pose a problem.

However, estimating the manikin's 3D location, rotation and its joint angles using only 2D keypoint locations is a non-trivial task.

3 BODY POSE ESTIMATION VIA OPTICAL KEYPOINT DETECTION

In this work, we follow the approach of *pose estimation via body keypoint detection*. The goal is to design a system which processes camera images from the HMD's point of view, directed at the manikin, in order to find location, rotation and joint angles of the manikin. After giving a high-level overview of the different processing steps, each step is described in detail.

3.1 Assumptions and overview

We assume that the HMD is equipped with at least one camera (we use the Valve Index HMD, which features a stereo camera). Further, we assume that the cameras are rigidly fixed to the HMD and that the HMD is accurately tracked by some internal or external tracking system, e.g. by the SteamVR tracking system. The manikin is assumed to look human, and its virtual counterpart should feature similar body proportions. Lighting conditions must allow the cameras to capture images of sufficient quality.

- *Preparation of the virtual patient model (off-line)*: For each body keypoint, an invisible *dummy object* is manually inserted into the virtual bone hierarchy at the correct location. For each bone of the model that corresponds to one of the manikin's movable limbs, we manually define the minimum/maximum permitted rotation angles for each axis, according to the limits of the respective mechanical joint.
- *Camera calibration (off-line)*: In a preprocessing step, the HMD cameras are calibrated intrinsically (determining parameters such as focal length and distortion coefficients) and extrinsically in relation to each other (determining the translational and rotational offset between each pair of cameras). Using the HMD tracking system, the camera positions and orientations in relation to the HMD are found, and we estimate the average time shift between tracking system data and camera image data.
- *Body keypoint detection (on-line)*: At runtime, camera images are processed by a body keypoint detector, resulting in the 2D pixel coordinates of detected keypoints and confidence score values. For each such observation, we also store position and orientation of the reference camera, using the data reported by the HMD tracking system and the calibration parameters determined earlier.
- *Initial estimation of position and orientation (on-line)*: Once a certain number of observations have been collected (fewer for real-time mode, more for a single, more accurate measurement), we estimate each body keypoint's 3D location using triangulation. We then find a translation and rotation that transform the keypoints of the virtual patient model (in some default pose) to the observed 3D locations with minimal error. This results in an initial estimate of the manikin's position and orientation.
- *Non-linear optimization of parameters (on-line)*: Finally, we apply non-linear optimization to further refine the manikin's position and orientation and determine rotation angles for

each joint, minimizing the reprojection error of the keypoints in the camera images, weighted by detection confidence scores.

3.2 Preparation of the virtual patient model

For our algorithm, it is important to know for each body keypoint supported by the keypoint detector (e.g. the 25 keypoints used by OpenPose) where that corresponding keypoint is located in the virtual patient model. The keypoints are manually inserted as invisible objects into the bone hierarchy (see Figure 2) in such a way that their respective location remains correct when bones are rotated. For example, the "nose" keypoint should be a child of the "head" bone. Each object stores the ID of the keypoint that it represents, e.g. "nose", "left elbow", "right knee", and so on. Keypoints should be placed close to where the keypoint detector would detect them if it was given a rendered image of the virtual patient model.

Next, the bones that correspond to one of the manikin's movable limbs must be identified. Every such bone is manually assigned a list of rotational degrees of freedom, each consisting of a rotation axis relative to the bone, a lower and an upper angular limit. These degrees of freedom should match the real ones to ensure that the set of physically possible manikin poses corresponds to the set of poses output by the algorithm.

Finally, the bones that do not correspond to any of the manikin's movable limbs must be manually rotated so that they match the respective part of the manikin. The algorithm will not determine a rotation for these bones, they will stay fixed relative to their respective parent bone.

Figure 2: A virtual patient model with 25 keypoints (yellow markers, left) manually inserted into the bone hierarchy (right). For each bone that corresponds to a movable limb of the manikin, we also define its rotational degrees of freedom.

3.3 Camera calibration

Making measurements in absolute units (meters, inches, etc.) by analyzing camera images requires the camera to be calibrated so that 2D pixel coordinates can be accurately mapped to 3D rays and vice versa. For pose estimation via keypoint detection in images captured by HMD-attached cameras, we perform a variety of different calibrations.

3.3.1 Intrinsic and extrinsic camera calibration. In the first step, we determine each camera’s projection matrix, as well as lens distortion coefficients and the positions and orientations of all cameras relative to one arbitrarily chosen reference camera. The standard calibration method is to have the cameras capture images of a known calibration target, e.g. a planar chessboard, from a variety of perspectives, and find the pixel coordinates of the inner corners where black and white squares touch. The various parameters of the camera model are then fitted iteratively, minimizing the re-projection error of the corner points. Image processing libraries such as OpenCV [3] provide built-in functions for performing this procedure.

3.3.2 HMD-relative and temporal camera calibration. In order to perform world-space measurements using a camera image, we need to know where the camera was located and how it was oriented when it captured the image. In our particular case, the cameras are attached to the HMD, and we know the HMD location and orientation from its tracking system (e.g. SteamVR tracking). However, we do not know the camera positions and orientations with respect to the local coordinate system of the HMD, i.e. where exactly the cameras are located on the HMD and how they are oriented.

Another unknown parameter is the delay Δt between the tracking system data and the camera images. For example, the Valve Index HMD cameras run at 54 frames per second, while the HMD is tracked at a much higher frequency and with low latency. Therefore, and since the image capturing process takes some time, camera images arriving at the computer at time t should be associated with an HMD position and orientation from slightly in the past at time $t - \Delta t$, found by interpolating between the latest measurement before and the earliest measurement after that point in time (linear interpolation for positions, spherical linear interpolation for orientations).

We find the reference camera’s HMD-relative position and orientation as well as Δt at the same time using non-linear optimization, again finding the parameters that minimize the reprojection error of the calibration target’s corner points. Reprojection means to compute the 2D pixel locations at which the cameras *should* see the corner points as a function of the currently estimated model parameters (HMD-relative position/orientation and Δt) and the already known intrinsic and extrinsic camera parameters. The reprojection error that is subject to minimization is the squared euclidean distance between the reprojected location and the actually observed location. The idea is that smaller reprojection errors can be achieved when the parameters being estimated are closer to their correct values.

The minimization can be achieved using a non-linear optimization algorithm such as the Levenberg-Marquardt algorithm. For the purpose of minimization, orientations should not be represented as 3×3 matrices or quaternions, but instead as rotation vectors, since they are not redundant and thus do not need normalization. A rotation vector is a three-dimensional vector whose direction and length represent the axis and angle of rotation, respectively.

3.4 Body keypoint detection

At runtime, camera images are first undistorted, removing any lens distortions that were determined during calibration, before being

processed by the body keypoint detector, e.g. OpenPose. For the sake of simplicity, we restrict the detector to a single person only and assume that no other people are visible in the images. Body keypoints belonging to other system users can be identified (and thus ignored) by looking at their head (HMD) and hand (controller) positions as reported by the tracking system. For every body keypoint identified, the detector outputs 2D pixel coordinates (see Figure 3) and a confidence score that indicates how confident the detector is that the keypoint is correct.

Figure 3: Body keypoints detected by OpenPose and lines connecting them, overlaid on top of the input image of a medical manikin captured by the stereo camera of a Valve Index HMD.

Typically, body keypoint detectors are trained on images showing people in upright poses (standing, walking, sitting, etc.). Our experiments showed that detection performance can drop significantly when images are rotated by 90° or more, up to the point where people are no longer detected at all. This is problematic for our particular medical training application, since paramedics often look at their patient from above, which can lead to the manikin being “upside down” in the camera images. We therefore feed four rotated variants of each camera image into the body keypoint detector (0° , 90° , 180° , 270°) and select the set of keypoints with the highest sum of confidence score values, rotating the coordinates of the detected keypoints back to 0° , if necessary.

Under some circumstances, body keypoints can be detected incorrectly, e.g. when a part of the body is occluded. Ideally, only correct keypoints should be processed further in order to increase the quality of the solution. With a calibrated multi-camera setup, it is possible to ignore some incorrect keypoints. Given the 2D location of a 3D point observed by one camera, the same point can be observed by the other camera only at certain 2D locations that lie on an *epipolar line*. This *epipolar constraint* serves as a test whether two observed 2D points correspond to the same 3D point. If the constraint is violated, i.e. the observed 2D location is too far away from the epipolar line, then we assume the keypoint to be invalid and remove it. A suitable distance threshold can be determined experimentally.

3.5 Initial estimation of position and orientation

In order for the subsequent non-linear optimization process to converge on a good solution, an initial estimate is required. In experiments, we found that an approximate position and orientation

of the manikin as a whole is sufficient and we do not need to estimate joint angles, i.e. they can remain at their default values.

In order to estimate the manikin's position and orientation, we find the 3D location of each observed body keypoint that has been observed by at least twice from different locations (e.g. in the left and right image of a stereo camera). Such 3D points can be reconstructed from their 2D pixel coordinates in each camera image via triangulation.

We now have an estimated 3D world position for a subset of body keypoints. The next step is to find a translation and rotation for the virtual patient model, in some neutral default pose, so that the distances between its body keypoints and the manikin's estimated body keypoints are minimized. This is a well-known problem called *rigid body transformation estimation* or *rigid point set registration*. For an overview of algorithms for solving such problems, see [7].

3.6 Non-linear optimization of parameters

With an estimated position and orientation of the manikin available, we apply non-linear optimization to find the optimal values of all parameters (position, orientation and joint angles) that minimize the body keypoints' reprojection errors in the camera images. The optimizer must respect the minimum and maximum allowed joint angles while finding a solution. We use the confidence score values reported by the keypoint detector as weights, i.e. reprojection errors of keypoints with high confidence are more important than those of keypoints with low confidence. The error (called residual r) that is subject to minimization is:

$$r = \sum_k \sum_i \|score_{k,i} \cdot (observed_{k,i} - projected_{k,i})\|^2$$

with:

- $score_{k,i}$ and $observed_{k,i}$ being the confidence score and observed 2D position of the manikin keypoint k in image i (determined by the body keypoint detector, a score of zero indicating that the keypoint was not detected),
- $projected_{k,i} = project(keypointPos_k(params), camData_i)$ being the projected 2D position of the virtual patient keypoint k in image i given its 3D position and the camera parameters,
- $keypointPos_k(params)$ being the 3D position of the virtual patient keypoint calculated according to the model parameters,
- $params = \langle manikinPos, manikinRot, \alpha_1, \dots, \alpha_n \rangle$ being the model parameters to be optimized (manikin position, orientation and joint angles), and
- $camData_i = \langle camPos_i, camRot_i, camCalib_i \rangle$ being the position, orientation and calibration parameters of the camera that captured image i .

When the optimizer has converged on a solution for the model parameters, we apply them to the virtual patient model (see Figure 4 for an example).

4 IMPLEMENTATION

In the following, we describe our implementation of the presented approach.

Figure 4: Stereo camera image of the medical manikin (top) and the corresponding HMD view of the virtual patient after body pose estimation using keypoint detection (bottom).

4.1 Hardware

We selected the Valve Index HMD and its controllers for our purposes. Both products use the SteamVR tracking technology. Our medical manikin is a “Code Blue III Adult” from Gaumard Scientific.

The Valve Index HMD is equipped with a 54-fps stereo camera that features a resolution of 960×960 pixels per camera and fish-eye lenses with a large field of view. The distance between the two cameras (baseline) is about 14 cm, which is significantly higher than with the VIVE Pro HMD, for example. A longer baseline improves triangulation accuracy along the depth axis. Accessing the camera images is simple, since the stereo camera presents itself as a single webcam device, placing the images of the left and right camera side-by-side.

The Valve Index controllers can be considered a hybrid of VR gloves and traditional controllers, since they feature a trigger, a stick and buttons, but also include a rudimentary form of finger tracking. A rubber strap keeps the controller firmly in the user's hand so that it is possible for the fingers to reach through it and physically interact with real-world objects such as the medical manikin.

4.2 Software

Our VR training application is developed using the Unity game engine. Calibration and manikin pose estimation are implemented in C++ as separate tools. OpenCV provides the necessary image processing and camera calibration algorithms, while the body keypoint detection is handled by OpenPose, which can detect 25 different body keypoints. For non-linear optimization, we make use of Ceres Solver [1], in particular due to its automatic differentiation feature, which can automatically generate partial derivatives of the to-be-minimized quantity (the reprojection errors, in our case) with respect to the adjustable parameters.

4.3 Code generation for virtual patient keypoint locations

In order to compute the reprojection errors that are subject to minimization, we need to compute the 3D world positions of the keypoints that we manually placed in the virtual patient model, as a function of all the model parameters. That means that, for example, we have to answer the question *“Where is the virtual patient’s nose keypoint in world space, given the model’s position, orientation and joint angles?”*. In the next step we can then compute the pixel coordinates where the cameras *should* see the keypoints and compare them to the actually observed locations.

The 3D world position of a keypoint depends on the transformations of all the bones in the skeleton of which it is a direct or indirect child. For example, the “nose” keypoint depends on the transformations of the bones “head”, “neck”, “spine 2”, “spine 1”, “spine” and “hips” in the standard human skeleton used by Unity and other tools. Each bone has a certain relative offset from its parent as well as a rotation. If the bone corresponds to one of the manikin’s movable limbs, then its rotation is defined by the model parameters, otherwise it is fixed.

Since it would be tedious and error-prone to manually write code to compute all 25 keypoints’ locations, we generate the relevant C++ code from within Unity. For each keypoint placed, we walk upwards in the bone hierarchy, collect the relative transforms of each bone visited and assemble lines of code step by step. If the bone is marked as movable, then its rotation is built from the rotational degrees of freedom associated with it, while also generating code to set the respective rotation limits.

The following excerpt from the code generated for our virtual patient model is intended to demonstrate the usefulness of code generation as opposed to writing such code by hand:

```
// "nose" keypoint world position computation
keypointWorldPosition[Keypoint::Nose] =
rotatePoint(rotatePoint(Vec3{0.9715276, 0.9715276,
0.9715276} * (rotatePoint(rotatePoint(rotatePoint(
rotatePoint(rotatePoint(rotatePoint(rotatePoint(
rotatePoint(Vec3{-0.0013, 0.062, 0.1398},
headRotation3X * xAxis), headRotation2Z * zAxis),
headRotation1Y * yAxis), Vec3{-0.03346, 0, 0}) +
Vec3{0, 0.4738197, 0}, spineRotation3Z * zAxis),
spineRotation2Y * yAxis), spineRotation1X * xAxis
), Vec3{1.581588, 0, 0}) + Vec3{0, 0, 1.11503}),
Vec3{-1.570796, 0, 0}), rotation) + position;

// Minimum and maximum rotation angles
lowerBound[Transform::HeadRotation1Y] = -1.0472;
upperBound[Transform::HeadRotation1Y] = 1.0472;
lowerBound[Transform::HeadRotation2Z] = -0.5236;
upperBound[Transform::HeadRotation2Z] = 0.5236;
lowerBound[Transform::HeadRotation3X] = -0.5236;
upperBound[Transform::HeadRotation3X] = 0.5236;
// ...
```

5 RESULTS AND DISCUSSION

We performed a basic evaluation of our system with respect to processing time and accuracy.

5.1 Processing time

The data reported in the following was collected on a laptop featuring an Intel Core i7-7700HQ CPU (2.8 GHz) with 16 GiB of RAM and a NVIDIA GeForce GTX 1060 with 6 GiB of RAM, running Windows 10. OpenPose 1.7.0, CUDA 10.2 and Ceres Solver 2.0.0 were used.

In our experiments involving a total of 1190 images showing the manikin in different poses and from different perspectives, detecting and filtering keypoints took an average of 116 ms per image. Left and right images of the stereo camera are considered separate images. Also taking into account the four rotation angles tried for every image, this amounts to 928 ms per capture (8 images).

The time spent on non-linear optimization following the keypoint detection depends on the total number of keypoint observations. For example, a stereo image in which all 25 keypoints are detected both left and right results in 50 keypoint observations (assuming no keypoints are rejected during filtering). We measured an average processing time of 0.9 ms per keypoint observation.

Combining both processing steps, we expect a single capture to require a processing time of less than a second. Assuming that the VR application itself utilizes most of the CPU and GPU resources, these results do not allow for real-time tracking of the manikin. Therefore, we decided to use the system in a “static” way in our research project, i.e. the manikin is captured once before starting a training session. The concrete scenario implemented in the project does not necessarily require physical alteration of the patient pose at this point.

5.2 Accuracy

We determined some estimates on the accuracy of the solutions computed by our system in a preliminary evaluation by repeating the following procedure:

- (1) Place the manikin in the room, varying positions, orientations and poses.
- (2) Capture a varying number of images of the manikin using the HMD stereo camera from different perspectives.
- (3) Run the pose estimation algorithm and display the virtual patient model with the parameters applied.
- (4) Touch the real manikin with a specific point on a VR controller at a number of predefined *touch targets* that can be identified easily (shoulders, elbows, wrists, knees, ankles, nose, eyes) and press a button. Each time, the distance between the point on the controller and the touch target on the virtual patient is recorded.

The recorded distances between real and virtual touch targets are an indicator for the quality of the solution. Smaller distances mean that there is less deviation between the manikin and the virtual patient.

We tested five different manikin poses (three sitting, two lying) and measured the distances at least three times for each pose. The number of captures used for pose estimation ranged from 10 to 50. Figure 5 shows a histogram of all measured distances. The majority of the distances were about 4 cm.

Although a user study on user experience could not be conducted so far because of the COVID-19 pandemic, we believe that the current level of accuracy must be improved in order to allow for a

Figure 5: Distribution of distances between real and virtual touch targets, rounded to the nearest value in whole centimeters. 206 distances measured, mean 4.8 cm, median 4.3 cm, standard deviation 2.8 cm.

comprehensively convincing haptic experience that supports fine detailed interaction. At this point, we use the tool primarily to support social interaction and plan examining effects on social presence due to the haptical presence of the manikin, where it matters if practitioners lay a hand on the patient’s shoulder to provide comfort or avoid touching a larger wound (see 6). For more precise medical interaction, e.g. specifically touching smaller body parts or applying treatment with a syringe, the combination of VIVE tracking, virtual patient and manikin pose estimation requires further improvement.

Figure 6: Our method is applied to a complex medical training prototype featuring an animated virtual patient for experimenting with social interaction and coarse wound treatment.

5.3 Ease of use, versatility, cost and complexity

Based on our method, an easy-to-use system can be built, since it only requires the user to capture the manikin using the HMD from a few different perspectives. Our implementation uses a see-through view and captures a stereo image when the trigger of the VR controller is pressed.

Our method can be applied to any medical manikin that looks sufficiently human to the body keypoint detector. Clothes do not

pose a problem since detectors like OpenPose were mainly trained on images of people wearing clothes. The system meets the low-cost and low-complexity requirements since it does not introduce any additional hardware in the form of sensors, markers, etc., provided that the HMD features a built-in camera.

6 FUTURE WORK

Currently, the relatively high processing time prevents the system from achieving real-time update rates. The processing time could be significantly reduced by not having to process four rotations of every camera image all the time. Once an approximate pose of the manikin is found, the ideal rotation of further images could be determined by analyzing the HMD orientation.

While we have not fully analyzed the various sources of errors, several avenues of potentially improving accuracy have been identified and will be explored in future work:

- *Adapting the virtual patient model further to make it more similar to the medical manikin:* If, for example, limbs of the virtual patient and the manikin differ in length, then it is impossible to achieve accurate solutions. The model geometry can be altered manually. Alternatively, the optimizer can be allowed to not only rotate limbs, but also slightly scale them in non-uniform ways, i.e. making them longer or shorter (although symmetry should be respected, e.g. left and right arm should be scaled equally). Another possibility is to use parametric 3D human body models such as STAR [16] or infer both body shape and body pose directly from camera images, as done in [13].
- *Specialization of the body keypoint detector:* The CNN used by OpenPose has been trained on images showing humans mainly in an upright standing pose. By explicitly training it on images showing medical manikins, seated or laying down, also in sideways or upside-down views, its performance could likely be improved. Artificially generating additional training data using computer graphics may also be an option.
- *Improving camera tracking:* Relying solely on the VIVE tracking data in order to retrieve the position and orientation of the HMD may introduce significant errors due to the fact that the exact point in time when an image was captured is unknown. Currently, the capture time can only be estimated. Deviations are especially problematic when the HMD is rotated. Ideally, HMD drivers would provide a timestamp or the HMD position and orientation with each captured camera image. Alternatively, in order to eliminate this time discrepancy, one could determine the position and orientation of the cameras directly from the images, for example by tracking feature points or large-scale markers within the environment.

7 CONCLUSION

In this work, we have analyzed the problem of determining the 3D pose of a medical manikin for the purpose of haptically augmenting a virtual patient in VR medical training. The most relevant requirements concern accuracy, ease of use, versatility, low cost and complexity, and interactive updates rates. We have given an overview of different solution approaches and presented our own

method based on detecting body keypoints of the manikin in HMD camera images and finding suitable pose parameters using non-linear optimization. We have performed a basic technical evaluation using the Valve Index HMD, which features a built-in stereo camera, and discussed the results.

While our current implementation meets our easy-to-use, low-cost and low-complexity requirements, due to limitations in accuracy, at this time we do not recommend using the system for precise medical interactions, but rather for supporting social interaction in VR. Currently, the processing time does not allow for real-time tracking.

Still, we are convinced that the chosen approach based on keypoint detection is advantageous in that it easily supports different models of medical manikins, virtual patients and environments. We are positive that the possible improvements we have identified can significantly improve the update rate and accuracy of the system, eventually allowing for its practical use in medical training.

ACKNOWLEDGMENTS

This work is funded by the German Federal Ministry of Education and Research (BMBF), grant number 01PV18006 (research project “ViTAWiN”).

REFERENCES

- [1] Sameer Agarwal, Keir Mierle, and Others. [n.d.]. Ceres Solver. <http://ceres-solver.org>.
- [2] Omamah Almousa, Joana Prates, Noor Yeslam, Dougal Mac Gregor, Junsong Zhang, Viet Phan, Marc Nielsen, Richard Smith, and Karim Qayumi. 2019. Virtual Reality Simulation Technology for Cardiopulmonary Resuscitation Training: An Innovative Hybrid System With Haptic Feedback. *Simulation and Gaming* 50, 1 (2019), 6–22. <https://doi.org/10.1177/1046878118820905>
- [3] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).
- [4] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. 2019. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Trans. on Pattern Analysis and Machine Intelligence* (2019). <https://doi.org/10.1109/TPAMI.2019.2929257>
- [5] Marco Chirico, Simone Bazurro, Andrea Calandrino, Manuela Chessa, Federica Imperiale, Serena Ricci, Alessio Riitano, Anna Siri, and Giancarlo Torre. 2020. Sharing is caring: A multidisciplinary first aid program. *9th International Workshop on Innovative Simulation for Health Care, IWISH 2020* September (2020), 47–51. <https://doi.org/10.46354/i3m.2020.iwish.009>
- [6] Salam Daher. 2017. Optical See-Through vs. Spatial Augmented Reality Simulators for Medical Applications. In *2017 IEEE Virtual Reality (VR)*. 417–418. <https://doi.org/10.1109/VR.2017.7892354>
- [7] David W. Eggert, Adele Lorusso, and Robert B. Fisher. 1997. Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Mach. Vis. Appl.* 9, 5/6 (1997), 272–290. <https://doi.org/10.1007/s001380050048>
- [8] Mark Fiala. 2005. ARTag, a Fiducial Marker System Using Digital Techniques. In *CVPR (2)*. IEEE Computer Society, 590–596. <https://doi.org/10.1109/CVPR.2005.74>
- [9] Elisa Girau, Fabrizio Mura, Simone Bazurro, Maura Casadio, Marco Chirico, Fabio Solari, and Manuela Chessa. 2019. A Mixed Reality system for the simulation of emergency and first-aid scenarios. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* (2019), 5690–5695. <https://doi.org/10.1109/EMBC.2019.8856777>
- [10] Mahdi Rahmani Hanzaki and Pierre Boulanger. 2020. Proxy Haptics for Surgical Training. In *Proceedings - 2020 22nd Symposium on Virtual and Augmented Reality, SVR 2020*. 134–143. <https://doi.org/10.1109/SVR51698.2020.00033>
- [11] Jason Hochreiter, Salam Daher, Arjun Nagendran, Laura Gonzalez, and Greg Welch. 2015. Touch Sensing on Non-Parametric Rear-Projection Surfaces: A Physical-Virtual Head for Hands-On Healthcare Training. In *2015 IEEE Virtual Reality (VR)*. 69–74. <https://doi.org/10.1109/VR.2015.7223326>
- [12] Matthias Hoppe, Beat Rossmay, Daniel Peter Neumann, Stephan Streuber, Albrecht Schmidt, and Tonja Katrin MacHulla. 2020. A Human Touch: Social Touch Increases the Perceived Human-likeness of Agents in Virtual Reality. *Conference on Human Factors in Computing Systems - Proceedings* (2020), 1–11. <https://doi.org/10.1145/3313831.3376719>
- [13] Muhammed Kocabas, Nikos Athanasiou, and Michael J. Black. 2020. VIBE: Video Inference for Human Body Pose and Shape Estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [14] Aaron Kotranza and Benjamin Lok. 2008. Virtual human + tangible interface = Mixed reality human an initial exploration with a virtual breast exam patient. *Proceedings - IEEE Virtual Reality* (2008), 99–106. <https://doi.org/10.1109/VR.2008.4480757>
- [15] Dieter Lerner, Stefan Mohr, Jonas Schild, Martin Göring, and Thomas Luiz. 2020. An immersive multi-user virtual reality for emergency simulation training: Usability study. *JMIR Serious Games* 8, 3 (2020), 1–10. <https://doi.org/10.2196/18822>
- [16] Ahmed A A Osman, Timo Bolkart, and Michael J. Black. 2020. STAR: A Sparse Trained Articulated Human Body Regressor. In *European Conference on Computer Vision (ECCV)*. 598–613. <https://star.is.tue.mpg.de>
- [17] Pierre Plantard, Edouard Auvinet, Anne-Sophie Le Pierres, and Franck Multon. 2015. Pose Estimation with a Kinect for Ergonomic Studies: Evaluation of the Accuracy Using a Virtual Mannequin. *Sensors* 15, 1 (2015), 1785–1803. <https://doi.org/10.3390/s150101785>
- [18] Jonas Schild, Christian Elsenbast, and Guillermo Carbonell. 2021. ViTAWiN – Developing Multiprofessional Medical Emergency Training with Mixed Reality. In *IEEE SeGAH 2021*.
- [19] Jonas Schild, Dieter Lerner, Sebastian Misztal, and Thomas Luiz. 2018. EPICSAVE - Enhancing vocational training for paramedics with multi-user virtual reality. *2018 IEEE 6th International Conference on Serious Games and Applications for Health, SeGAH 2018* (2018), 1–8. <https://doi.org/10.1109/SeGAH.2018.8401353>